Revista Española de Innovación, Calidad e Ingeniería del Software

REICIS

Volumen 6, Número 3 (especial XI JICS), noviembre, 2010

Web de la editorial: www.ati.es
Web de la revista: www.ati.es/reicis
E-mail: calidadsoft@ati.es
ISSN: 1885-4486

Publicado por la Asociación de Técnicos de Informática (ATI), Via Laietana, 46, 08003 Barcelona.

Secretaría de dirección: ATI Madrid, C/Padilla 66, 3º dcha., 28006 Madrid

Contenidos

REICIS

Editorial

REICIS

El Grupo de Calidad del Software de ATI (www.ati.es/gtcalidadsoft) sigue consolidado su posición como principal promotor de la disciplina de ingeniería y calidad del software con la duodécima edición de las Jornadas sobre Innovación y Calidad del Software (las tradicionales JICS). Con su actividad desde 1997 (véase en su web la sección de actividades que contiene todo su recorrido histórico), el grupo de Calidad del Software de ATI es la entidad más veterana en este ámbito en España. Un año más, las JICS siguen potenciando la presencia iberoamericana en este foro de promoción de la cultura de la calidad del software y de la innovación en el desarrollo de sistemas y aplicaciones en la ya segunda edición de la Conferencia Iberoamericana de Calidad del Software (CICS). Por otra parte, las XII JICS han trabajado con el apoyo de una de las plataformas de innovación en software como es INES (www.ines.org.es/) que ha incorporado una mesa redonda con representantes muy significativos de empresas relacionadas con el desarrollo y la calidad del software.

Como es habitual y gracias a la colaboración del comité de programa, se sigue cuidando la calidad de los trabajos presentados, eminentemente prácticos pero rigurosos, aceptados entre los remitidos en la convocatoria de contribuciones: las ponencias aceptadas (con una tasa de aceptación del 61,5%) han sido sometidos a un completo proceso de revisión. Por supuesto, no cabe olvidar el apoyo de instituciones como la Universidad de Alcalá que ha contribuido a su organización con su apoyo financiero a través de su Vicerrectorado de Investigación sino también de CIFF (www.ciff.net) que ha cedido su sede y ha aportado su máximo apoyo. En definitiva, el evento más completo con toda la información disponible en la página del grupo de Calidad del Software (www.ati.es/gtcalidadsoft) acorde a su trayectoria pionera en España, está proporcionando, a través de la Asociación de Técnicos de Informática, el apoyo para la productividad y la calidad en los proyectos de software, un medio más para remontar la crisis que nos afecta.

Luis Fernández Sanz
Editor

Presentación

REICIS

En este número especial de septiembre de 2009 de REICIS, por segunda vez en la historia de nuestra revista, esta publicación se convierte en el vehículo de difusión del evento decano en España en el ámbito de la ingeniería y la calidad del software: las Jornadas de Innovación y Calidad del Software (JICS) que alcanzan así su undécima edición desde su inicio en 1998. De nuevo, el Grupo de Calidad del Software de ATI (www.ati.es/gtcalidadsoft) ha cuidado que los trabajos aceptados hayan sido sometidos a un completo proceso de revisión por el comité de programa.

Este número especial constituye, en definitiva, la publicación de las actas de las XII JICS y, por ello, cuenta con un tamaño mayor del habitual. Desde aquí agradecemos la labor del comité de programa compuesto por la siguiente lista de expertos:

- M. Idoia Alarcón, Universidad Autónoma de Madrid
- José A. Calvo-Manzano, Universidad Politécnica de Madrid
- M.José Escalona, Universidad de Sevilla
- José Ramón Hilera, Universidad de Alcalá
- Esmeralda Mancheño, Vass
- Antonia Mas, Universitat de les Illes Balears
- Gurutze Miguel, ING
- Andrés Rodríguez, BME
- Pilar Romay, CEU
- Tanja Vos, Universidad Politécnica de Valencia

Luis Fernández Sanz

Taxonomía de factores críticos para el despliegue de procesos software

Sussy Bayona, Jose Calvo-Manzano, Gonzalo Cuevas, Tomás San Feliu
Universidad Politécnica de Madrid, Facultad de Informática, Departamento de Lenguajes y Sistemas Informáticos e Ingeniería del Software
sbayona@mpsei.fi.upm.es, {jacalvo, gcuevas, tsanfe}@fi.upm.es

Resumen

Diversos métodos, modelos y estándares para la mejora de procesos software han sido desarrollados y adoptados por las organizaciones para mejorar sus procesos software. Sin embargo, a pesar de los esfuerzos realizados, presentan aun dificultades en el despliegue de sus procesos a través de la organización. Esto es debido a que en su gran mayoría, los esfuerzos realizados están más orientados hacia los aspectos técnicos, soslayando los aspectos relacionados con las personas. Existe un conjunto de factores que condicionan el éxito del despliegue de los procesos nuevos o que han sido modificados. En este artículo, se presenta una taxonomía de factores críticos que condicionan el éxito del proceso de despliegue, que se traduce en lograr la institucionalización de los procesos. La elaboración de la taxonomía de factores críticos de éxito se sustenta en una revisión sistemática de la bibliografía existente en las bases de datos especializadas y en experiencias en organizaciones que han desplegado procesos basados en el modelo CMMI.

Palabras clave: Taxonomía de factores críticos, despliegue de procesos, CMMI

A taxonomy of the critical success factors for software process deployment

Many methods, models and standards for software process improvement have been developed. They have been adopted by organizations to improve their software processes. However, despite the efforts, they still come up against difficulties in their process deployment through the organization. This is because the vast majority of these efforts focus more on the technical aspects, bypassing aspects related to the people. There is a set of factors that influence the successful deployment of new or modified processes. This paper presents a taxonomy of critical factors in software process deployment, which results in achieving the institutionalization of processes. The development of the critical success factors taxonomy is based on a systematic review of existing literature on specialized databases and industrial experiences that have been deployed processes based on CMMI.

Key words: Taxonomy of critical success factors, process deployment, CMMI.

Bayona, S., Calvo-Manzano, J.A., Cuevas, G. y San Feliu, T. "Taxonomía de factores críticos para el despliegue de procesos software", REICIS, vol. 6, no.3, 2010, pp.6-16. Recibido: 29-10-2010; revisado: 14-11-2010; aceptado: 23-11-2010

1. Introducción

Diversos modelos y estándares han sido creados para la mejora de procesos, sin embargo la implementación de dichos modelos y estándares en las organizaciones presenta dificultades. Entre las dificultades se pueden mencionar: (1) esfuerzos de mejora que no están alineados con los objetivos del negocio, (2) falta de liderazgo y compromiso visible en los esfuerzos de mejora, (3) procesos que no responden a las necesidades del negocio y (4) esfuerzos orientados a los aspectos técnicos dejando de lado las estrategias basadas en los aspectos sociales [1].

Según Niazi [2], el problema de la mejora de procesos no es la falta de estándares o modelos, sino la falta de una estrategia para implementar dichos estándares o modelos. No considerar los aspectos sociales en una estrategia de despliegue de procesos, hace peligrar la institucionalización de los procesos desplegados.

Desplegar procesos basados en cualquiera de los modelos y/o estándares para la mejora de procesos requiere de una estrategia para lograr el uso y la adopción de los nuevos procesos. Esta estrategia debe de estar basada en la gestión del cambio y enfocada fundamentalmente en las personas. Este enfoque facilitará el proceso de transición a los cambios que implica el despliegue de los nuevos procesos, y permitirá minimizar la resistencia a dichos cambios.

A pesar de que lo anteriormente dicho parece tan elemental, al momento de ponerlo en práctica se deja de lado, considerando que el aspecto relacionado con el personal es asunto de otras áreas de la organización.

Se ha detectado que la mayoría de las investigaciones de mejora de procesos están enfocadas en la parte tecnológica, pero pocas mencionan otros factores importantes como la cultura, la gestión del cambio, las personas, la comunicación, y la formación durante y después del proceso de despliegue. Mc. Dermid and Bennet [3] han argumentado que los factores humanos para la mejora de procesos software han sido ignorados y esto ha impactado fuertemente en los procesos de mejora. Según Zahran [4], la falta de adecuación de propuestas de mejora a las necesidades de la organización, en la implementación de la mejora de procesos, es una de las razones más comunes del fracaso de las iniciativas de mejora.

Identificar los factores que condicionan el éxito o fracaso del proceso de despliegue de procesos es fundamental. Sin embargo, es necesario homogeneizar y clasificar dichos factores, los cuales son descritos por diferentes términos según los diferentes autores. Algunos factores, compromiso de la alta dirección, participación del personal (iniciativas de abajo hacia arriba), definición de procesos (procesos fáciles de entender y usar), disponibilidad de recursos (tiempo del personal, recursos), sensibilización (fase de introducción), y comunicación (comunicación efectiva) son conceptos que tienen una aceptación general, pero al nombrarlos se usan diferentes términos.

Es entonces necesario contar con un método para poder clasificar los factores críticos y mantener un lenguaje común en la organización. Para ello, se cuenta con dos fuentes de información: la bibliográfica, realizada a través de una revisión sistemática de artículos y estudios contenidos en las bases de datos bibliográficas, y los factores identificados en las organizaciones desarrolladoras de software, durante el despliegue de sus procesos.

Luego, para identificar estos factores, es necesario revisar la documentación científica resultante de investigaciones empíricas u organizacionales sobre los factores críticos que han condicionado los procesos de mejora y/o despliegue de procesos. Así también, analizar las experiencias de despliegue de las organizaciones para identificar los factores críticos que condicionan el éxito del despliegue de procesos.

Con este objetivo, en el presente artículo se presenta una taxonomía de factores críticos que impactan en el proceso de despliegue y las fases del método desarrollado para construir la taxonomía. Estos factores, una vez identificados y clasificados en una taxonomía, se constituyen en una herramienta útil a ser utilizada en una estrategia de despliegue de procesos.

Este artículo está organizado como sigue. La sección 2 describe el método de investigación para la identificación de los factores críticos del proceso de despliegue, la sección 3 describe la taxonomía de factores críticos del despliegue de procesos y el método utilizado para su construcción, la sección 4 presenta los beneficios de la taxonomía como una herramienta para desarrollar la estrategia de despliegue de procesos y la sección 5 presenta las conclusiones.

2 Método de Investigación

Lograr la institucionalización de los procesos en la organización está condicionado por factores técnicos y factores relacionados con las personas, por lo que es importante, que en la estrategia que usa la organización para desplegar los procesos, estos factores estén identificados y considerados.

Este es el propósito del método de investigación que se ha seguido para identificar los factores críticos en la mejora y/o despliegue y clasificarlos. La investigación se enfoca en conocer qué factores tienen un impacto sobre la mejora y/o despliegue de procesos.

La investigación se ha realizado teniendo en cuenta dos fuentes de información: (1) revisión sistemática de la literatura existente y (2) las experiencias de despliegue en cinco organizaciones desarrolladoras de software, que son descritas a continuación.

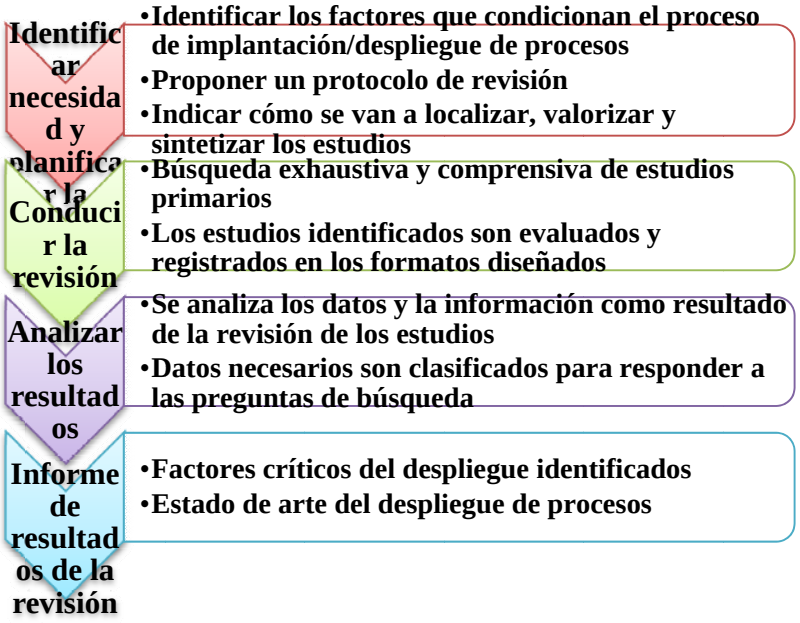


Figura 1. Etapas para realizar la revisión sistemática

a) Factores identificados durante la revisión sistemática de artículos y publicaciones, como resultado de la lectura de artículos, presentaciones, informes técnicos contenidos en bases de datos especializadas como Science@Direct, IEEE Computer, ACM Digital, Springerlink, Institute for Scientific Information(ISI) WEB knowledge, Wiley InterScience; artículos y presentaciones de conferencias

especializadas como Software Engineering Process Group (SEPG) Conference Series del Software Engineering Institute y European Systems & Software Process Improvement and Innovation (EUROSPI), así como informes, artículos y presentaciones de Crostalk, IT Governance y Google Scholar. Todas relacionadas con la mejora de procesos y despliegue o implementación de mejoras de proceso. Para realizar la revisión sistemática se ha seguido el método planteado por Kitchenham [5], Biolchini [6]. La Figura 1 muestra las etapas y actividades del método seguido para realizar la revisión sistemática.

Los resultados de la revisión de los estudios primarios comprendidos desde 1995 hasta 2010, muestran la relevancia y el interés que existe para identificar los factores críticos, que condicionan el éxito de una iniciativa de mejora. Sin embargo, son más los estudios relacionados con los factores para la mejora de procesos que los estudios de factores en el despliegue de procesos.

Del análisis de los estudios revisados, se desprende que el factor compromiso de la alta dirección es crítico en la mejora y despliegue de procesos, así como también la formación y la comunicación. Los factores definición de procesos y participación del personal están muy relacionados, ya que permiten que las personas se sientan motivadas para usar los procesos definidos debido a su participación, a diferencia de que se les imponga el uso de los nuevos procesos. Otro factor mencionado es que la organización cuente con una metodología formal de despliegue de procesos que sirva de guía al proceso de despliegue. Considerando que el despliegue de procesos es un cambio y este cambio debe ser gestionado para minimizar la resistencia del personal, la gestión del cambio es un factor crítico de éxito para el proceso de despliegue. Varios autores coinciden en que las políticas mal diseñadas en la organización es una barrera para la mejora de procesos.

La Tabla 1 muestra un extracto (24 de un total de 60) de factores críticos identificados en la revisión sistemática y su frecuencia en los estudios. Los factores críticos identificados en 26 estudios primarios de la revisión sistemática son registrados en la columna “factores críticos”, usando la denominación dada por el estudio. La columna frecuencia indica el número de veces que aparece el factor crítico en los estudios revisados.

N.	Factores críticos	Frecuencia
1	Compromiso de la alta dirección	25
2	Objetivos de mejora claros, relevantes y aplicables/ objetivos medibles	24
3	Asignación de responsabilidades clara y compensada	13
4	Participación del personal /Iniciativas Bottom-up	23
5	Personas altamente respetadas	11
6	Tiempo del personal y recursos	10
7	Creación de equipos	14
8	Agentes de cambio y líderes de opinión	14
9	Fortalecimiento de la comunicación y la colaboración	20
10	Gestión del proceso de mejora (seguimiento)	12
11	Proveer mayor entendimiento	7
12	Estabilidad en los cambios de los procesos	4
13	Adoptar iniciativas de mejora	7
14	Descongelar la organización	3
15	Políticas de la organización	13
16	Formación	24
17	Disponibilidad de Recursos	18
18	Diferencias de cultura organizacional	13
19	Personal con conocimientos y habilidades	18
20	Incentivos tangibles	4
21	Mentoring	20
22	Sensibilización	7
23	Experiencia del personal	16
24	Metodología formal	10

Tabla 1. Relación de 24 factores críticos identificados y la frecuencia en los estudios

En la Tabla 1 podemos observar como el factor recursos, se presenta como el factor *Tiempo del personal y recursos* y *disponibilidad de recursos.*

b) Factores identificados como resultado de la experiencia de despliegue de procesos en organizaciones desarrolladoras de software que usan como modelo de referencia Capability Maturity Model Integration (CMMI) [7] para la definición de sus procesos. Para identificar los factores que condicionan el despliegue de procesos en las organizaciones, se ha llevado a cabo una investigación en cinco organizaciones distribuidas en América Latina y Europa.

Para identificar los factores se llevaron a cabo las siguientes actividades:

- Definir los objetivos y el alcance de la investigación.
- Identificar los aspectos a investigar y desarrollar el plan de trabajo.
- Identificar a los responsables del despliegue de procesos en las organizaciones.
- Identificar los procesos desplegados en la organización (uso, aceptación, complejidad).
- Desarrollar un cuestionario con preguntas abiertas y cerradas sobre los factores críticos identificados en el proceso de despliegue.
- Conducir la encuesta y las entrevistas en las organizaciones.
- Procesar los datos y analizar los resultados de los cuestionarios, entrevistas, y la revisión de los procesos.

3 Método para la construcción de una taxonomía de factores críticos

Con los resultados obtenidos de las actividades anteriores se cuenta con dos inventarios de factores a considerar en el despliegue de procesos, y es necesario uniformizar y utilizar un lenguaje común.

Para ello, una actividad básica que se ha desarrollado es la elaboración de una taxonomía para identificar y clasificar los factores críticos basada en la revisión sistemática y la experiencia de las organizaciones que han conducido procesos de despliegue, y en el conocimiento de los expertos.

El objeto de la taxonomía es desarrollar una clasificación manejable de los factores críticos del despliegue de procesos y facilitar a las organizaciones la identificación de los factores que pueden afectar al proceso de despliegue y disponer de un inventario de dichos factores identificados.

El resultado de la identificación de factores es una lista conteniendo los factores de éxito que han sido identificados y su categoría correspondiente.

Los objetivos principales para establecer la taxonomía son:

- Servir de apoyo durante la elaboración de la estrategia de despliegue de procesos.
- Facilitar la búsqueda y agrupamiento de la información relevante.

Para establecer esta taxonomía, se ha desarrollado un método basado en la revisión sistemática de métodos y modelos usados para la elaboración de taxonomías [8], [9], [10], [11], [12], [13], [14], [15], [16].

En esta sección, se explicará el método para el diseño de la taxonomía de factores del despliegue de procesos.

El método ha sido desarrollado con el objetivo de que sirva de guía para construir la taxonomía de factores críticos del despliegue de procesos. El método propuesto consta de 5 fases.

1. **Fase 1: Planificación**. Consiste en la planificación del proyecto que tendrá como resultado el diseño e implementación de la taxonomía de factores críticos del despliegue de procesos. Los productos a obtener en esta fase son: (1) Plan de trabajo para la elaboración de la taxonomía; y (2) Equipo de trabajo para la elaboración de la taxonomía.
2. **Fase 2: Identificación y extracción de la información**. El objetivo es alinear el plan de trabajo con la necesidad de información de la organización. En esta fase se identifican las fuentes de información, los términos o variables a usar, las definiciones que formarán parte de la taxonomía, entre otros. La extracción de la información necesaria para la elaboración de la taxonomía puede provenir de fuentes internas y externas. Las fuentes internas son: (1) las revisiones que se realizan con los usuarios respecto a la taxonomía, (2) las encuestas realizadas al personal para identificar las necesidades, (3) las políticas que se seguirán para que la taxonomía tenga un sentido y sea de utilidad para la organización, y (4) la información de los representantes de todas las áreas involucradas. Las fuentes externas son la información proveniente de otras organizaciones como (1) documentación científica relacionada con el tema en estudio y (2) casos de negocios existentes, experiencias similares de otras organizaciones. Los productos a obtener en esta fase son: (1) inventario general de información para la construcción de la taxonomía, (2) políticas de uso de la taxonomía, (3) características de la tecnología a utilizar y (4) lista de representantes de todas las áreas involucradas.
3. **Fase 3: Diseño y construcción de la taxonomía**. El objetivo de esta fase es el diseño y la construcción de la taxonomía haciendo uso del inventario de términos. Identificar el primer nivel de categorización y los demás niveles hasta determinar la estructura final de la taxonomía. Los productos a obtener en esta fase son: (1)

categorización de términos del primer nivel, (2) taxonomía general y (3) diccionario de categorías y subcategorías (metadatos).

4. **Fase 4: Pruebas y validación**. Con el objetivo de asegurar que la taxonomía diseñada sea de utilidad a los usuarios y a los objetivos que plantearon su diseño, se deben realizar las pruebas necesarias y ser validada. Los productos a obtener en esta fase son: (1) taxonomía validada, (2) diccionario de categorías validadas y (3) subcategorías validadas.
5. **Fase 5: Despliegue de la taxonomía**. Con el objetivo de que la taxonomía sea utilizada por los usuarios, ésta debe ser desplegada a través de la organización. Los productos a obtener en esta fase son: (1) formación en la taxonomía y (2) taxonomía disponible a los usuarios.

En la Figura 2 se muestra las principales actividades y productos de las fases del método para elaborar la taxonomía de factores críticos.

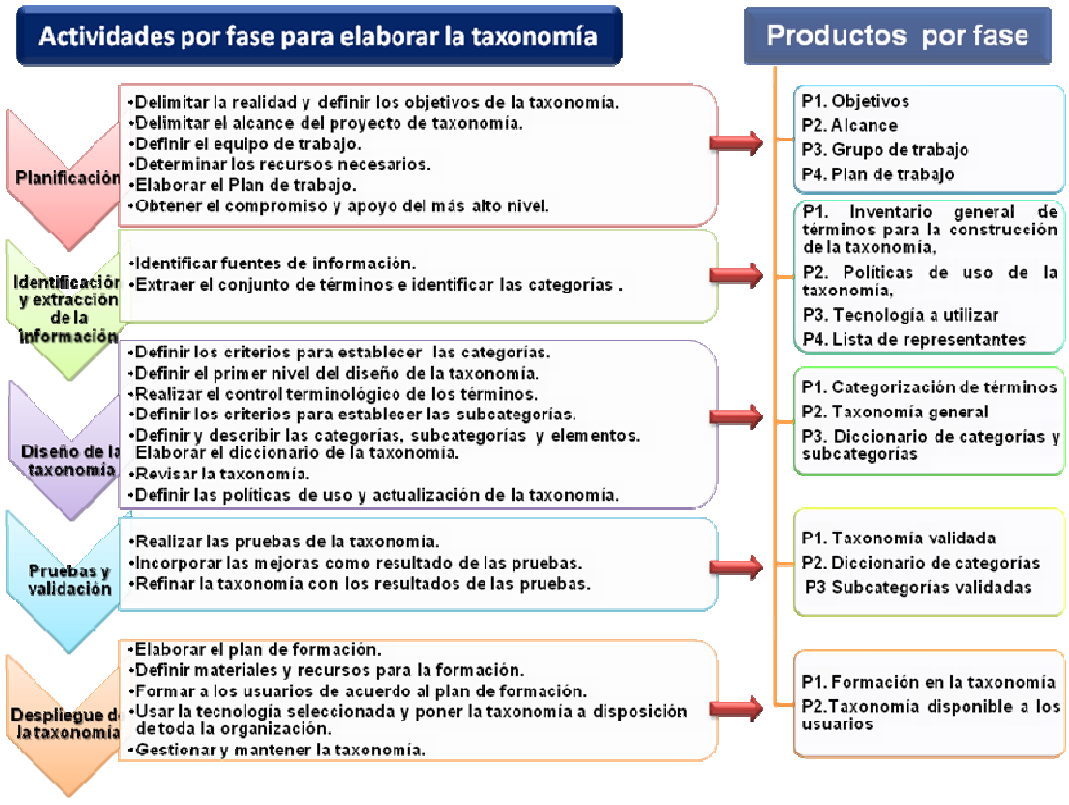


Figura 2. Actividades por fase de la taxonomía y los productos por fase

Como resultado de la aplicación de la taxonomía de factores críticos en el despliegue de procesos, se identifican un número limitado de categorías (ver Figura 3).

Estas categorías fueron definidas después de realizar una revisión de literatura sobre investigaciones de factores críticos en la mejora de procesos y despliegue de procesos, y las clasificaciones utilizadas por los autores [17], [18] [19], [20], [21].

La taxonomía obtenida integra cinco categorías relacionadas con el objeto de estudio. Las categorías identificadas son:

1. **Organización**: muchos factores que no son contemplados en el proceso de despliegue dependen de la organización en la que se va a llevar a cabo el despliegue de procesos.
2. **Personas***:* el despliegue de procesos está basado en las personas en todos sus niveles: individuos, grupos, equipos, organización. etc.
3. **Procesos***:* los procesos son la entrada del despliegue de proceso y éste puede estar condicionada por diversos factores.
4. **Producto***:* el producto de calidad, entregado en tiempo, y con el presupuesto y las funcionalidades requeridas.
5. **Otros**: aquí se contemplan otros factores que no se encuentran en las categorías anteriores.

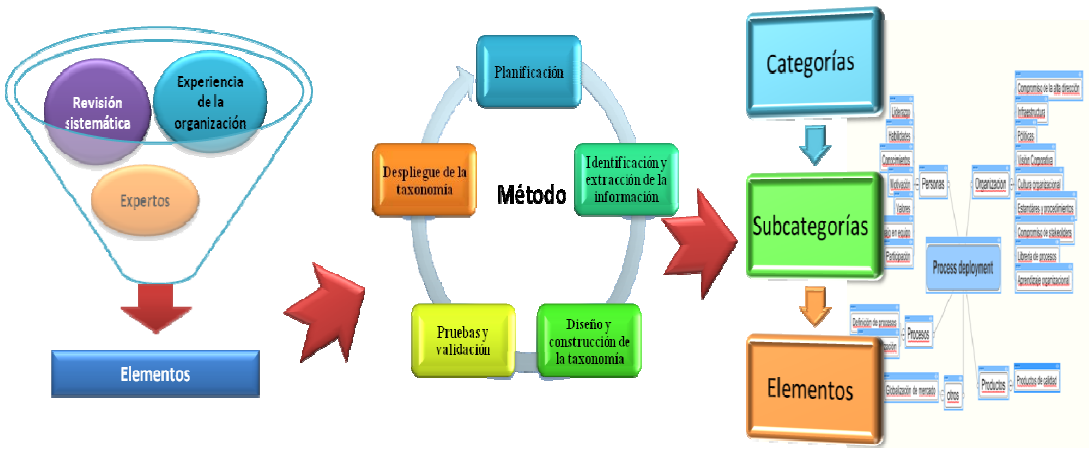


Figura 3. Taxonomía de factores críticos del proceso de despliegue

Una vez identificadas las categorías, se identifica en el inventario de factores los elementos que están relacionados, agrupándolos en subcategorías.

A continuación, se presenta la relación de factores según categoría y las subcategorías identificadas, como resultado del proceso de clasificación.

En la categoría Organización las subcategorías identificadas son:

- ***Compromiso de la Alta Dirección:*** es un factor crítico que debe ser tenido en cuenta antes de iniciar cualquier iniciativa de mejora. El despliegue de los procesos definidos por la organización debe ser considerado un objetivo estratégico de la organización. Es decir, debe haber coherencia entre los objetivos estratégicos de la organización, las metas a alcanzar con el despliegue de procesos y los beneficios para la organización.
- ***Infraestructura:*** cualquier iniciativa de despliegue de procesos que no cuente con los recursos necesarios, como presupuesto, personal calificado para el desarrollo de las tareas, recursos materiales, técnicas y tecnología necesaria a desplegar está orientada al fracaso.
- ***Políticas:*** la falta de políticas claras en la organización es un problema y el exceso de políticas también lo es. Por lo que se debe tener cuidado al establecerlas, cuidando de que sean las necesarias y factibles de implementar.
- ***Visión corporativa:*** contar con una visión corporativa y establecer los mecanismos que permitan que todos los colaboradores se alienen con ella y apunten a esa dirección.
- ***Cultura organizacional:*** es importante reconocer las costumbres, creencias y mitos de la organización en las que se desplegarán los procesos y deben ser consideradas al definir los procesos y en el proceso de despliegue.
- ***Estándares y procedimientos:*** contar con estándares y procedimientos es fundamental para la organización porque de esta forma se gestiona el conocimiento de la organización.
- ***Compromiso de las partes interesadas:*** responder a sus requerimientos creando un mecanismo de diálogo y entendimiento mutuo para que las necesidades de todos los grupos sean tenidas en cuenta.

- ***Aprendizaje organizacional:*** retroalimentación con lecciones aprendidas. Aprender de los errores para no volverlos a repetir. Incrementar el aprendizaje de la organización compartiendo experiencias y la retroalimentación de las lecciones aprendidas.

En la categoría Personas, las subcategorías identificadas son:

- ***Liderazgo:*** para conducir el proceso de despliegue es necesario que el personal posea liderazgo. Es decir, personas con capacidad de influir en el personal y de motivarlos para el logro de los objetivos.
- ***Habilidades:*** es el grado de competencia de una persona para alcanzar un objetivo, es una aptitud innata de la persona.
- ***Comunicación:*** la comunicación es un elemento fundamental que permite mantener al personal informado de las actividades y resultados obtenidos del proceso de despliegue. Elimina los temores y malentendidos del personal.
- ***Conocimientos:*** el conocimiento es el conjunto de información almacenada a través del proceso de aprendizaje o la experiencia. El proceso de despliegue requiere de personal con conocimientos de modelos y procesos software, de gestión de proyectos software, experiencia en definir procesos (tácito), de técnicas de comunicación, de técnicas de medición, de herramientas estadísticas de acuerdo al rol que le competa desempeñar.
- ***Actitud:*** actitud de aprender, auto motivación constante, motivación por la calidad, capacidad para motivar a las personas, actitud positiva al cambio, motivación para el uso de los procesos.
- ***Valores:*** podemos mencionar: la ética en el trabajo, productividad, colaboración, alto nivel de compromiso por las actividades encomendadas, colaboración, compromiso con el logro de la calidad y satisfacción del cliente, sentido de pertenencia y responsabilidad en sus acciones.
- ***Formación:*** es importante que la organización cuente con capacidad para gestionar la formación de los colaboradores, mediante un programa de formación. Éstos deben responder a las necesidades de la organización y, para ello, se debe contar con instructores cualificados y con experiencia. Se apoya en el material de

formación. La falta de formación en los cambios realizados, no permite que las personas obtengan las habilidades para desarrollar las nuevas tareas.

- ***Trabajo en equipo:*** el trabajo en equipo con una *visión* compartida del equipo orienta los esfuerzos del grupo hacia las metas y objetivos, logrando un trabajo colaborativo. Las directrices deben ser uniformes para todo el equipo para evitar diferencias y posibles descontentos de los miembros.
- ***Participación:*** es el grado de participación del personal en las actividades del despliegue. La participación del personal en la definición de los procesos es clave porque permite que se sientan parte del proceso y se minimice la resistencia a los cambios.
- ***Roles y responsabilidades:*** definir los roles y responsabilidades de los miembros de un proyecto facilita el desarrollo de las actividades, caso contrario se produce el caos durante el desarrollo de las actividades.

En la categoría Procesos, las subcategorías identificadas son:

- ***Definición de los procesos:*** los procesos deben responder a las necesidades de la organización y estar alineados con los objetivos estratégicos de la organización. Pueden estar elaborados haciendo uso de un modelo de procesos de referencia o como una mezcla de varias que se complementen.
- ***Librería de procesos:*** disponer de una plataforma automatizada que contenga la librería de procesos permite que los procesos estén organizados y accesibles a todas las partes interesadas en el proyecto. La librería debe mantenerse actualizada y la gestión de cambios de versiones es crucial.
- ***Institucionalización:*** es el objetivo del despliegue de procesos, consiste en que los procesos sean utilizados y adoptados por los usuarios en toda la organización. Los niveles de aceptación y uso de los procesos desplegados serán los indicadores del logro de los objetivos.
- ***Despliegue de procesos:*** contar con una estrategia de despliegue de la librería de procesos permite desarrollar un conjunto de actividades de manera sistemática para lograr que los procesos sean usados e institucionalizados.

En la categoría Producto, las subcategorías identificadas son:

- ***Auditoría y calidad:*** la auditoría y calidad de los procesos permite evaluar si se están ejecutando los procesos de acuerdo a como han sido definidos. Así mismo, es necesario verificar si están siendo usados por el personal de acuerdo a lo establecido en el proceso.
- ***Calidad del producto:*** productos en tiempo, coste y funcionalidades de acuerdo a las necesidades del usuario.

En la categoría de Otros, la subcategoría identificada es:

- ***Globalización de Mercado:*** el proceso de despliegue debe tener en cuenta la diferencia de lenguajes y diferencias de zonas horarias, así como la cultura del lugar en el que se está llevando a cabo el proceso de despliegue.

Para analizar la presencia de los factores a nivel de subcategorías de la taxonomía, se ha llevado a cabo una investigación en cinco organizaciones que cumplían con los siguientes criterios definidos para su selección: (1) que estén dedicadas al desarrollo y mantenimiento de software, (2) que hayan implementado modelos de madurez, (3) que estén distribuidas en países con diferentes culturas, (4) que sean empresas medianas o grandes, y (5) que cuenten con una librería de procesos para el desarrollo de proyectos software.

Con este propósito, se ha elaborado un cuestionario sobre factores críticos y su impacto en el despliegue de procesos. El cuestionario fue enviado a los responsables de despliegue de procesos de estas organizaciones. Además se establecieron entrevistas con los responsables de despliegue de procesos y los responsables de calidad de cada organización, usando diferentes medios de comunicación (teleconferencia, chat, teléfono).

Los resultados del análisis de los datos provenientes de los cuestionarios se presentan en la Tabla 2, que muestra la taxonomía de factores críticos a nivel de subcategorías y su presencia en el proceso de despliegue de procesos en cada una de las organizaciones analizadas.

Del análisis de los resultados de las entrevistas realizadas, se observa que aquellas organizaciones que tomaron en cuenta los factores para definir su procedimiento o estrategia de despliegue han logrado que sus procesos sean usados y adoptados en un mayor porcentaje. Las organizaciones que para la definición de sus procesos contaron con la participación del personal, no presentaron resistencia de parte del personal.

Categorías/subcategorías	Organizaciones				
	1	2	3	4	5
1. Organización					
• Compromiso alta dirección	√	√	√	√	√
• Infraestructura	√	√	-	√	√
• Políticas	√	√	-	√	√
• Visión Corporativa	√	√	-	√	
• Cultura Organizacional	√	√	√	√	√
• Estándares y procedimientos	√	√	√	√	√
• Compromiso de partes interesadas	√	√	-	√	-
• Aprendizaje organizacional	√	√	-	√	-
2. Personas					
• Liderazgo	√	√	√	√	√
• Habilidades	√	√	√	√	√
• Comunicación	√	√	√	√	√
• Conocimientos	√	√	√	√	√
• Motivación	√	-	√	-	-
• Valores	√	-	√	-	√
• Formación	√	√	√	√	√
• Trabajo en equipo	√	√	√	√	√
• Participación	√	-	-	√	-
• Gestión del cambio	√	-	√	-	-
• Roles y responsabilidades	√	√	√	√	-
3. Procesos					
• Definición de los procesos	√	-	-	√	-
• Librería de procesos	√	√	√	√	√
• Institucionalización	√	-	√	-	√
• Despliegue de procesos	√	-	-	√	-
4. Producto					
• Auditoria de calidad	√	√	√	√	√
5. Otros					
• Globalización del mercado	√	-	-	-	-

Tabla 2. Factores identificados en organizaciones dedicadas al desarrollo y mantenimiento de software

4. Beneficios de la taxonomía de factores críticos

Uno de los beneficios de la taxonomía es que una vez identificados los factores críticos del despliegue de proceso, se está en condiciones de plantear un método para el despliegue de procesos que integre estos factores, y asegurar el proceso de despliegue. La taxonomía de factores basada en la investigación de experiencia en las organizaciones y en la revisión sistemática, ha permitido identificar los factores relacionados con los aspectos técnicos y los aspectos sociales que deben ser incorporados en una estrategia de despliegue a fin de lograr la institucionalización de los procesos.

En este punto es importante aclarar que el método tomará algunos factores como precondiciones, antes de dar inicio al despliegue de procesos en la organización, tales como:

- Obtener el compromiso de las partes interesadas claves en el proyecto.
- Contar con el compromiso de la alta dirección al más alto nivel.
- Contar con los recursos necesarios para llevar a cabo el despliegue de los procesos.
- Alinear de forma estratégica los procesos con las necesidades del negocio.
- Tener el liderazgo para llevar a cabo el proceso del despliegue.
- Tener muy claro los motivos del cambio y el por qué del cambio.
- Tener claro los objetivos y los beneficios a alcanzar, tanto para la organización como para los empleados, del cambio a realizar.
- Contar con procesos definidos y medidos adaptados a las necesidades de la organización en la que se va a implementar (incluyendo las guías de adaptación).
- Contar con una biblioteca de procesos automatizada que permita el acceso y uso en línea de los participantes del despliegue.

5. Conclusiones

A pesar de la existencia de diversos métodos, modelos y estándares para la mejora de procesos software, se presentan dificultades durante la implementación de los procesos, debido a que las organizaciones cuando van a implementar los procesos, en su mayoría, están enfocadas en resolver los aspectos técnicos y dejan de lado otros factores relacionados con los aspectos sociales. Específicamente los factores relacionados con las

personas, que son las que ejecutan las actividades.

Identificar los factores que condicionan el éxito o fracaso del proceso de despliegue de procesos es fundamental. Sin embargo, es necesario homogenizar y clasificar dichos factores.

Diversos autores han clasificado los factores críticos, para la mejora de procesos. Sin embargo, no existe evidencia de la clasificación de factores para el despliegue de procesos, lo que ha motivado el interés por contar con una taxonomía de factores con enfoque en el despliegue de procesos, que incorpore factores como definición de procesos, despliegue de procesos, librería de procesos e institucionalización.

En el presente artículo, se ha recurrido a dos técnicas: (1) revisión de la literatura existente sobre factores críticos de éxito en el proceso de mejora y despliegue de procesos, obteniéndose un inventario de factores que luego de un proceso de revisión y clasificación de dichos factores se obtiene la taxonomía y (2) el análisis de la presencia de los factores contenidos en la taxonomía a nivel de subcategorías, en organizaciones de software.

Contar con un método para la elaboración de la taxonomía nos ha permitido ordenar y clasificar los factores críticos del proceso de despliegue, uniformizar los conceptos y que éstos puedan ser incorporados en una estrategia que considere los factores enfocados en las personas.

La taxonomía de factores, basada en la investigación de experiencia en las organizaciones y en la revisión sistemática, ha permitido comprobar la necesidad de que se incida no sólo en aspectos técnicos, sino incorporar aspectos sociales a fin de lograr la institucionalización de los procesos.

Agradecimientos

Este trabajo ha sido patrocinado por Everis Foundation y la Universidad Politécnica de Madrid mediante la Cátedra de Mejora del Proceso Software en el Espacio Iberoamericano.

Referencias

[1] Messnarz R., Ekert D., Reiner M., O'Suilleabhain G.., "Human resources based improvement strategies the learning factor", *Software Process: Improvement and Practice,* vol. 13, nº 4, pp 355-362, 2008.

[2] Niazi, M., Wilson D., Zowghi, "A framework for assisting the design of effective software process improvement implementation strategies", *Journal of Systems and Software*, vol.78, nº 2, pp. 204-222, 2005.

[3] Mc Dermid J., Bennet K., "Software Engineering research a critical appraisal", *IEEE proceedings on Software Engineering*, pp. 179-186, 1999.

[4] Zahran S. Software Process Improvement. Practical Guidelines for Business Success. Addison-Wesley, 1998.

[5] Kitchenham B., "*Guidelines for performing Systematic Literature Reviews in software engineering*", EBSE Technical Report EBSE-2007-01, Keele University, 2007

[6] Biolchini, J., Gomes, M., Cruz, N., Horta T., *Systematic Review in Software Engineering*, Technical Report RT-ES679/05, Software Engineering and Computer Science Department, 2005.

[7] Chrissis M., Konrad M., Shrum S., *CMMI Guidelines for Process Integration and Product Improvement (Second Edition)*, Addison Wesley, 2007

[8] Valerio G., "*Desarrollo de taxonomías*", 2003. http://memoriadigital.lacoctelera.net/post/2006/04/05/desarrollo-taxonomias, consultado el 24 de junio 2009.

[9] Crandall M. *Using Taxonomies Effectively in the Organization. Microsoft Information Services,* 2000, www.infotoday.com/kmworld2000/presentations/crandall.ppt, consultado el 9 de junio 2009.

[10] Craig, S., *A Taxonomy of Information Systems Audits, Assessments and Reviews*, SANS Institute, 2007.

[11] Verity *Classification, Taxonomies and You,* Verity White Paper, 2004, http://www.weitkamper.com/download/verity/verity_mk0648.pdf, consultado el 6 de junio 2009.

[12] Debar, H. Dacier, Wespi A., "A revised taxonomy for intrusion-detection systems", *Journal Annals of Telecommunications*, vol. 55, nº 7, pp. 361-378, 2000.

[13] Graef J., "*Managing taxonomies strategically*" Montague Institute Review, 2001. http://www.montague.com/abstracts/taxonomy3.html, consultado el 9 junio 2009.

[14] Bruno D., Richmond H., "The truth about taxonomies". *Information Management Journal,* Vol. 37, nº 2, pp. 48-52, 2003.

[15] Ayala, C., "Systematic Construction of Goal-Oriented COTS Taxonomies", *Proceedings of the 3rd Doctoral Consortium at the 18th Conference on Advanced Information Systems Engineering,* Luxembourg 5-9 June, 2006,

[16] Centelles M. Taxonomías para la categorización y la organización de la información en sitios Web, 2005. http://www.hipertext.net, consultado 30 de enero de 2007.

[17] Kaltio T., Kinnula A, "Deploying the defined software process". *Software Process Improvement and Practice,* vol 5, pp. 65–83, 2000.

[18] Hantos, P., Gisbert, M., 2000. Identifying software productivity improvement approaches and risks: construction industry case study. *IEEE Software* vol. 17, nº 1, pp. 48–56, 2000.

[19] Hall T., Rainer A. and Baddoo N., "Implementing Software Process Improvement: An Empirical Study", *Software Process Improvement and Practice*, vol. 7, pp. 3-15, 2002.

[20] Guerrero F., Eterovic Y., "Adopting the SW-CMM in a Small IT Organization", *IEEE Software* julio-agosto, pp 29-35, 2004.

[21] Wilson, D., Hall, T. and Baddoo, N., "A framework for evaluation and prediction of software process improvement success", *Journal of Systems and Software*, vol. 59, nº 2, pp. 135–142, 2007.

Sistema de Gestión Integrado según las normas ISO 9001, ISO/IEC 20000 e ISO/IEC 27001

Antoni Lluís Mesquida, Antònia Mas, Esperança Amengual, Ignacio Cabestrero
Departamento de Matemáticas e Informática, Universitat de les Illes Balears
{antoni.mesquida, antonia.mas, eamengual}@uib.es, nacho.cabestrero@gmail.com

Resumen

Dada la gran aceptación que tuvo en su momento la implantación de un Sistema de Gestión de Calidad (SGC) de acuerdo con la norma ISO 9001, actualmente la mayoría de organizaciones que deciden implantar una nueva norma para gestionar sus servicios, como ISO/IEC 20000, o la seguridad de su información, como ISO/IEC 27001, normalmente ya cuentan con un SGC basado en ISO 9001. Con el objetivo de facilitar a las empresas la implantación de estas normas se ha realizado un estudio, tanto para analizar las posibles relaciones existentes entre los requisitos de los sistemas de gestión propuestos por estas normas, como para identificar los requisitos no compartidos entre ellos. En este artículo se presenta un nuevo Sistema de Gestión Integrado que amplía los requisitos de un SGC según ISO 9001 con los requisitos específicos de los otros dos estándares antes mencionados.

Palabras clave: Sistema de Gestión Integrado, Sistema de Gestión de Calidad (SGC), ISO 9001, ISO/IEC 20000, ISO/IEC 27001, ISO/IEC 9004.

Integrated Management System according to ISO 9001, ISO/IEC 20000 and ISO/IEC 27001

Abstract

As a consequence of the successful implementation of Quality Management Systems (QMS) according to ISO 9001 performed some years ago by an important number of organizations, nowadays when these companies decide to implement new standards, such as ISO/IEC 20000 to manage their services or ISO/IEC 27001 to manage information security, they have already implemented a QMS according to ISO 9001. In order to facilitate the implementation of these new standards to these organizations a study has been carried out, both to analyse the existing relations between the requirements proposed by these standards and to identify other unshared requirements. This paper presents a new Integrated Management System which widens the scope of the ISO 9001 QMS with the specific requirements of the two above mentioned standards.

Key words: Integrated Management System, Quality Management System (QMS), ISO 9001, ISO/IEC 20000, ISO/IEC 27001, ISO/IEC 9004.

Mesquida, A.L., Mas, A., Amengual, E., Cabestrero, I., "Sistema de Gestión Integrado según las normas ISO 9001, ISO/IEC 20000 e ISO/IEC 27001", REICIS, vol. 6, no.3, 2010, pp.25-34. Recibido: 8-11-2010; revisado: 14-11-2010; aceptado: 24-11-2010

1. Introducción

Numerosas organizaciones del sector TI han optado por la implantación de sistemas de gestión con el objetivo de garantizar la eficacia y fiabilidad de sus procesos de negocio. Estas organizaciones normalmente han implantado sus sistemas de gestión de calidad, gestión de servicios y seguridad de la información, entre otros, de forma independiente o escasamente integrada. Sin embargo, en todos los sistemas de gestión existen ciertos elementos comunes que pueden ser gestionados de un modo integrado.

La necesidad de obtener una visión global de los sistemas de gestión, haciéndolos compatibles entre sí, de forma que permita establecer unos objetivos alineados y facilite la toma de decisiones, ha provocado que muchas organizaciones del sector se hayan cuestionado la existencia de sistemas separados y deseen integrar sus sistemas de gestión.

Durante los últimos años han surgido diversas iniciativas para ofrecer una solución a la falta de integración entre los sistemas de gestión. En ese sentido, PAS 99:2006 *Specification of common management system requirements as a framework for integration* [1], publicada por el *British Standards Institution* (BSI), es el primer marco que ha sido elaborado y constituye una especificación de requisitos comunes de los sistemas de gestión. Este estándar británico fue desarrollado como respuesta a la demanda del mercado de alinear los procesos y procedimientos en una estructura que permitiera operar con mayor eficacia. PAS 99 organiza sus requisitos bajo las seis categorías de elementos comunes propuestos en la ISO Guide 72:2001 *Guidelines for the justification and development of management system standards* [2], una norma para justificar y desarrollar estándares de sistemas de gestión.

En 2008, la *International Organization for Standardization* (ISO) publicó *The integrated use of management system standards* [3] que ofrece una guía sobre cómo integrar los requisitos de múltiples estándares de sistemas de gestión, ISO o no ISO, con el sistema de gestión de una organización.

En España, AENOR publicó en 2005, la norma UNE 66177:2005 [4] con la intención de proporcionar directrices para desarrollar, implantar y evaluar procesos de integración de los sistemas de gestión existentes en una organización.

Por otra parte, nuestro grupo de investigación, MiProSoft, en el año 2002 desarrolló un modelo de implantación simultánea de las normas ISO 9001 e ISO/IEC 15504 durante el

proyecto QuaSAR I [5] [6]. A lo largo de estos años hemos seguido trabajando en esta misma línea, y nuestro último resultado está relacionado con facilitar la implantación de la norma ISO/IEC 27001:2005 [7] en una empresa que ya tiene un nivel de madurez 1 según ISO/IEC TR 15504-7:2008 (Pathfinder) [8].

El propósito de este artículo es identificar e interpretar los requisitos del sistema de gestión de calidad de tres normas ISO, analizando tanto los requisitos que estas normas tienen en común como los requisitos no compartidos entre ellas. Concretamente las normas consideradas son las siguientes:

- ISO 9001:2008 Sistemas de gestión de la calidad - Requisitos [9]
- ISO/IEC 20000-1:2005 Tecnología de la información - Gestión del servicio - Parte 1: Especificaciones [10] e
- ISO/IEC 27001:2005 Tecnología de la información - Técnicas de seguridad - Sistemas de Gestión de la Seguridad de la Información (SGSI) - Requisitos [11].

El objetivo final de la investigación en el que se enmarca este trabajo es llegar a elaborar una guía completa para la alineación y/o integración de los sistemas de gestión de las tres normas anteriores que permita la implantación efectiva de más de una de ellas en una organización.

2. Estándares de sistemas de gestión analizados

2.1. ISO 9001:2008 Sistemas de gestión de la calidad - Requisitos

Esta norma internacional especifica los requisitos de un sistema de gestión de la calidad. La edición actual, ISO 9001:2008, sustituye a la tercera edición (ISO 9001:2000) que ha sido modificada para clarificar puntos en el texto y aumentar la compatibilidad con otras normas.

ISO 9001:2008, en su apartado 0.4 Compatibilidad con otros sistemas de gestión, hace referencia a la integración con otros sistemas de gestión: “Esta norma internacional no incluye requisitos específicos de otros sistemas de gestión, tales como aquellos particulares para la gestión ambiental, gestión de la seguridad y salud ocupacional, gestión financiera o gestión de riesgos. Sin embargo, esta norma internacional permite a una organización alinear o integrar su propio sistema de gestión de la calidad con requisitos de sistemas de gestión relacionados. Es posible para una organización adaptar su(s) sistema(s) de gestión

existente(s) con la finalidad de establecer un sistema de gestión de la calidad que cumpla con los requisitos de esta norma internacional".

Asimismo, se han desarrollado normas específicas que constituyen una guía para la aplicación de la norma ISO 9001 en un determinado campo:

- ISO/IEC 90003:2004 *Software engineering - Guidelines for the application of ISO 9001:2000 to computer software* [12] proporciona una guía para la aplicación de ISO 9001 en la adquisición, suministro, desarrollo, operación y mantenimiento del software.
- ISO/IEC TR 90005:2008 *Systems engineering - Guidelines for the application of ISO 9001 to system life cycle processes* [13] hace lo propio con el área de sistemas.

2.2. ISO/IEC 20000-1:2005 Tecnología de la información - Gestión del servicio - Parte 1: Especificaciones

La norma ISO/IEC 20000-1 promueve la adopción de un enfoque de procesos integrados, para una provisión eficaz de servicios gestionados que satisfaga los requisitos del negocio y de los clientes.

Con la intención de desarrollar una guía para la aplicación de la norma ISO 9001 a la gestión de servicios de TI, ISO inició en 2008 un nuevo proyecto denominado ISO/IEC NP 90006 *Information technology - Guidelines for the application of ISO 9001:2000 to IT service management* [14]. Sin embargo, trascurridos dos años, el proyecto sigue en la misma fase inicial.

2.3. ISO/IEC 27001:2005 Tecnología de la información - Técnicas de seguridad - Sistemas de Gestión de la Seguridad de la Información (SGSI) - Requisitos

Esta norma internacional proporciona un modelo para la creación, implementación, operación, supervisión, revisión, mantenimiento y mejora de un Sistema de gestión de Seguridad de la Información (SGSI).

ISO/IEC 27001, en su apartado 0.3 Compatibilidad con otros sistemas de gestión, asegura que esta norma internacional sigue las pautas marcadas en las normas ISO 9001:2000 e ISO 14001:2004 para asegurar una implementación integrada y consistente con las mencionadas normas de gestión. Esta norma internacional está diseñada para posibilitar a una organización el adaptar su SGSI a los requisitos de los sistemas de gestión

mencionados. Más concretamente, en el anexo C.1 de la misma se muestra la relación entre este estándar y la norma ISO 9001:2000.

3. Estudio de las relaciones entre los sistemas de gestión de las tres normas

Después de años de relaciones y de trabajo continuado con las empresas del sector TIC de nuestro entorno más próximo en la aplicación de resultados de investigación, hemos podido observar que, en la mayoría de casos, cuando una empresa decide implantar una norma relativa a la gestión de servicios o a la gestión de la seguridad de la información, ya ha tenido otras experiencias previas, principalmente, con la implantación de alguna versión de la norma ISO 9001. Por este motivo, para realizar el trabajo que se presenta en este artículo, hemos partido de la norma ISO 9001:2008 y hemos realizado una investigación en dos frentes:

- Por una parte hemos analizado las relaciones y correspondencias entre los requisitos de los sistemas de gestión de la norma ISO 9001 y los de la norma ISO/IEC 20000-1.
- Por otra parte, hemos hecho lo mismo entre la norma ISO 9001 y la norma ISO/IEC 27001.

Ambos estudios han sido llevados a cabo siguiendo una estrategia iterativa, en la cual, cada uno de los requisitos de la normas ISO/IEC 20000-1 e ISO/IEC 27001 han sido comparados con todos los requisitos de la norma ISO 9001. Durante el análisis, se han establecido tres tipos distintos de correspondencias que se detallan en el siguiente apartado.

3.1. Tipos de relaciones

- **Relación total**. El requisito de la norma ISO/IEC 20000-1 o ISO/IEC 27001 en cuestión ya está contemplado por algún requisito de la norma ISO 9001. En este caso, al definir el nuevo sistema de gestión integrado, no se deberá añadir ningún aspecto específico, referente a la gestión de servicios de TI o a la seguridad de la información, al SGC ya implantado.

  Un ejemplo de este tipo de relación es el siguiente: La norma ISO/IEC 20000-1 trata los requisitos de la documentación en su apartado 3.2: "Los proveedores del servicio deben facilitar documentos y registros para asegurar una planificación,

operación y control de la gestión del servicio efectivas". Sin embargo, este requisito ya queda cubierto por la norma ISO 9001 en su apartado 4.2.1.d: "La documentación del sistema de gestión de la calidad debe incluir los documentos, incluidos los registros que la organización determina que son necesarios para asegurarse de la eficaz planificación, operación y control de sus procesos". Del mismo modo, la norma ISO/IEC 27001 también cubre este requisito en su apartado 4.3.1.g: "Los procedimientos documentados que necesita la organización para asegurar una correcta planificación, operación y control de sus procesos de seguridad de la información, y para describir cómo medir la eficacia de los controles".

- **Relación parcial**. El requisito de la norma ISO/IEC 20000-1 o ISO/IEC 27001 en cuestión amplía algún requisito de la norma ISO 9001 con aspectos propios de la gestión de servicios de TI o de la seguridad de la información.

  Un ejemplo de este tipo de relación es el caso de los requisitos relacionados con el compromiso de la dirección, recogidos en el apartado 5.1 de la norma ISO 9001. En este caso, los requisitos del SGC de la norma ISO 9001 deben ser ampliados con aspectos específicos de la gestión de servicios, recogidos en los apartados 3.1.e, f y g de la norma ISO/IEC 20000-1, y con aspectos específicos de la seguridad de la información, detallados en el apartado 5.1.c de la norma ISO/IEC 27001.

- **Inexistencia de relación.** Cuando las normas ISO/IEC 20000-1 o ISO/IEC 27001 añaden requisitos propios de la gestión de servicios de TI o de la seguridad de la información.

  Un ejemplo de este caso se da en los apartados 4.1, 4.2, 4.3 y 4.4 de la norma ISO/IEC 20000-1 en los que se amplían, con aspectos específicos de la gestión de servicios de TI, las definiciones de las cuatro etapas de la metodología Planificar-Hacer-Verificar-Actuar (PHVA), introducidas en el apartado 0.2 de la norma ISO 9001.

3.2. Elaboración de guías de implantación de sistemas de gestión

A partir de las relaciones extraídas en el estudio, se han elaborado dos guías de implantación de sistemas de gestión. La primera de ellas ha sido diseñada con el objetivo de

implantar el sistema de gestión de servicios de TI que propone la norma ISO 20000-1 a partir del sistema de gestión de la norma ISO 9001. La segunda guía pretende facilitar la implantación del sistema de gestión de seguridad de la información de la norma ISO/IEC 27001 a partir del sistema de gestión de la norma ISO 9001. Ambas guías guardan el mismo formato que las guías de aplicación de ISO 9001 referenciadas anteriormente [12] [13] [14].

4. Un nuevo sistema de gestión integrado

El sistema de gestión integrado resultante de esta investigación parte de los requisitos del sistema de gestión de calidad propuestos por la norma ISO 9001 y los amplía con todos los requisitos específicos de gestión de servicios de TI y de gestión de seguridad de la información que describen las normas ISO/IEC 20000-1 e ISO/IEC 27001.

La tabla 1 muestra en la primera columna únicamente los requisitos de la norma ISO 9001 que deberían ser ampliados con requisitos propios de gestión de servicios, especificados en la segunda columna, y con requisitos propios de la seguridad de la información, que se muestran en la tercera columna de la tabla. Es decir, la tabla recoge todas las relaciones parciales detectadas descritas en el apartado 3.1.

Es importante observar que el SGC integrado también deberá contemplar los requisitos específicos de gestión de servicios de TI y de gestión de seguridad de la información. Estos requisitos son los que no han guardado relación con ninguna categoría de requisitos de la norma ISO 9001, es decir, el tercer tipo de relación descrito en el apartado 3.1.

5. Conclusiones

En este artículo se ha ofrecido una visión de la situación actual de los estándares de sistemas de gestión más comúnmente demandados por las organizaciones del sector TI con la intención, tanto de presentar los modelos existentes, como de identificar sus elementos comunes, para crear un nuevo sistema de gestión integrado. El sistema de gestión integrado presentado toma como base los requisitos del SGC propuestos por la norma ISO 9001 y los amplía con todos los requisitos específicos de gestión de servicios de TI y de gestión de

seguridad de la información que describen las normas ISO/IEC 20000-1 e ISO/IEC 27001 respectivamente.

ISO/IEC 9001:2008	ISO/IEC 20000-1:2005	ISO/IEC 27001:2005
0.2 Enfoque basado en procesos	**4 Planificación e implementación de la gestión del servicio**	0.2 Enfoque por proceso
1 Objeto y campo de aplicación	**1 Objeto y campo de aplicación**	
4 Sistema de gestión de la calidad	**3 Requisitos de un sistema de gestión**	**4 Sistema de gestión de seguridad de la información**
4.1 Requisitos generales		4.1 Requisitos generales
4.2 Requisitos de la documentación		4.3 Requisitos de la documentación
4.2.1 Generalidades	3.2 Requisitos de la documentación	4.3.1 Generalidades
4.2.2 Manual de la calidad		4.3.1 Generalidades
4.2.3 Control de los documentos	3.2 Requisitos de la documentación	4.3.2 Control de documentos
4.2.4 Control de los registros	3.2 Requisitos de la documentación	4.3.3 Control de registros
5 Responsabilidad de la dirección		**5 Responsabilidad de la dirección**
5.1 Compromiso de la dirección	3.1 Responsabilidad de la dirección	5.1 Compromiso de la dirección
5.2 Enfoque al cliente	3.1 Responsabilidad de la dirección	
5.4 Planificación	4.1 Planificación de la gestión del servicio (Planificar)	
5.5.2 Representante de la dirección	3.1 Responsabilidad de la dirección	
5.6 Revisión por la dirección		**7 Revisión del SGSI por la dirección**
5.6.1 Generalidades		7.1 Generalidades
5.6.2 Información de entrada para la revisión		7.2 Datos iniciales de la revisión
5.6.3 Resultados de la revisión		7.3 Resultados de la revisión
6 Gestión de los recursos		5.2 Gestión de los recursos
6.1 Provisión de recursos		5.2.1 Provisión de los recursos
6.2.2 Competencia, formación y toma de conciencia	3.3 Competencia, concienciación y formación	5.2.2 Concienciación, formación y capacitación
7.3 Diseño y desarrollo	5 Planificación e implementación de nuevos servicios o de servicios modificados	
7.5 Producción y prestación del servicio	4.2 Implementación de la gestión del servicio y provisión de los servicios (Hacer)	
8.2.2 Auditoría interna	4.3 Monitorización, medición y revisión (Verificar)	**6 Auditorías internas del SGSI**
8.2.3 Seguimiento y medición de los procesos	4.3 Monitorización, medición y revisión (Verificar)	
8.5 Mejora	4.4 Mejora continua (Actuar)	**8 Mejora del SGSI**
8.5.1 Mejora continua	4.4 Mejora continua (Actuar)	8.1 Mejora continua

ISO/IEC 9001:2008	ISO/IEC 20000-1:2005	ISO/IEC 27001:2005
8.5.2 Acción correctiva		8.2 Acción correctiva
8.5.3 Acción preventiva		8.3 Acción preventiva

Tabla 1. Sistema de gestión integrado

Durante esta investigación hemos podido observar que, debido al gran número de elementos comunes entre los tres sistemas de gestión, el esfuerzo necesario para implantar el sistema de gestión integrado propuesto, sería mucho menor que el esfuerzo que supondría implantar las tres normas de manera independiente.

Siguiendo la línea de investigación iniciada en este trabajo, los autores esperan como trabajo futuro ampliar el enfoque del sistema de gestión de la calidad proporcionado por la norma ISO 9001 con el modelo propuesto por la reciente norma ISO 9004:2009 *Managing for the sustained success of an organization - A quality management approach* [15]. Esta norma proporciona una orientación para la mejora sistemática y continua del desempeño global de la organización, promoviendo la revisión de su SGC utilizando una autoevaluación según un modelo de madurez por niveles.

Agradecimientos

Esta investigación ha sido posible gracias al soporte ofrecido por los proyectos coordinados TIN2007-67843-TIN2007-67843-C06-04 *"Modelos de simulación basados en ontologías y mejora de procesos para arquitecturas orientadas a servicios"*, SOAQSim y TIN2010-20057-TIN2010-20057-C03-03: *"Simulación aplicada a la gestión de equipos, procesos y servicios*", Sim4Gest.

Referencias

[1] *PAS 99:2006 Specification of common management system requirements as a framework for integration*, British Standards Institution, 2006.

[2] *ISO Guide 72:2001 Guidelines for the justification and development of management system standards*, ISO, 2001.

[3] *The integrated use of management system standards*, ISO, 2008.

[4] *UNE 66177 Sistemas de gestión - Guía para la integración de los sistemas de gestión*, AENOR, 2005.

[5] Amengual, E. y Mas, A., "A New Method of ISO/IEC 15504 and ISO 9001:2000 Simultaneous Application on Software SMEs". *Proceedings of the Joint ESA - 3rd*

International SPICE Conference on Process Assessment and Improvement. Marzo de 2003, pp. 87-92, 2003.

[6] Mas, A. y Amengual, E., "A Method for the Implementation of a Quality Management System in Software SMEs", En: British Computer Society, *Proceedings of the Twelfth International Conference on Software Quality Management. Marzo de 2004*, pp. 61-74, 2004.

[7] Mas, A., Mesquida, A.L., Amengual, E. y Fluxà, B., "Best practices to facilitate ISO/IEC 27000 implementation". *Proceedings of the 5th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2010). Julio de 2010*, pp. 192-198.

[8] *ISO/IEC TR 15504-7:2008 Information technology - Process assessment - Part 7: Assessment of organizational maturity*, ISO/IEC, 2008.

[9] *ISO 9001:2008 Quality management systems - Requirements*, ISO, 2008.

[10] *ISO/IEC 20000-1:2005 Information technology - Service management - Part 1: Specification*, ISO/IEC, 2005.

[11] *ISO/IEC 27001:2005 Information technology - Security techniques - Information security management systems - Requirements*, ISO/IEC, 2005.

[12] *ISO/IEC 90003:2004 Software engineering - Guidelines for the application of ISO 9001:2000 to computer software*, ISO/IEC, 2004.

[13] *ISO/IEC TR 90005:2008 Systems engineering - Guidelines for the application of ISO 9001 to system life cycle processes*, ISO/IEC, 2008.

[14] *ISO/IEC NP 90006 Information technology - Guidelines for the application of ISO 9001:2000 to IT service management*, ISO/IEC.

[15] *ISO 9004:2009 Managing for the sustained success of an organization - A quality management approach,* ISO/IEC, 2009.

Implantación de CMMi nivel de madurez 2 en una PYME

Fernando Ramos
Icosís Grupo Avalon. Sevilla.
fernando.ramos@grupoavalon.es

Olimpia Torres[(1)], Nicolás Sánchez[(2)], Manuel Alba[(2)]
[(1)]Icosís Grupo Avalon. Sevilla, [(2)]Grupo IWT2. Fundación Fidetia. Sevilla
olimpia.torres@grupoavalon.es, {nicolas.sanchez, manuel.alba}@iwt2.org

Resumen

La situación económica actual requiere que las pequeñas y medianas empresas (PYMES) crezcan ofreciendo servicios de mayor calidad que les permitan aumentar su competitividad. En el entorno de empresas orientadas al desarrollo del software, optar por conseguir algunas de las certificaciones internacionales de calidad puede ser un aspecto muy relevante y bien valorado; ahora bien, el proceso que una pyme debe llevar a cabo para obtener una de estas certificaciones puede resultar complejo y costoso. Este artículo presenta un caso de éxito en el que se muestra como la empresa ICOSIS, una pyme del grupo AVALON ha conseguido la acreditación en el nivel de madurez 2 del modelo CMMi. El siguiente texto muestra los antecedentes, el proceso de implantación y los resultados.

Palabras clave: calidad, CMMi, pyme, competitividad.

CMMi maturity level 2 implantation in a SME

Abstract

The current economic situation requires that a growth of small and medium enterprises (SME) by offering higher quality services which allow them to increase their competitiveness. Within business environment oriented to software development, choosing for achieve some of those international quality certifications might be a relevant and quite appreciated aspect; however, the process that a SME has to go through in order to achieve one of these certifications could be complex and expensive. This article introduces a success case that shows how the company ICOSIS, a SME of AVALON Group, has achieved the credentials on CMMi Maturity level 2. The following text shows precedents, implementation process and the results obtained

Key words: quality, CMMi, sme, competitiveness.

Ramos, F., Torres,O., Sánchez, N. y Alba, M., "Implantación de CMMi nivel de madurez 2 en una PYME", REICIS, vol. 6, no.3, 2010, pp.35-46. Recibido: 8-11-2010; revisado: 14-11-2010; aceptado: 23-11-2010

1. Antecedentes

1.1. Breve Caracterización de Icosís Grupo Avalon

Icosís Grupo Avalon (www.icosis.es) es una empresa fundada en 1992. Su actividad industrial se basa en el desarrollo de de software a medida, principalmente sistemas Web, servicios de oficinas de gestión de proyectos y calidad, proyectos de investigación y desarrollo, y el servicio de profesionales informáticos bajo demanda. Su facturación anual ronda los 2 millones de euros y cuenta con una plantilla media de 65 profesionales. En 2010 Icosís se ha integrado en el grupo de empresa de Avalon Tecnologías de la Información (www.grupoavalon.es), ampliando de esta forma su ámbito geográfico de actuación de Andalucía a nivel nacional e internacional.

Los principales clientes de Icosís Grupo Avalon se pueden resumir en tres grandes grupos:

- Administración pública.
- Empresas TIC privadas.
- Empresas privadas no TIC

Respecto a los usuarios finales de los sistemas que desarrolla, estos van desde usuarios cuyo trabajo tan sólo precisa unos conocimientos mínimos de ofimática, hasta usuarios expertos tanto en tecnologías como en el uso de herramientas software.

Respecto a las iniciativas de calidad o mejora, Icosís Grupo Avalon cuenta con las siguientes certificaciones:

- UNE-EN-ISO 9001-2008 [1], Sistema de Gestión de Calidad implantado en 2003.
- UNE 166002 [2], Sistema de Gestión de I+D+i implantado en 2005.
- UNE-EN-ISO 14001-2004 [3], Sistema de Gestión Ambiental implantado en 2005

1.2. Elección de CMMi (Capability Maturity Model Integration)

En la mayoría de las empresas TIC, sobre todo las PYMES, el proceso de desarrollo y mantenimiento del software está basado en la improvisación, es decir, las personas tienen un conocimiento y unas habilidades pero no siguen un proceso formalizado.

Como consecuencia de este hecho se producen una serie de problemas de coste y planificación, de imposibilidad de predecir la calidad del producto final y de dependencia

excesiva de las personas (según el informe CHAOS 2009 [4] sólo el 32% de los proyectos de software desarrollados terminaron con éxito). Por tanto, se plantea la necesidad de utilizar procesos software eficaces para que los equipos puedan utilizar de forma rutinaria las mejores prácticas y técnicas de gestión, con el objeto de que los desarrollos acaben con éxito y en los plazos previstos. A esto se une el hecho de que en este sector es difícil establecer, cerrar y controlar los cambios en requisitos del cliente, lo que alarga y complica enormemente los desarrollos.

Después de estudiar los diferentes modelos existentes para seguir el proceso de mejora de manera que estuvieran recopiladas las buenas prácticas en la producción de software y pudiesen cuantificarse las mejoras alcanzadas, la dirección de Icosís Grupo Avalon optó por CMMi [5] por varias razones:

- Modelo complementario y compatible con otras normas en las que se basaba hasta el momento el Sistema de Gestión Integrada establecido en Icosís.
- Modelo construido en base a una larga historia de implementación y avalado por buenos resultados a nivel mundial.
- Modelo que utiliza niveles de madurez para medir la mejora estableciendo áreas de proceso asegurando la mejora continua. El modelo indica siempre qué hacer pero no cómo hacerlo.
- Posibilidad de participación en el proyecto PYMETICA CMMi (Proyecto de Mejora de los Procesos de Desarrollo y Mantenimiento de Sistemas y Productos de Software - http://cmmieticom.com), organizado por Eticom (Asociación de Empresarios de Tecnologías de la Información y Comunicación de Andalucía http://www.eticom.com), con una subvención importante tanto en consultoría externa como en el proceso de acreditación (SCAMPI A).

2. Implantación

2.1 Proceso de Implantación

El proceso de implantación del modelo CMMi ha pasado por varias fases hasta la culminación de la acreditación formal en Junio de 2009, fecha en la que tuvo lugar el SCAMPI A [6] por parte del SEI (Software Engineering Institute) [7]. La figura 1 muestra las fases de dicho proceso de implantación.

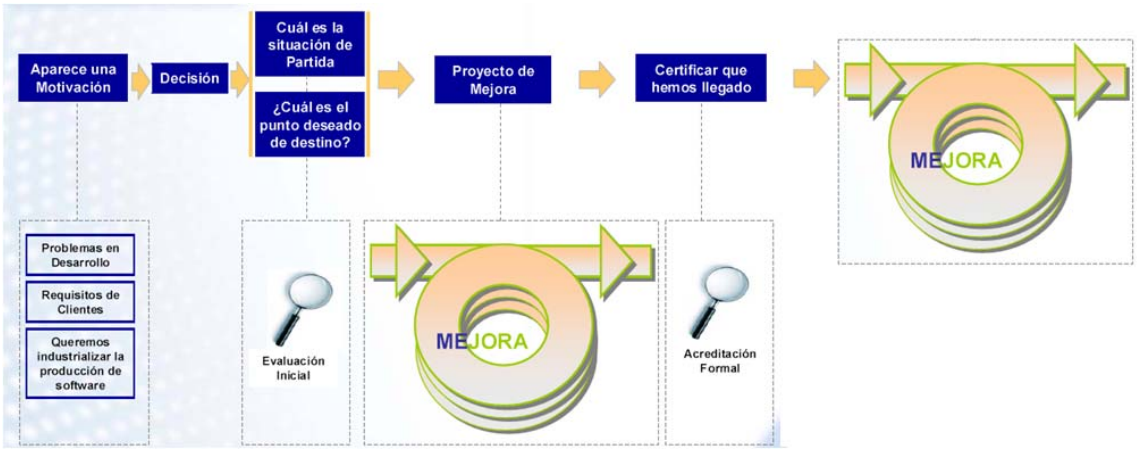


Figura 1. Proceso de Implantación del Modelo CMMi.

El paso inicial fue la necesidad de estandarizar la forma de trabajo de manera que tuviera un reflejo en los costes del proyecto y en su plazo de ejecución, sorteando las dificultades de la definición de requisitos por parte del cliente, del control de los posibles cambios de los mismos y de que el conocimiento no fuese de las personas sino de la organización.

El siguiente paso fue tomar la decisión por parte de la Dirección de Icosís Grupo Avalon de involucrarse en un proceso de mejora; realizándose en este momento un estudio de los posibles modelos a seguir. Paralelamente surgió el proyecto PYMETICA CMMi de Eticom, que ofrecía una subvención del Ministerio de Industria para alcanzar la acreditación en el nivel de madurez 2 del modelo. Dentro de esta subvención, se establecía la ayuda de una consultora externa especializada en la implantación de CMMi.

Icosís Grupo Avalon, designó un grupo de tres personas con conocimiento suficiente del funcionamiento de la empresa para que, junto con los consultores externos, llevasen a cabo una evaluación inicial identificando las áreas de proceso en las que se llevaría a cabo la mejora, concretamente:

- Gestión de proyectos
- Gestión de requisitos
- Gestión de la configuración
- Aseguramiento de la calidad
- Medición y análisis.

De igual forma se fueron identificando las debilidades para cada una de estas áreas y se pusieron las bases para el desarrollo de un activo de procesos, conjunto de plantillas y documentos de descripción de dichos procesos, en las dichas áreas.

A partir de un primer esbozo de estos procedimientos, el equipo de mejora consideró necesario involucrar a más personal de la empresa, principalmente aquellos perfiles relacionados con la gestión de proyectos, como son los Gerentes de Proyecto. Con estas incorporaciones se definieron formas de trabajo, responsabilidades de cada uno de los perfiles participantes en los distintos proyectos de desarrollo y posibles herramientas a incorporar a fin de adaptar la nueva forma de trabajo.

El siguiente paso consistió en integrar todos los procedimientos que se estaban definiendo con otras normas en las que Icosís Grupo Avalon ya estaba certificado y en las que se basaba hasta el momento el Sistema de Gestión establecido en la organización (fundamentalmente las normas ISO 9001 e ISO 166002) en todo lo referente a planificación, gestión, pruebas y aseguramiento de la calidad.

Una vez definida y documentada la nueva forma de trabajo, la consultora externa propuesta por el programa PYMETICA CMMi de Eticom hizo una revisión de dicha definición para cada área de proceso, así como de las herramientas y de las plantillas que servían como registro o evidencia de los distintos procesos. Esta evaluación, denominada GO/NO GO, suponía la recta final hacia la acreditación y comprobaba si la definición cumplía con los requisitos exigidos por el nivel de madurez 2 de CMMi. En Icosís Grupo Avalon el reto se superó de forma satisfactoria y se procedió a la preparación de las siguientes fases.

Con el activo de procesos definido, documentado y evaluado positivamente se procedió a la fase de formación de personal; para ello se organizaron, para todo el personal de producción, sesiones formativas y seminarios para cada una de las áreas de proceso, completándose con formación individualizada según perfiles de trabajo en aras a asegurar que todos los profesionales involucrados fueran conscientes tanto de sus tareas como de sus responsabilidades a la hora de acometer un proyecto de desarrollo.

Una vez que los profesionales estuvieron formados y comenzaron a desarrollar su labor en base al nuevo modelo de trabajo definido, el siguiente paso fue superar el SCAMPI A. En el proceso, llevado a cabo por un Lead Appraiser acreditado por el SEI, se eligió una

muestra representativa de todo el personal involucrado en la producción de software de la empresa, fueron entrevistados y tuvieron que demostrar, con evidencias, cuál era su forma de trabajo, cómo asumían las responsabilidades y las funciones que tenían asignadas según su perfil.

El resultado de 3 días de entrevistas fue un informe favorable y la emisión por parte del SEI de la acreditación de Icosís Grupo Avalon como empresa con CMMi nivel de madurez 2 (http://sas.sei.cmu.edu/pars/pars_detail.aspx?a=13013).

Aunque el proceso de acreditación del nivel de madurez 2 de CMMi terminó con el SCAMPI A; a la hora de poner en práctica lo definido en el activo de procesos surgen muchas posibilidades de mejora que deben canalizarse, analizarse y llevarse a la práctica. Es por esto que después de conseguir la acreditación se constituyó un Comité de Mejora integrado tanto por Gerentes de Proyecto como por el personal del Departamento de Calidad de Icosís Grupo Avalon. El Comité se reúne semanalmente, analiza las acciones de mejora registradas y en caso de ser viables se modifica el activo de procesos. Gracias a este modelo se asegura la mejora continua y se apuesta por subir al nivel de madurez 3 de CMMi, de hecho ya se realizan algunas de las prácticas exigidas por dicho nivel.

2.1 Esfuerzo

El proceso de implantación del modelo CMMi comenzó en Enero de 2008 y finalizó en Junio de 2009 con la acreditación del nivel de madurez 2 tras superar el SCAMPI A. El esfuerzo realizado durante estos 18 meses se resume en la siguiente tabla:

Tareas Alto Nivel	Coordinador	Equipo de Implantación	Equipo de Desarrollo
Formación recibida	43	91	
Adaptación de activo de procesos	220	720	
Adaptación de herramientas		463	
Formación al personal de desarrollo			268
Formación impartida	6	171	
Coordinación y seguimiento	95	180	
Total	364	1.625	268

Tabla 1. Esfuerzo en horas para la implantación del nivel de madurez 2 de CMMi en Icosís Grupo Avalon.

Tras la acreditación se ha constituido un Comité de Mejora cuyo esfuerzo durante 2009 se resume en la siguiente tabla.

Tareas Alto Nivel	Coordinador	Comité de Mejora
Comité de mejora	60	320
Revisión activo de proceso y adaptación de herramientas		720
Total	60	1.040

Tabla 2. Esfuerzo en horas del comité de mejora del activo de procesos del nivel de madurez 2 de CMMi en Icosís Grupo Avalon

3. Resultados

3.1 Evaluación de resultados

Después de un largo proceso de implantación, formación y aprendizaje por parte de todo el personal de Icosís Grupo Avalon implicado en el desarrollo de software, podemos afirmar que la implantación del modelo CMMi ha sido positiva en líneas generales, si bien es cierto que no siempre la nueva forma de trabajo es aceptada por todos y hay quien cree que aumenta en exceso el coste y la burocracia de un proyecto. Desde el Departamento de Calidad se intenta demostrar gracias a las métricas de los proyectos que se van desarrollando que, a pesar de dedicar tiempo a la gestión de las distintas áreas de proceso, la rentabilidad de los proyectos aumenta puesto que se gestionan de forma adecuada los recursos en plazo y tiempo, se aumenta enormemente el control del producto desarrollado, tanto en configuración como en pruebas que se pasan al software, y se asegura la mejora continua del proceso.

Concretamente, las ventajas aportadas en las distintas áreas de proceso son:

- Gestión de Proyectos: el hecho de planificar cada fase del proyecto teniendo en cuenta los recursos destinados para ello y combinándolo con los entregables exigidos por el cliente, hace que el control de los plazos de entrega sea exhaustivo y que además haya control sobre las posibles desviaciones en plazo y coste de cada fase. Es decir, cuando un proyecto se desvíe de lo planificado se sabrá perfectamente en que fase se produce la desviación con lo que podrá atajarse el problema más rápidamente. Además, el hecho de hacer una buena gestión de los posibles riesgos asociados al proyecto con el establecimiento de acciones preventivas y/o correctivas, supone anticiparse a posibles problemas y buscar soluciones.

- Gestión de requisitos: esta área de proceso es una de los más beneficiadas con la implantación del modelo CMMi, puesto que siempre ha sido difícil acordar los requisitos con el cliente y sobre todo, controlar los posibles cambios de los mismos por parte del cliente una vez definidos inicialmente. Con la implantación de esta área de proceso se ha conseguido que, previamente al comienzo de los trabajos de desarrollo, los requisitos del cliente estén prácticamente acordados mediante la creación de una línea base. El cambio en los mismos se controlan y gestionan con cambios en dicha línea base, identificando y controlando todos aquellos ECS (Elementos de Configuración Software) a los que afecte dicho cambio. Cabe destacar también la ventaja que supone el hecho de definir las pruebas unitarias y las pruebas de sistemas relacionadas con los requisitos mediante matrices de trazabilidad (para lo que se usa un profile de la herramienta Enterprise Architect basado en la metodología NDT [8]) así como registrar los resultados de dichas pruebas, lo que nos permite asegurar que el software ha sido testado.
- Gestión de la Configuración: esta área de proceso era el mayor punto débil en Icosís Grupo Avalon y también el que mayor esfuerzo sigue necesitando después de la implantación de CMMi. El avance de incluir la figura de Ingeniero de Configuración en cada proyecto como máximo controlador de los ECS fundamentales del mismo, supone el control exhaustivo desde el registro de entornos al código fuente propiamente dicho. A esto se une el hecho de que cualquier desarrollo debe pasar al menos una auditoría de configuración en la que se asegura el control que debe llevar el Ingeniero de Configuración.
- Aseguramiento de la Calidad: el hecho de concienciar a todo el personal de desarrollo de la importancia del registro de No Conformidades y el análisis y seguimiento de las mismas, ha sido un paso importante de cara a solucionar una serie de problemas que se repetían sistemáticamente en los proyectos, al mismo tiempo que se asegura la mejora continua. Por otra parte, las revisiones a las que el Departamento de Calidad somete, tanto a los proyectos en el momento de terminar determinadas fases, como a los entregables, supone detectar una serie de

problemas que no se transmiten ni a otras fases del ciclo de vida del proyecto ni por supuesto al cliente.

- Medición y análisis: esta área de proceso es la que más trabajo está costando implantar, ya que a veces y aunque las métricas fundamentales en un proyecto estén claras, es difícil conseguir datos porque no siempre se cuenta con las herramientas apropiadas o incluso hay veces en las que es complejo y costoso en tiempo poder llegar a obtener resultados fiables. Por tanto, sigue siendo una tarea pendiente para mejorar el conseguir mejores métricas que sirvan para tener un histórico en el que basar estimaciones posteriores y con las que poder demostrar la rentabilidad del uso de esta metodología de trabajo.

3.1 Algunas medidas

A continuación presentamos algunas de las métricas contempladas desde 2008:

Métricas	2008	2009	2010 (hasta 2º semestre)
Proyectos que requieran esfuerzos en garantía menor que el 2,5% del esfuerzo total.(1)	0,51	0,93	0,85
Proyectos con desviaciones en coste mayor del 10%.(2)	0,53	0,67	0,73
Reclamaciones.(3)	0,05	0,02	0,00
No conformidades.(4)	0,47	0,68	0,20

Tabla 1. Ejemplo de métricas analizadas en Icosís Grupo Avalon desde 2008.

(1) Debido al avance realizado en la gestión de la configuración y al aumento y la definición de las pruebas y a las auditorias de Calidad y de Configuración a las que se somete el proyecto se está consiguiendo que el número de proyectos que requieran esfuerzos superiores al 2.5% del total sea cada vez menor.

(2) Más del 20% de los profesionales que componen el área de desarrollo de Icosís Grupo Avalon están acreditados por Aidit [9] como personal investigador en base al Real Decreto 278/2007 [10], lo que significa que llevamos a cabo muchos proyectos de investigación y desarrollo utilizando nuevas tecnologías. Esta característica implica un aumento en las desviaciones debido al desconocimiento implícito en este tipo de trabajos, pero a su vez está contribuyendo a crear una base de datos muy amplia que nos servirá a

medio y largo plazo como fuente de información para estimaciones de proyectos relacionados con estas nuevas tecnologías.

(3) Hasta 2009, las reclamaciones eran transmitidas por los clientes a través de los Gerentes de Proyecto. A partir de 2009 se puso en marcha un sistema de información que proporciona una relación con el cliente que le permite transmitir sus reclamaciones directamente de una manera más ágil y sencilla. Cabe señalar que incluso habiendo facilitado la vía de comunicación, las reclamaciones han disminuido considerablemente.

(4) Hasta 2009 solo se registraban no conformidades de la capa media-alta de dirección de Icosís Grupo Avalon (Gerentes de Proyecto y Directores). A partir de 2009 se ha extendido el registro de no conformidades a todos los niveles de Icosís Grupo Avalon y aún así, en 2010 está descendiendo su número.

4. Conclusiones

Existe un amplio conjunto de estándares, normas y metodologías destinados a todo tipo de organizaciones y que nos ayudan a la mejora de la calidad de la empresa, la gestión del control, la gestión de los servicios, la mejora en el proceso de desarrollo del software, etc. De todos ellos Icosís Grupo Avalon se ha decantado por el Modelo CMMi, fundamentalmente porque entendemos que la calidad de un producto viene determinada en gran medida por la calidad del proceso que se utiliza para desarrollarlo y mantenerlo.

Tras el intenso trabajo de estos dos últimos años, estamos en disposición de afirmar que la implantación del nivel de madurez 2 de CMMi, pese al ser el primer nivel, es muchas veces el más difícil de lograr. La implantación de un Modelo de estas características es un proceso largo y costoso que puede significar varios años de esfuerzo. Este proceso requiere que se cambie la forma de trabajar de la organización, lo que la mayoría de las veces implica un cambio cultural en la misma. Prueba de ello es la alta inversión en recursos humanos y técnicos que hemos realizado y la constante labor de concienciación que se ha llevado y se sigue llevando a cabo en Icosís Grupo Avalon. También es necesario reseñar que el apoyo de la dirección (sin él no se tendría suficiente autoridad en los momentos difíciles) y los consejos de los consultores expertos en esta materia se han tornado imprescindibles para la consecución de este nivel.

Sin embargo, y pese al esfuerzo, las ventajas son evidentes, y es que el proceso de desarrollo de software en Icosís Grupo Avalon ha pasado del individualismo de los desarrolladores al conjunto de la organización, repartiéndose de esta forma el conocimiento y reduciéndose el riesgo asociado a dicha personalización, aumentando además la productividad y los beneficios en los proyectos. La implantación de este Modelo está aportando a nuestra organización un gran margen de mejora; está provocando la reducción de reclamaciones y el gasto de recursos durante la garantía de los proyectos, y además, nos está ayudando a industrializar el proceso de construcción de software. La consecuencia directa de estos cambios es que estamos haciendo más competitiva la empresa y estamos ganando en prestigio y reconocimiento externo, llegando incluso a ser reconocido como un factor importante en la selección de proveedores de software por parte de nuestros clientes.

Agradecimientos

Gracias al grupo de Ingeniería Web y Testing Temprano (www.iwt2.org) del Departamento de Lenguajes y Sistemas de la Universidad de Sevilla por estar siempre disponibles y prestarnos, sin condiciones, tanto sus servicios como la metodología NDT, junto con las herramientas asociadas a la misma.

Referencias

[1] AENOR, *UNE-EN ISO 9001:2008, Sistemas de gestión de la calidad. Requisitos. (ISO 9001:2008)*

[2] AENOR, *UNE 166002:2006, Gestión de la I+D+i: Requisitos del Sistema de Gestión de la I+D+i.*

[3] AENOR, *UNE-EN ISO 14001:2004, Sistemas de gestión ambiental. Requisitos con orientación para su uso. (ISO 14001:2004)*

[4] Informe CHAOS 2009. http://www1.standishgroup.com/newsroom/chaos_2009.php, 7 de noviembre de 2010.

[5] Beth Chrissis, M., Konrad, M., Shrum,S., *CMMi FOR DEVELOPMENT, VERSIÓN 1.2. CMMi Second Edition, Guidelines for Process Integration and Product Improvement*. Addison-Wesley, 2007.

[6] SCAMPI Upgrade Team, *Standard CMMI Appraisal Method for Process Improvement (SCAMPI) A, Version 1.2: Method Definition Document,* Handbook CMU/SEI-2006-HB-002 August 2006.

[7] SEI. *Software Engineering Institute (Instituto de Ingeniería del Software).* http://www.sei.cmu.edu, 18 de noviembre de 2010.

[8] NDT. *Navigational Development Technique.* http://www.iwt2.org/ndt.php, 8 de noviembre de 2010.

[9] Aidit, Agencia de Acreditación en Investigaciones, Desarrollo e Innovación Tecnológica. http://www.e-aidit.com, 18 de noviembre de 2010.

[10] REAL DECRETO 278/2007, de 23 de febrero, sobre bonificaciones en la cotización a la Seguridad Social respecto del personal investigador. Ministerio de la Presidencia, Gobierno de España. http://www.boe.es/boe/dias/2007/02/24/pdfs/A08037-08040.pdf, 18 de noviembre de 2010.

Pruebas de aceptación en sistemas navegables

J. Ponce, F.J. Domínguez-Mayo, M.J. Escalona, M. Mejías,
D. Pérez, G. Aragón, I. Ramos
Departamento de Lenguajes y Sistemas Informáticos, Universidad de Sevilla
josepg@us.es

Resumen

En los últimos años la importancia de la fase de pruebas ha ido creciendo hasta el punto de que se establece su necesidad como una clave para el aseguramiento de la calidad de los productos software. A pesar de este interés y necesidad, sin embargo, la fase de pruebas es compleja y aún hoy no existen protocolos generales o normas ampliamente aceptadas que certifiquen qué se entiende por una buena fase de pruebas o no. Si a esto lo acompañamos de los nuevos requerimientos que las nuevas tecnologías y desarrollos incluyen, como por ejemplo los sistemas navegables, nos encontramos con que existe poco soporte para los equipos de desarrollo a la hora de ofertar una estrategia de pruebas efectiva. Este trabajo trata de presentar un análisis del estado del arte de un conjunto de pruebas muy específicas: las pruebas de aceptación. Y, concretamente, se centra en estudiar qué ofertas hay para el tratamiento de este tipo de pruebas en sistemas navegables. El artículo pretende ser una visión general que permite establecer puntos de trabajos iniciales para la definición de nuevas estrategias de pruebas en este campo.

Palabras clave: D.2 Software Engineering, D.2.5.k Testing strategies, D.2.5.l Test design, D.2.19 SoftwareQuality/SQA

Acceptance Testing for Navigation Systems

Abstract

In recent years the importance of the testing phase has become a key factor to ensuring the quality of software products. However, despite this interest and need, the testing phase is a very complex phase and, even today, there are no general protocols and widely accepted standards to certify what is meant by a good phase of testing or not. If new requirements that include new technologies and developments, such as navigation systems, must be also considered we find that there is little support for development teams when they looks for an effective test strategy. This paper aims to present an analysis of the state of the art of a very specific set of tests: acceptance tests. And, specifically, it is focused on studying recent solutions to deal with acceptance tests in navigable systems. The article is a general view of the state of the art and the base to the definition of new test strategies in this field.

Key words: D.2 Software Engineering, D.2.5.k Testing strategies, D.2.5.l Test design, D.2.19 SoftwareQuality/SQA

Ponce, J. Domínguez-Mayo,F.J., Escalona, M.J., Mejías, M., Pérez, D., Aragón, G. y Ramos, i., "Pruebas de aceptación en sistemas navegables", REICIS, vol. 6, no.3, 2010, pp.47-55. Recibido: 8-11-2010; revisado: 14-11-2010; aceptado: 22-11-2010

1. Introducción

Dentro de las fases que conforman el ciclo de vida, la fase de pruebas es fundamental a la hora de asegurar la calidad del producto entregado orientada, entre otras muchas cosas, a la validación y verificación del sistema. Por verificación se entiende que se esté construyendo el producto correctamente y por validación que se esté construyendo el producto correcto. En dicha fase se intenta que el software sea entregado con la mayor calidad posible, entendiendo como producto de calidad aquel que cumpla con las características marcadas por algunos estándares entre los que destaca ISO/IEC 9126 [1] siendo éste un estándar internacional para la evaluación de la calidad del producto software. Por otro lado, SQuaRE (Software Product Quality Requirements and Evaluation) es una revisión de ISO/IEC 9126 y tiene las mismas características de calidad del software pero extendidas con nuevas subcaracterísticas para las características ya definidas en ISO 9126. El objetivo general de la creación del estándar ISO/IEC 25000 SQuaRE es organizar, enriquecer y unificar las series que cubren dos procesos principales: especificación de requerimientos de calidad del software y evaluación de la calidad del software, soportada por el proceso de medición de calidad del software. La serie ISO/IEC 25000:2005 reemplaza a dos estándares relacionados: ISO/IEC 9126 e ISO/IEC 14598. Por tanto, el objetivo de la fase de pruebas es que el producto final acabe cumpliendo las expectativas creadas por el mismo, ciñéndose de esta manera al estándar establecido.

En cuanto a estándares de pruebas se refiere, existen algunos como BS 7925 o IEEE Std 1008 [2] los cuales están primordialmente enfocados a pruebas unitarias que son llevadas a cabo en fases tempranas dentro de la fase pruebas. Sin embargo, existen áreas no cubiertas por los estándares anteriores como son aspectos organizativos, proceso y gestión de las pruebas, técnicas funcionales y no funcionales, etc. El objetivo del nuevo estándar ISO/IEC 29119 [3] es, por tanto, el de unificar los anteriores estándares en uno solo para intentar cubrir la mayoría de áreas no cubiertas por los mismos hasta el momento.

La idea general que se tiene para validar aplicaciones o sistemas informáticos son realizar un conjunto de pruebas de diferente índole y tipo, desarrolladas por un conjunto de personas pertenecientes a unas y otras entidades, ejecutadas por personas que de nuevo pueden pertenecer o no, a esas mismas o diferentes entidades, que por uno u otro lado componen toda esa maraña que rodea a el complejo proceso de creación de software, donde

una de las fases más importantes del ciclo de vida por donde va pasando dicho software, es el estudio, diseño, ejecución y validación de estas pruebas que van a dar una visión o valoración, muy importantes al software que finalmente se está construyendo. Dentro de este trabajo, tras una introducción a la fase de pruebas, nos centraremos primeramente dentro de toda la tipología de pruebas que tenemos en las de aceptación del sistema, pruebas que se encargan de realizar una muestra de individuos pertenecientes al colectivo de usuarios finales, bajo la supervisión de miembros de otros grupos de trabajo pertenecientes al proyecto. Abordaremos dichas pruebas, para finalmente centrarlas, en la aplicación de pruebas para validar sistemas navegables, poniendo como ejemplo de este tipo de sistemas cualquier sistema Web, haciendo un estudio de las características más importantes a cubrir intentando reflejar su repercusión en la mejora de calidad.

El hecho de centrarnos en las pruebas de aceptación, surge de intentar reforzar la visión de que el usuario integrado en dicha fase, de forma temprana, ayudaría a mejorar dicho proceso desde la fase de planificación y diseño de las pruebas, con las consiguientes mejoras de muchos aspectos cuantificables y deseables como pueden ser:

- Aumento en la calidad del software integrado
- Minimización de costes
- Aumento de la fiabilidad en los resultados del proyecto
- Se produce un incremento de la satisfacción del cliente al utilizar un software con una cantidad de errores inferior.
- Se incrementa la eficiencia del proceso de desarrollo.

En el apartado 2, se presenta de forma general la situación actual en las Pruebas de Aceptación así como las diferentes modalidades de pruebas. Seguidamente, en el apartado 3 se describen las diferentes Pruebas de Aceptación pero orientadas a Sistemas Navegables. Finalmente, en el apartado 4 se sintetizan una serie de conclusiones a tener en cuenta en base a las diferentes características deseables en sistemas Web.

2. Estudio de la situación actual en Pruebas de Aceptación

Dentro del tipo de pruebas que tenemos para validar el software, una de las más importantes son las de aceptación (user's tests). Son aquellas pruebas que son diseñadas por el propio equipo de desarrollo en base a los requisitos funcionales especificados en la fase

de análisis para cubrir todo ese espectro, y ejecutadas por el propio usuario final, no por todos evidentemente, pero sí por una cantidad de usuarios finales significativo que den validez y conformidad al producto que se les está entregado en base a lo que se acordó inicialmente.

Dependiendo de la complejidad del sistema a probar, si está o no dividido por módulos, etc. la realización de dichas pruebas se ejecuta de forma diferente. Si estuviera una aplicación dividida en módulos, se tratarían esto como subsistemas y estudiando si tienen o no la suficiente complejidad como para tratarlos de forma diferente, habría que realizar sesiones de prueba de aceptación diferentes. Las pruebas de aceptación, tienen dos tipos de procedimiento para realizarlas, las llamadas pruebas alfa y beta.

2.1. Pruebas alfa

Es aquella en la que se le entrega a un usuario final todo el producto final, junto a su documentación correspondiente para que este, en presencia del desarrollador y en entornos previamente preparados para el proceso de dichas pruebas, vaya informando de todo lo que vea que no está bien, que no se cumple, etc.

2.2. Pruebas beta

Es la que se le proporciona a usuarios finales, situados en lugares concretos de los puestos de trabajo donde finalmente será el software implantado, para que sea de nuevo el usuario y sin estar nadie presente del resto de grupos de trabajo, el que de nuevo vuelva a emitir unos informes de resultados e impresiones de la aplicación o sistema software desarrollado.

Independiente del proceso seguido, también tendríamos que prestar especial atención a la documentación entregada al cliente para su posterior evaluación, ya que la entrega de aplicación y documentación, debería tener la misma calidad.

3. Pruebas de Aceptación en Sistemas Navegables

La prueba de una determinada aplicación Web, ejemplo tomado dentro de los múltiples tipos de sistemas navegables debido en parte por la gran demanda de los mismos que hay, así como el enfoque que se han encauzado en los últimos años hacia la aplicación software basado en la Web, es una colección de actividades para descubrir errores en el contenido, la funcionalidad, utilidad, navegabilidad, rendimiento, seguridad de dicha aplicación. Dentro

de las pruebas de aceptación se intentará cubrir revisiones en una o varias dimensiones de calidad mostradas a continuación (basándonos en [4][5]):

- El contenido, evaluándose tanto a nivel sintáctico hasta el nivel semántico. En el primero se harán revisiones de vocabulario, puntuación, gramática, etc. y en el segundo se valora la corrección de la información presentada, la consistencia, así como la falta de ambigüedad en lo aportado.
- La función se probará para descubrir errores que indiquen la falta de conformidad por parte del cliente. Cada función de la aplicación se valorará en su corrección, inestabilidad así como la conformidad con diferentes estándares de programación.
- La estructura se valorará que garantice adecuadamente el contenido y la función de la aplicación, que es extensible y que puede soportarse conforme se agregue nuevo contenido o funcionalidad.
- La usabilidad, para asegurarse que la aplicación es capaz de soportar a cada tipo de usuario que se pueda conectar a ella pudiendo aprender y aplicar toda la sintaxis y semántica de la navegación requerida.
- La navegabilidad se probará para que toda la estructura de navegación esté correcta y descubrir errores tan simples como usuales de enlaces muertos, inadecuados o erróneos.
- El rendimiento, la compatibilidad, seguridad son otros aspectos a tener en cuenta en dichas revisiones para que la satisfacción del cliente sea máxima y la confianza del producto la más alta posible.

Cuando finalizan las pruebas de integración es el momento de comenzar dichas pruebas. La idea es elaborar un plan de pruebas que agrupe la mayor parte posible de las dimensiones de calidad anteriormente expuestas y que siempre busquen demostrar la conformidad del cliente con respecto a los requerimientos inicialmente definidos basándose en la medida de lo posible, en los estándares de pruebas comentados arriba

Para cada una de las pruebas que mencionamos a continuación, se realizarían técnicas tantos de caja blanca, aquellas que fueran más unitarias, así como de caja negra con vistas a probar de mejor forma la funcionalidad del sistema.

3.1. Pruebas de contenido

En relación a las pruebas de contenido que pueden ser tan triviales como errores tipográficos menores o tan significativos como información incorrecta, organización inadecuada o violación de leyes de la propiedad intelectual, siendo la misión de las pruebas de contenido intentar descubrirlos y muchos más antes que el usuario final los encuentre. Dichas pruebas tienen por objetivo, descubrir errores sintácticos, errores gramaticales, etc. en documentos de texto, representaciones gráficas y otros medios, descubrir errores semánticos, errores en la precisión o completitud de la información y encontrar errores en la organización o estructura del contenido presentado al usuario.

3.2. Pruebas de interfaz de usuario

La verificación y validación de una interfaz de usuario de una aplicación Web ocurre dentro de tres puntos distintos dentro del proceso de software, al nivel de análisis, al de diseño y al de pruebas. Durante la prueba, la atención es centrada en la ejecución de aspectos específicos de la aplicación de la interacción con el usuario, conforme salgan por la sintaxis y la semántica. Una de las funciones será dar validez vínculos, formularios, HTML dinámico, contenido de streaming, etc.

Del mismo modo, el usuario implícitamente irá realizando pruebas para medir la usabilidad del software, donde se indicará el grado en el que el usuario podrá interactuar efectivamente con la aplicación, es decir, en cuánto facilita la vida la aplicación al usuario. Características como la interactividad, legibilidad, estética, accesibilidad son aquellas que son lo más deseable y valorables dentro de las categorizaciones de usabilidad.

3.3. Pruebas de navegación

Este conjunto de pruebas tiene dos vertientes ya que por un lado, puede ser predecible los objetivos del usuario que llega a un determinado punto, y por otro resultar impredecible ya que el usuario puede elegir otro camino al previsto inicialmente, con lo cual, la misión de estas pruebas van a derivar en garantizar que son funcionales todos los mecanismos que permiten al usuario recorrer la aplicación.

Para ello, podrían ser probado muchos de los elementos que existen en la aplicación, como vínculos de navegación, anclas de páginas que nos redireccionan a algún lugar dentro de la misma, redirecciones hacia enlaces caídos, mapas de sitios, motores de búsqueda internos, etc.

3.4. Pruebas de componente

Las pruebas a nivel de componente se hacen al nivel de realizar las llamadas pruebas de función donde generalmente son las respuestas de nuestro sistema ante la entrada de datos en un determinado formulario, donde tras introducirlos, seleccionamos pulsando algún botón el inicio de la ejecución del proceso. Buscan encontrar errores dentro de estas funcionalidades. Aquí son usuales métodos de pruebas de caja negra tal y como particiones de equivalencia, análisis de valores límite, etc.

3.5. Pruebas de configuración

Dentro de la variabilidad de configuraciones posibles dentro de aplicaciones Web, tales como hardware, sistemas operativos, navegadores, capacidad de almacenamiento, velocidades de comunicación de red y otros muchos dentro del lado cliente, el error dependiendo de la configuración de un determinado cliente a otro pueden ser desde muy sutiles a muy significativos. Normalmente el usuario valida algunas configuraciones del lado cliente para dar conformidad a lo que está probando.

3.6. Pruebas de rendimiento

Cuando un usuario se pone delante de una aplicación Web y al cargar la página el tiempo es frustrante, el usuario está probando el rendimiento de la aplicación ante las diferentes peticiones. El usuario validará en cuestión de tiempos si son razonables o no los diferentes accesos a las diferentes secciones de la aplicación, si determinadas consultas a bases de datos en caso de existir son los suficientemente rápidas, si no habría que optimizarlas. Para ello un concepto importante es el de carga, que es el nivel de peticiones que recibe el servidor concurrentemente y valorar el tiempo de respuesta para cada uno de ellos.

3.7. Pruebas de seguridad

El usuario tendrá que validar que el sistema cumple con todas las restricciones de seguridad exigidas en los requerimientos del mismo. Para ello comprobará los errores con los diferentes firewalls instalados, errores en las autenticaciones, encriptado así como en las autorizaciones. Todas ellas con el fin de descubrir vulnerabilidades o huecos en la seguridad de la aplicación.

4. Conclusiones y trabajo futuro

De todos los estudios revisados hasta el momento en la bibliografía existente, aparece latente la necesidad de la realización del tipo de pruebas anteriormente definidos, con la suficiente calidad para poder ajustarse a todas y cada una de las especificaciones que nos marcan los estándares anteriormente mencionados así como otros específicos como W3C [6], WAI[7], etc. como podemos observar en proyectos reales [9]. Como trabajos futuros, podríamos por un lado, plantear la relación de cada una de las pruebas del tipo anteriormente citadas, en qué atributos de calidad termina redundando de forma positiva para que la calidad final del producto mejore y por otro, la de realizar un profundo estudio de herramientas para el tipo de pruebas de aceptación como Canoo WebTest, JBehabe, Concordion, Fitnesse, etc.

Agradecimientos

Esta investigación ha sido apoyada por el proyecto QSimTest (TIN2007-67843-C06_03) y por el proyecto Tempros (TIN2010-20057-C03-02) del Ministerio de Educación y Ciencia, España.

Referencias

[1] ISO/IEC 9126-1: Software engineering - Product quality - Part 1: Quality model. First edition, 2001.

[2] http://standards.ieee.org/reading/ieee/std_public/description/se/1008-1987_desc.html

[3] Tuya, J., "Hacia el nuevo estándar de pruebas ISO 29119". *XI Jornadas de Innovación y Calidad del Software (JICS)*, Alcalá de Henares, 2009.

[4] Piattini, M. G., Calvo-Manzano, J. A., Cervera, J., and Fernandez, L., "Análisis y diseño de aplicaciones informáticas de gestión, una perspectiva de ingeniería del software", Alfaomega, pp. 419-469, 2004.

[5] Ingeniería del software: Un enfoque práctico. Séptima Edición. Roger S. Pressman, McGrawHill. Capítulos 17, 18 y 19.

[6] http://www.w3c.es/

[7] http://www.w3.org/WAI/

[8] Méndez, E., Pérez, M., Mendoza, L.: Aplicación de un Método para Especificar Casos de Prueba de Software en la Administración Pública. PRIS 2007: Taller sobre Pruebas

en Ingeniería del Software. Actas de Talleres de Ingeniería del Software y Bases de Datos (JISBD 2007), Zaragoza, España, 2007.

[9] Ponce, J., Escalona, M.J., Gómez, A., Luque, M. y Molina, A. "Definición de una política de pruebas en la gestión cultural: aplicación al desarrollo del proyecto Mosaico", REICIS Vol. 6, No. 2, 2010

Análisis de métricas básicas y herramientas de código libre para medir la mantenibilidad

Emanuel Irrazábal[1,2], Javier Garzás[1,2]

1. Kybele Consulting S.L.
{emanuel.irrazabal,javier.garzas}@kybeleconsulting.com

2. Kybele Research
{emanuel.irrazabal,javier.garzas}@urjc.es

Resumen

Durante los últimos años, la calidad se ha convertido en uno de los temas principales de la Ingeniería del Software. En ese sentido, el control de calidad debe realizarse desde un punto de vista cuantitativo y cualitativo siendo necesario establecer mediciones sistemáticas en todo el ciclo de vida del producto software. La mayoría de las pequeñas organizaciones no tienen la capacidad de desarrollar o adquirir herramientas para verificar la calidad de sus productos software y a la vez desarrollar el software que comercializan y que constituye su principal negocio. En ese contexto, las herramientas de código abierto emergen como una opción para obtener el soporte técnico con el cuál recolectar los datos. En este trabajo, se estudia cómo las herramientas de código abierto cumplen con las necesidades de medición de la calidad de acuerdo con la norma ISO/IEC 9126. Nos hemos enfocado en la característica de mantenibilidad debido a su relevancia histórica y a su impacto directo en los costes totales.

Palabras clave: ISO 9126, mantenibilidad, métricas del producto, herramientas

Abstract

During the last years, quality is becoming a hot topic in the context of Software Engineering. The quality control should be done from a quantitative and qualitative point of view, for what it is necessary to establish measurement systems throughout the product lifecycle. Most small organizations can not cope with the development of tools to check the quality of their software products plus the development of the software that constitutes their main business. In this context, open-source tools emerge as the answer to provide with the technical support to collect the information needed to assess the quality of software assets. In this work, we review how existing open-source tools fulfill the needs for quality measures raised when you want to assess product quality according to the ISO/IEC-9126 standard. We focus on maintainability, since it has been historically recognised as one the most relevant, given its direct impact over costs.

Key words: ISO 9126, maintainability, product metrics, tools.

Irrazábal, E. y Garzás, J., "Análisis de métricas básicas y herramientas de código libre para medir la mantenibilidad", REICIS, vol. 6, no.3, 2010, pp.56-65. Recibido: 8-11-2010; revisado: 14-11-2010; aceptado: 24-11-2010

1. Introducción

En un mercado competitivo y en evolución como el actual, la calidad del software y la medición del software son cada vez más importantes [1]. La calidad en el software tiene influencia directa sobre los costes finales, considerándoselo también un factor de diferenciación entre las organizaciones que desarrollan [2].

La calidad del software puede ser descrita desde diferentes puntos de vista, cuando se trata del desarrollo software la calidad se relaciona tradicionalmente con el producto software y con el proceso de desarrollo [3]. En este sentido, existe una tendencia global de institucionalizar las prácticas de calidad en los desarrollos software. Esta tendencia se evidencia en las acciones emprendidas por las organizaciones de desarrollo para cumplir con modelos de referencia bien conocidos, como CMMI-DEV [4] o ISO/IEC 15504 [5]. Un caso particular son las 198 organizaciones de desarrollo software que en España se encuentran certificadas en el modelo CMMI hasta septiembre de 2010 [4] y que posicionan al país en el quinto lugar a nivel mundial y el primer lugar en Europa. Sin embargo, la principal crítica al proceso de evaluación de la calidad ha sido la falta de pruebas para demostrar que seguir un modelo de referencia para los procesos software garantiza la calidad del producto software resultante [6]. De hecho, la institucionalización del proceso de desarrollo puede incluso institucionalizar los malos resultados. En 2008 Maibaum y Wassyng señalaron que las evaluaciones de la calidad deben basarse en evidencias extraídas directamente de los atributos del producto software en lugar de evidencias tomadas del proceso de desarrollo [7]. Como respuesta a esto, han surgido diferentes modelos para la evaluación de la calidad del producto software [8][9][10]. Sin embargo, no hay consenso sobre cómo este tipo de modelos han de ser utilizados en las organizaciones.

En particular, la Organización Internacional de Normalización (ISO) publicó la norma ISO / IEC 9126:1991 Ingeniería de Software - Calidad del producto [8]. Esta norma especifica un modelo de calidad general, identificando las siguientes características de calidad: funcionalidad, fiabilidad, eficiencia, usabilidad, mantenibilidad y portabilidad, y la forma en que se descomponen en subcaracterísticas. Cabe destacar que existe actualmente una nueva norma asociada con la calidad del producto, la norma ISO/IEC 25010 [9] pero que actualmente se encuentra en borrador (estado 50.20) de acuerdo al catálogo oficial de

ISO[1]. Por todo ello este trabajo se centrará en la característica de mantenibilidad de la norma ISO/IEC 9126, ya que ha sido históricamente reconocida como uno de los más relevantes, teniendo en cuenta su impacto directo sobre el coste de desarrollo y mantenimiento del producto software. Estudios previos señalan al mantenimiento como la fase que más recursos requiere a lo largo del ciclo de vida del producto, dos veces superior a los costes de desarrollo [11][12]. A pesar del hecho de que el modelo ISO 9126 no especifica un conjunto de medidas para evaluar la capacidad de mantenimiento de productos software [13] sí establece las características deseables de dichas mediciones. Tienen que ser objetivas, independientes y reproducible, expresada por medio de escalas válidas y suficientemente precisas para apoyar comparación fiable [14]. Con estas premisas, el enfoque más conveniente parece el uso de herramientas comerciales para la captura de las mediciones. Sin embargo, algunas organizaciones no pueden permitirse estas herramientas. Varios estudios han revelado que la aplicación de modelos de calidad en las pequeñas y medianas empresas (PYME) es una tarea muy difícil [15][16] ya que implica grandes inversiones de dinero, tiempo y recursos humanos. Por consiguiente, en estas organizaciones es necesario adaptar las prácticas de ingeniería de software a su tamaño y a la naturaleza de sus actividades [17]. En particular, las PYMEs han buscado el apoyo de los proyectos de código abierto para automatizar la evaluación de la calidad del software que desarrollan. Las soluciones de código abierto han demostrado ser tan eficaces como las comerciales [18][19] mientras que implican un menor coste de licencia y mantenimiento.

Este trabajo evalúa el apoyo técnico de herramientas de código abierto actuales que recogen métricas básicas para evaluar la característica de mantenibilidad especificada en ISO/IEC 9126. La sección 2 presenta un resumen de las métricas básicas propuestas por los modelos de medición actuales y la sección 3 detalla los criterios de selección y lista las herramientas obtenidas. Por último en la sección 4 se resumen las conclusiones del estudio.

2. Elección de las métricas asociadas a la mantenibilidad

Debido a la imposibilidad de encontrar un único modelo de medición lo suficientemente reconocido y que detalle el conjunto de métricas básicas recomendadas para medir la mantenibilidad, se presenta un resumen de los modelos de medición más destacados. Se

[1] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35733

han recopilado diversos modelos de medición de la mantenibilidad que relacionan un conjunto de métricas de calidad obtenidas a partir del código fuente con las subcaracterísticas de la mantenibilidad. Se han tenido en cuenta los modelos de medición de herramientas libres y de código propietario, modelos de medición reconocidos por entidades de certificación, así como otros modelos de medición [20][21][22][23][24][25].

Métricas de calidad	Analizabilidad	Cambiabilidad	Estabilidad	Capacidad de ser probado
Complejidad ciclomática (CC)	X	X		X
Cantidad de las instrucciones (LOC)	X			
Promedio del tamaño de las instrucciones (PLOC)	X	X		X
Frecuencia de los comentarios (PCOM)	X			
Peso de los métodos por clase (WMC)	X			
Número de clases base (NCB)	X			
Número de saltos incondicionales (NSI)		X	X	X
Número de niveles anidados (NNA)		X		X
Acoplamiento entre objetos (CBO)	X	X	X	X
Ausencia de cohesión (LCOM)	X	X	X	X
Profundidad del árbol de herencia (DIT)	X	X	X	X
Componentes llamados directamente (CCD)			X	
Número de hijos (NOC)	X	X	X	X
Número de salidas de estructuras condicionales (NSE)				X
Respuestas de una clase (RFC)				X
Número de mensajes enviados en un método (NOM)	X	X	X	X
Cobertura de las pruebas unitarias (COB)			X	X
Número de errores de las pruebas unitarias (UTE)			X	X
Violaciones en el código fuente (VCF)	X			
Violaciones de estilo (VST)	X			
Distancia a la secuencia principal (D)		X	X	
Número de dependencias cíclicas (CYC)	X	X		
Acoplamiento aferente y eferente (CAF, CEF)		X	X	
Código duplicado (CDU)	X	X		

Tabla 1. Resumen de las métricas internas relacionadas con las subcaracterísticas de la mantenibilidad de acuerdo a la norma ISO/IEC 9126

Como resultado se ha obtenido la Tabla 1 que resume el conjunto de métricas internas básicas que se encuentran relacionadas con las subcaracterísticas de analizabilidad,

cambiabilidad, estabilidad y capacidad de ser probado, las cuales definen a la característica de mantenibilidad de acuerdo con la norma ISO/IEC 9126. Se han agregado acrónimos a las métricas para facilitar la construcción de tablas posteriores.

3. Elección de las herramientas

Una vez obtenido el resumen de las métricas básicas se ha pasado a la selección de las herramientas de código libre con las que se puede analizar el código fuente del producto software a ser evaluado. Se ha seguido el enfoque utilizado en [26], revisando los repositorios de proyectos de código abierto existentes y seleccionando el repositorio principal a examinar. La búsqueda se ha complementado con la selección de herramientas realizada en trabajos similares.

De acuerdo con estudios previos [27] el repositorio SourceForge alberga actualmente la mayor cantidad de proyecto de código abierto sobre otros como Codeplex, Google Code o Kenai. En particular, apoya a más de 260.000 proyectos de software con una comunidad de más de 2.7 millones de usuarios registrados. A continuación, se han identificado las palabras típicas con las que llevar a cabo la búsqueda en el repositorio y así obtener las herramientas de medición más valoradas por los usuarios. De acuerdo con la lectura de la norma ISO 9126 se ha concluido una serie de palabras claves utilizadas para realizar la búsqueda en el sitio Web de www.sourceforge.net. En las búsquedas se han tenido en cuenta solo los lenguajes de programación más populares. Se han identificado las herramientas asociadas con los lenguajes .NET (tanto en VB.NET y C #), Java y C/C++, ya que estos son los marcos de desarrollo más populares de acuerdo con el índice TIOBE[2], actualizado en noviembre de 2010.

En la Tabla 2 se enumeran las principales herramientas encontradas de acuerdo con el nivel de actividad de la comunidad en términos de número de visitas en el sitio Web correspondiente, las entradas de los foros relacionados con cada herramienta, descargas, etc. Estas estadísticas han sido obtenidas en enero de 2010.

[2] http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

Herramientas	% de Actividad	Descargas	Frase de búsqueda	Lenguaje de programación
• CheckStyle	• 99.94%	• 4.473.961	• Source Code Analysis	• JAVA
• FindBugs	• 99.93%	• 606.868	• Static Analysis	• JAVA
• PMD	• 99.38%	• 508.312	• Source Code Analysis	• JAVA
• CCCC	• 92.76%	• 64.523	• Source Code Analysis	• C
• JUnit	• 99.74%	• 2.900.703	• Test	• JAVA
• NUnit	• 99.68%	• 1.908.561	• Test	• .NET
• CPPUnit	• 98.97%	• 545.474	• Test	• C
• EMMA	• 99.95%	• 1.487.428	• Source Code Analysis, Test	• JAVA

Tabla 2. Herramientas encontradas

Herramientas	Métricas calculadas
JavaNCSS	CC, LOC
PMD/CPD	VCF, CDU
CheckStyle	VST
FindBugs	VCF, CDU
JDepend	CYC,D,CAF,CEF
CCCC	CBO,DIT,NOM, NOC ,NOM ,RFC, LOC, PCOM, CC, WMC, D,CEF, CAF
StyleCop	VCF
FxCop	VST
JUnit	UTE
CPPUnit	UTE
NUnit	UTE
EMMA	COB
Analyst4j	CC, LOC, PLOC, PCOM, CBO,DIT,LCOM,NOC,NOM,RFC,WMC
C&K Java Metrics	CBO, DIT, LCOM, NOC, NOM, RFC
Dependency Finder	LOC, NNA, CCD, DIT, NOC, NOM
Eclipse Metrics	CC, LOC, CAF, CEF, D, LCOM, CEF, DIT, NOC, NOM
OOMeter	CBO, DIT, LCOM, NOC
Semmie	DIT, LCOM, NOC, NOM, RFC
Understand for Java	CBO,DIT, LCOM, NOC, NOM
VizzAnalyzer	LOC, CBO, DIT, LCOM, NOC, NOM, RFC, WMC
CodePro AnalytiX	CC, LOC, PLOC, PCOM, NCB, NSI, NNA, CCD, D, CYC, CAF, CEF, CBO, DIT, LCOM, NOC, NOM, RFC, WMC, CDU

Tabla 3. Resumen de las herramientas

Para completar la selección anterior, se incluyen herramientas adicionales. En primer lugar, se han considerado las herramientas de análisis estático de código de software libre para. NET, FxCop y StyleCop [28]. También se incluyen herramientas que miden el

acoplamiento y la complejidad ciclomática ya que han sido ampliamente reconocidos como los indicadores más valiosos para evaluar la capacidad de mantenimiento de sistemas orientados a objetos [29]. Se han incluido las herramientas JavaNCSS y JDepend como las más representativas de estas métricas, aunque hayan podido ser incluidas otras similares. Por último se incluyen otras herramientas libres analizadas en otros trabajos [30][31]. En la Tabla 3 se enumeran todas las herramientas seleccionadas y se especifican las mediciones que cada una realiza.

4. Conclusión

Una vez analizados los resultados obtenidos se ha concluye que las herramientas actuales de código abierto sirven como un buen punto de partida hacia la obtención de los indicadores necesarios para apoyar el modelo de medición descrito en la norma ISO 9126. Con el fin de simplificar el uso de estas herramientas, la gran mayoría de ellas pueden ser incorporadas directamente en IDEs de desarrollo. No todas las herramientas obtienen los mismos resultados [30] ni tienen la misma fiabilidad en cuanto a falsos positivos y falsos negativos a la hora de buscar violaciones en el código fuente [31], por lo que es indispensable una tarea de selección de herramientas. Se destaca a la herramienta Code Pro AnalytiX, que evalúa código fuente escrito en lenguaje Java, como una de las más completas en cuanto a cantidad de métricas básicas que obtiene. Aún así, es conveniente comprobar antes la precisión de todas estas herramientas en casos reales, especialmente a las que realizan cálculos más complejos.

Para trabajos futuros se considera importante continuar con este tipo de análisis, estudiando la relación entre las herramientas de código libre y las necesidades de los modelos de procesos existentes, como el Proceso de Medición y Análisis (PMA) de CMMI o el proceso de medición descrito en la norma ISO / IEC 12207:2008 [14]. En este caso será necesario también incluir otro tipo de herramientas, para derivar métricas y relacionar los resultados con las necesidades de negocio de la organización.

Agradecimientos

Este trabajo ha sido parcialmente financiado por el Centro para el Desarrollo Tecnológico Industrial (CDTI) (IDI-20090175), dependiente del Ministerio de Ciencia e Innovación y el

proyecto MODELO-CAOS (TIN2008-03582/TIN) financiado por el Ministerio de Ciencia y Educación (TIN2005-00010/).

Referencias

[1] Rifkin, S., "Guest editor's introduction: software measurement", *IEEE Software*, vol. 26 nº 3, pp. 70, 2009.

[2] Piattini, M., García, F., Garzás, J. and Genero, M., *Medición y estimación del software: técnicas y métodos para mejorar la calidad y productividad del software*, Ra-ma, 2008.

[3] Garvin, D.A., "What does "product quality" really mean?", *Sloan management review*, vol. 26, p.25, 1984.

[4] ISO, *ISO/IEC 15504-5 Information technology — process assessment — part 5: an exemplar process assessment model*, ISO, 2006.

[5] Ebert C., "Guest editor's introduction: how open source tools can benefit industry", *IEEE Software*, vol. 26, pp. 50-51, 2009.

[6] Kitchenham, B. and Plfeeger, S.L., "Software quality: the elusive target", *IEEE Journal*, vol. 13 nº 1, pp. 12-21,1996.

[7] Maibaum T, Wassyng, A.,"A product-focused approach to software certification", Computer, vol. 41 nº 2, pp. 91-93, 2008.

[8] ISO, *ISO/IEC Std. 9126 Software product evaluation–quality characteristics and guidelines for their use*, ISO, 2001.

[9] ISO, *ISO/IEC Std. 25010 Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models*, ISO, 2010.

[10] Milicic, D. y otros, *Software quality attributes and trade-offs*, Blekinge Institute of Technology, 2005.

[11] Frazer, A., "Reverse engineering- hype, hope or here?", Software Reuse and Reverse Engineering in Practice, vol. 12, pp. 209-243. 1992.

[12] Pressman, R. *Ingeniería del software: un enfoque práctico*, McGraw-Hill, 2002

[13] Boehm, B., Characteristics of software quality, North Holland Press, 1978.

[14] ISO, *ISO/IEC 12207 Systems and software engineering - software life cycle processes*, ISO, 2008.

[15] Saiedian, H. and Carr, N., "Characterizing a software process maturity model for small organizations" , *ACM SIGICE Bulletin*, vol. 23 nº 1, pp. 2-11, 1997.

[16] Staples, M., Niazi, M., Jeffery, R., Abrahams, A., Byatt, P. and Murphy, R., "An exploratory study of why organizations do not adopt CMMI", *Journal of systems and software*, vol. 80 nº 6, pp. 883-895, 2007.

[17] Dybå, T, "Factors of software process improvement success in small and large organizations: an empirical study in the scandinavian context". En *European Software Engineering Conference (esec) / Foundations of Software Engineering (sigsoft fse), Helsinki (Finlandia), 1-5 de septiembre de 2003*, pp.148-157, 2003.

[18] SEI, *Process maturity profile - cmmi for development scampi class a appraisal results 2010 mid-year update - septiembre 2010*, SEI, 2010.

[19] Hawkins, R.E., "The economics of open source software for a competitive firm", *Netnomics*, vol. 6 nº 2, pp. 103-117, 2004.

[20] Luijten, B., Visser, J. and Zaidman, A., "Faster defect resolution with higher technical quality of software", *Proceeding of the 4th International Workshop on Software Quality and Maintainability (SQM'10), Madrid (España), 15-18 de marzo de 2010*, pp. 11-20 ,2010.

[21] Alshayeb, M., "Empirical investigation of refactoring effect on software quality" , *Information and software technology*, vol. 51, pp.1319-1326, 2009.

[22] Heitlager, I., Kuipers, T. and Visser, J., "A practical model for measuring maintainability". En *Quality of Information And Communications Technology (Quatic 2007), Lisboa (Portugal), 12-14 de septiembre de 2007*, pp. 30 - 39, 2007.

[23] Kataoka, Y., Imai, T., Andou, H. and Fukaya, T. 2002. "A quantitative evaluation of maintainability enhancement by refactoring", En *18th IEEE Iinternational Conference On Software Maintenance (ICSM'02),Montreal (Canadá), 3-6 de octubre de 2002*, pp.576-585, 2002.

[24] Mouchawrab, S. and Briand, L.C. "A measurement framework for object-oriented software testability", *Information and software technology*, vol. 47 nº 15, pp. 979-997, 2005.

[25] Samoladas, I., Gousios, G., Spinellis, D. and Stamelos, I., "The SQO-OSS quality model: measurement based open source software evaluation". *Open source development, communities and quality*, vol. 275, pp. 237-248. 2008.

[26] von Wangenheim C.G, Hauck, J., von Wangenheim, A., "Enhancing open source software in alignment with CMMI-Dev", *IEEE Software*, vol. 26 nº 2, pp. 59-67, 2009.

[27] Dror, G.F., Gillian, Z.H. and Stephen, R.S., "An empirically-based criterion for determining the success of an open-source project", IEEE Computer Society, Proceedings of the Australian Software Engineering Conference, Sidney (Australia), 18-21 de abril 2006, pp.363-368, 2006.

[28] Mitchell, S., "Static analysis tools for .net", MSDN Magazine, vol. 94, 2008.

[29] Li, W. and Henry S., "Object-oriented metrics that predict maintainability". Journal of systems and software, vol. 23 nº 2, pp. 111-122. 1993.

[30] Lincke, R., Lundberg, J. and Löwe, W., "Comparing software metrics tools". En ACM, Proceedings of the 2008 International Symposium on Software Testing and Analysis, Seattle (USA), 20-24 de julio de 2008, pp. 131-142, 2008.

[31] Plösch, R., Mayr, A., Pomberger, G. and Saft, M., "An approach for a method and a tool supporting the evaluation of the quality of static code analysis tools", *In proceedings of SQMB 2009 Workshop, held in conjunction with SE 2009 conference*, Kaiserslautern (Alemania), 2 de marzo de 2009, pp. ,2009.

Reduciendo distancia en proyectos de Desarrollo de Software Global Ágiles con técnicas de ingeniería de requisitos

Mariano Minoli
Assertum Tecnologías
mariano.minoli@assertum.es

Valeria de Castro, Javier Garzás
Universidad Rey Juan Carlos
{valeria.decastro, javier.garzas}@urjc.es

Resumen

La tendencia a incluir agilidad en los métodos de desarrollo de software ha sido un tema recurrente durante los últimos años en ámbitos académicos y empresariales. Los equipos de desarrollo que trabajan de forma distribuida (conocidos como equipos *GSD*, *Global Software Development*) no han sido una excepción. Múltiples propuestas abogan por la adopción de técnicas ágiles para el desarrollo y la administración de estos equipos de desarrollo. Aunque a primera vista esta evolución hacia lo ágil parece natural, en realidad surgen múltiples interrogantes relacionados a combinar *agilidad* con el factor más complicado de tratar en equipos distribuidos: la distancia. Desde nuestro punto de vista, técnicas existentes de ingeniería de software pueden ser útiles para abordar esta problemática. En este artículo presentamos el uso de técnicas de ingeniería de requisitos como herramienta para estrechar la brecha que se produce al mezclar agilidad y distancia.

Palabras clave: Desarrollo Global de Software, Métodos Ágiles, Ingeniería de Requisitos.

Using Requirement Engineering Techniques to reduce Distance on Agile Global Software Development Projects

Abstract

The trend of including agility on software development methods has been recurrent in recent years both in academia and business. Global Software Development (GSD) teams have not been an exception, many proposals advocate for the adoption of agile techniques for this kind of teams. Although at first glance the trend towards agile seems natural, there actually are many questions related to combining agility with distance. From our point of view, existing software engineering techniques can be useful in addressing this problem. In this paper we will present the use of requirements engineering techniques as a tool to bridge the gap that occurs when mixing these two things: agility and distance.

Key words: Global Software Development, Agile Methods, Requirement Engineering.

Minoli, M., Castro, V. y Garzás, J.,"Reduciendo distancia en proyectos de Desarrollo de Software Global Ágiles con técnicas de ingeniería de requisitos", REICIS, vol. 6, no.3, 2010, pp.66-75. Recibido: 8-11-2010; revisado: 14-11-2010; aceptado: 19-11-2010

1. Introducción

Dos de las tendencias que han marcado el camino en el mundo del desarrollo de software en los últimos años han sido el Desarrollo Global de Software (*GSD, Global Software Development*) y las metodologías Ágiles. GSD es la evolución natural del negocio de desarrollo de software a la descentralización de tareas (como la programación o las pruebas) hacia sitios remotos, normalmente más rentables [1]. Estos sitios remotos pueden ser algo tan cercano como otra empresa dentro de la misma ciudad o país, o algo tan lejano como empresas en otro continente, incluso con diferente zona horaria, cultura, etc. Esta tendencia ha sido motivada por cuestiones de negocio, más que por cuestiones relacionadas a la ingeniería de software. Es por ello que muchas de las metodologías de desarrollo de software están siendo adaptas a estos entornos [2] [3]. Los métodos de desarrollo ágil [4] han ganado protagonismo en los últimos años como la reacción de la comunidad a las metodologías formales, sobre-documentadas, que tradicionalmente han dominado el mundo de la ingeniería de software. La necesidad de combinar estas tendencias parece ineludible, sin embargo genera cuestiones que deben ser respondidas [5] [6]. La investigación existente en este área se ha focalizado en analizar el impacto de la distancia en proyectos de GSD (ágiles y no ágiles) [7] [8] [9], sin embargo existen pocos trabajos que propongan prácticas para resolver los problemas (normalmente de comunicación) generados por la distancia.

Nuestra investigación nos ha llevado a plantear la hipótesis de que se podrían utilizar herramientas existentes de ingeniería de software para apoyar la implantación metodologías ágiles en entornos de desarrollo global de software (A-GSD). En particular hemos escogido la utilización de técnicas de ingeniería de requisitos (IR). Históricamente la mala gestión de requisitos ha sido motivo de frecuentes fracasos de proyectos de desarrollo [11]. En este trabajo se propone la adaptación y el uso de técnicas conocidas de IR con el objetivo de disminuir la problemática relacionada a la comunicación en equipos de A-GSD. El objetivo de utilizar estas técnicas (con cierto grado de adaptación) es el de cubrir la brecha existente entre los miembros de equipos que se encuentras distribuidos geográfica, temporal y culturalmente sin afectar el carácter ágil de los procesos empleados por equipo de trabajo.

El presente artículo se estructura de la siguiente manera. En la sección 2 se presentan los trabajos relacionados con esta propuesta. En la sección 3 se presenta la aportación de este trabajo. La sección 4 resume las conclusiones y futuros trabajos.

2. Trabajos relacionados

En esta sección se presentarán las principales áreas que fundamentan nuestra propuesta. En la sección 2.1 expondremos la problemática del desarrollo en entornos A-GSD. En la sección 2.2 presentaremos trabajos realizados en torno a gestión de requisitos en entornos GSD (no ágiles) que serán utilizados como base para nuestra propuesta.

2.1. Desarrollo Global de Software Ágil (A-GSD)

El mundo de los negocios TIC ha evidenciado en los últimos años una irreversible tendencia hacia la descentralización de las operaciones. La aparición de Internet a mediados de los años noventa y distintas circunstancias económicas han favorecido la práctica de brindar servicios desde centros de producción remotos. El desarrollo de software ha sido una de las disciplinas donde se ha hecho más frecuente esta relocalización de la fuerza de trabajo. Hoy en día es ampliamente aceptado el hecho de que muchas de las principales compañías producen total o parcialmente su software en ubicaciones diferentes a donde se encuentra el cliente final. Esta tendencia [1] se conoce con el nombre de Desarrollo Global de Software (GSD).

En paralelo, en los últimos años han ganado popularidad un conjunto de nuevas técnicas y métodos de desarrollo con el nombre de metodologías ágiles [12]. Existen estudios [13] [14] que demuestran que en ciertos entornos (normalmente equipos pequeños con un alto grado de implicación del usuario) estas técnicas pueden resultar más productivas que los métodos de desarrollo tradicionales. Parece lógico que los equipos de GSD estén siendo conducidos a seguir esta tendencia para poder aprovechar todo su potencial [2]. Sin embargo, la combinación de GSD y métodos ágiles, no es trivial. Aunque la primera reacción a mezclar Métodos Ágiles y GSD ha sido de escepticismo [15], trabajos recientes [5] [6] han demostrado su factibilidad, siempre que ciertos aspectos sean tenidos en cuenta. El denominador común detrás de los aspectos que deben ser considerados es la *comunicación* entre los integrantes del equipo [8] [16] [17]. Si tenemos en cuenta que la mayor parte de las prácticas propuestas por las metodologías ágiles se basan en la interacción entre los participantes, la brecha resulta evidente. Existen varios estudios que presentan diferentes enfoques para disminuir esta brecha [7] [9] [18], sin embargo ninguno de ellos lo hace desde el punto de vista de la ingeniería del software. Además ninguno de ellos tiene en cuenta el aspecto ágil en los equipos de proyecto GSD.

2.2. Gestión de Requisitos en entornos GSD

Desde la aparición de GSD como tendencia mundial, varios autores han escrito sobre el impacto de la distribución geográfica de equipos de trabajo en la ingeniería de requisitos [19] [20] [21]. Estos estudios han concluido en la necesidad de adaptar las técnicas de IR a los entornos GSD [19]. Según [22], los mayores desafíos en la gestión de requisitos en entornos distribuidos de trabajo son: Lograr una correcta comunicación, facilitar herramientas de gestión del conocimiento y mitigar diferencias culturales y de zonas horarias. El problema de estos enfoques es que ninguno de ellos ha tenido en cuenta el aspecto ágil, han sido pensados para GSD, no para A-GSD. Es por ello que nuestra propuesta selecciona y adapta un conjunto de técnicas de IR tradicionales [23] y distribuidas [19] adaptadas con el objetivo concreto de disminuir la brecha de comunicación causada por la combinación de distribución y agilidad en equipos A-GSD. En la próxima sección se proponen adaptaciones de estas técnicas.

3. Requisitos en A-GSD

Una buena comunicación es la piedra angular de los proyectos A-GSD. Todas las disciplinas asociadas a la IR implican comunicación. La IR tradicional propone una serie de técnicas para estas disciplinas. Pueden utilizarse estas técnicas para mantener principios del manifiesto ágil como promover a los individuos y las interacciones por sobre los procesos y las herramientas, el software en funcionamiento sobre la documentación, la colaboración del cliente sobre la negociación de contratos y la respuesta al cambio sobre los planificaciones rigurosas [24]. En la siguiente sección se propone la utilización de técnicas de IR como complemento a las técnicas ágiles con el objetivo de ser usadas en A-GSD.

3.1. Uso de técnicas de IR en A-GSD

La mayor parte de los autores [23] [25] [26] [27] [28] están de acuerdo en que las principales disciplinas relacionadas a la IR son: elicitación, análisis y negociación, documentación, validación y gestión de los requisitos. Para los propósitos de este trabajo, se usarán estas disciplinas para agrupar las herramientas propuestas. Según [28], algunos de los principales desafíos que aparecen al incluir la componente ágil a proyectos distribuidos son: falta de visibilidad del estado del proyecto, pérdida del contexto técnico y de negocio,

falta de una infraestructura común, disminución del caudal de comunicación y disminución de la confianza.

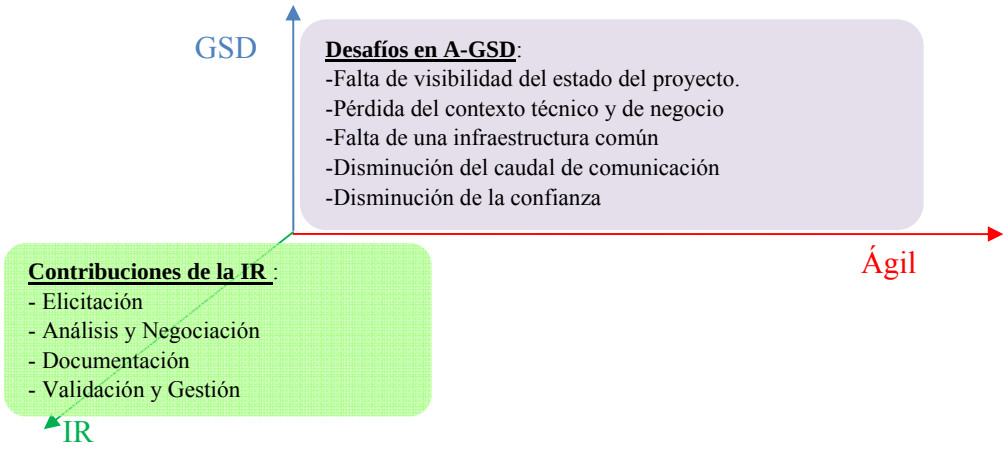


Figura 1. Contribuciones de la IR en entornos A-GSD

Problemática en A-GSD	Disciplinas IR	Técnicas
Falta de visibilidad del estado del proyecto	Análisis y Negociación	Priorización de requisitos
	Validación y Gestión	Trazabilidad de requisitos Repositorio único de requisitos
Pérdida del contexto técnico y de negocio	Elicitación	Técnicas tradicionales de elicitación de requisitos (cuestionarios y encuestas, entrevistas, análisis de documentación existente) Técnicas de elicitación grupales (*brainstorming* y grupos de enfoque)
	Análisis y Negociación	Joint Application Development (JAD)
Falta de una infraestructura común	Documentación	Prototipos Técnicas de ingeniería conducida por modelos
	Validación y Gestión	Repositorio único de requisitos
Disminución del caudal de comunicación	Análisis y Negociación	Joint Application Development (JAD) Registro de la interacciones con el usuario
	Documentación	Casos de Uso Sitios Wiki
Disminución de la confianza	Elicitación	Técnicas tradicionales de elicitación de requisitos (cuestionarios y encuestas, entrevistas, análisis de documentación existente). Técnicas de elicitación grupales (*brainstorming* y grupos de enfoque).
	Análisis y Negociación	Registro de la interacciones con el usuario

Tabla 1. Relación de técnicas de IR propuestas

Nuestra hipótesis plantea que todos estos problemas pueden ser resueltos aplicando técnicas de IR. La Figura 1 muestra estos desafíos (surgidos a incluir el aspecto Ágil en GSD) y como podrían aportar la inclusión de técnicas de IR. En la Tabla 1 se resumen las

técnicas propuestas, agrupadas por cada disciplina. En las próximas secciones se describirán cada una de estas técnicas.

No es el propósito de este trabajo describir estás conocidas técnicas, en las próximas secciones nos limitaremos a explicar aspectos relacionados a cómo utilizarlas en entornos A-GSD.

3.2.1 Uso de técnicas de elicitación de requisitos en A-GSD

La información recolectada durante la elicitación de requisitos siempre debe ser interpretada, analizada, modelada y validada antes de que los ingenieros de requisitos puedan documentarla [23]. En entornos A-GSD, las técnicas tradicionales de elicitación de requisitos pueden aplicarse sin cambios significativos. Técnicas de elicitación de requisitos tradicionales como cuestionarios y encuestas, entrevistas y análisis de documentación existente pueden ser usadas sin necesidad de adaptarlas a entornos A-GSD. Aquí es importante tener en cuenta la utilización de medios digitales que faciliten el intercambio de la información a través de Internet. Por otro lado, gracias a herramientas digitales de comunicación grupal, es posible también el empleo de elicitación de requisitos grupales como *brainstorming* y grupos de enfoque. En [30] puede encontrarse una recopilación de productos de comunicación grupal.

3.2.2. Uso de técnicas de análisis y negociación de requisitos en A-GSD

Los procesos de análisis y negociación están íntimamente relacionados a la elicitación de requisitos. La técnica JAD (*Joint Application Development*) es una sesión de grupo (o taller), con un enfoque de análisis estructurado. Durante las sesiones JAD los desarrolladores y los clientes pueden discutir las características del producto [31]. Este tipo de discusión puede llevarse a cabo en una forma distribuida utilizando las tecnologías actuales de comunicación.

3.2.3. Uso de técnicas de documentación de requisitos en A-GSD

Las técnicas de requisitos propuestas para equipos ágiles [32] [33] tienden a minimizar la documentación generada durante el proyecto. Sin embargo, para el caso de A-GSD, consideramos que este comportamiento debe ser modificado. Según estudios analizados [34] existe una relación de complementación entre las técnicas de documentación de IR tradicional (como los casos de uso) y las técnicas de requisitos ágiles (como los Story

Cards). Nuestra hipótesis postula que, en A-GSD, la utilización de Casos de Uso mejora la comunicación debido a que permite la formalización de requisitos. Esta formalización permite facilitar la comunicación cuando los integrantes del equipo no se encuentran en la misma ubicación física. Otras técnicas de documentación como los prototipos o las técnicas de ingeniería dirigida por modelos pueden ser usadas para complementar los casos de uso. Una decisión importante que deberá tomar el equipo es el límite con respecto a cuánto documentar. En [35], Scott Ambler defiende: "El objetivo es implementar requisitos, no documentarlos". Nuestra consideración al respecto en entornos distribuidos es que, aunque este espíritu debe mantenerse, no menos importante es considerar que cierto grado de formalización contribuirá a disminuir la brecha de comunicación en equipos A-GSD.

3.2.4. Uso de técnicas de validación y gestión de requisitos en A-GSD

Las técnicas de documentación descritas en la sección anterior deben servir como base para la validación por parte del usuario y su posterior gestión. Es muy importante disponer de un repositorio único que aglutine todos los documentos generados durante la etapa de documentación. Algunos autores recomiendan la utilización de Wikis [36] como herramienta que centralice todos los esfuerzos de documentación y a la vez admitan información no estructurada.

4. Conclusiones y trabajo futuro

En este trabajo se ha propuesto la introducción de técnicas de IR en A-GSD para reducir la *distancia* y obtener una mejor comunicación entre los integrantes de equipos distribuidos. Se ha seleccionado un conjunto de técnicas conocidas que pueden aportar valor a procesos de desarrollo distribuido ágil, y se ha incluido una descripción de cómo estas técnicas pueden ser utilizadas.

Aunque a primera vista la introducción de estas técnicas tradicionales podría ser vista como un contraste a los métodos ágiles, nuestra opinión es que complementan muy bien a los métodos ágiles en equipos distribuidos. Así, una mayor documentación es necesaria en equipos A-GSD, en contraste a que ocurre en equipos ágiles que se encuentran ubicados en el mismo sitio.

Son necesarios más estudios para determinar cómo deberían ser aplicadas estas técnicas en metodologías concretas como eXtreme Programming, SCRUM o Crystal

Family entre otras. Por otro lado, todas las herramientas propuestas necesitan ser probadas en entornos de desarrollo reales. Ambas, propuestas y experiencias serán presentadas en trabajos futuros.

Agradecimientos

Este trabajo está parcialmente financiado por el proyecto MODEL CAOS financiado por el Ministerio de Ciencia y Tecnología de España (Ref. TIN2008-03582) y Agreement Technologies (CONSOLIDER CSD2007-0022).

Referencias

[1] Herbsleb, JD, Moitra, D., "Global Software Development" *IEEE Software, 2001*.

[2] Holmström, H, Fitzgerald, B, Ågerfalk, PJ, "Agile practices reduce distance in global software development", *Information Systems Summer 2006 ABI/INFORM Global*, pp. 7.

[3] Vax, M., Michaud, S., "Distributed Agile: Growing a Practice Together" *Agile Conference,* 2008

[4] Cockburn, A. *Agile Software Development* Addison Wesley, 2001.

[5] Paasivaara, M., Lassenius, C., "Could Global Software Development Benefit from Agile Methods?" *International Conference on Global Software Engineering, 2006.*, pp. 109 - 113.

[6] Ramesh, B, Cao, L,Mohan, K y Xu., P, "Can distributed software development be agile?" *Communications of the ACM*, vol. 49, nº 10, 2006.

[7] Prikladnicki, Rafael, Audy, Jorge Luis Nicolas, Evaristo, Roberto, "Global Software Development in Practice Lessons Learned", *Software Process Improvement 2003*, pp. 267–281.

[8] Herbsleb, JD "Global software engineering: The future of socio-technical coordination". *Int. Conference on Software Eng, Future of Software Engineering*, pp. 188-198, 2007

[9] Carmel, E. Agarwal, R. "Tactical approaches for alleviating distance in global software development.", *IEEE Software*, vol 18, nº 2, pp. 22 – 29, 2001.

[10] Prior, P., "Requirements Management in a Distributed Agile Environment.", 2º *WEC,* 2005.

[11] McConnell, S., "Avoiding S. E. classic mistakes", *Softare IEEE,* vol. 13, nº 5, 1996.

[12] Dyba, T., Dingsoyr, T., "What Do We Know about Agile S. D.?", Soft. IEEE, pp. 6-9, 2009

[13] Cohn, M., *User Stories Applied: For Agile Software Development.* Addison-Wesley, 2004.

[14] Sutherland, J., Schwaber, K, *The Scrum Papers.* Scrum, 2007.

[15] Daniel B. Markham, Roanoke, Virginia, "Agile Won't Work Implementing Agility in Non-standard Teams", *IEEE DOI 10.1109/AGILE. 32*, 2009.

[16] Paasivaara, M., Lassenius, C., "Collaboration practices in global inter-organizational software development projects", *GSD: Growing Opportunities, Ong. Challenges*, pp. 183–199.

[17] Herbsleb, JD, Mockus, A, Finholt, TA, Grinter, RE, "An empirical study of global software development: distance and speed", *23rd Int. Conf. on Software Eng. Toronto,* pp. 81-90, 2001.

[18] Ye , Y., Nakakoji, K., Yamamoto, Y., "Measuring and Monitoring Task Couplings of Developers and Development Sites in Global Software Development", *Software Engineering Approaches for Offshore and Outsourced Development*, pp. 181-195, 2009.

[19] Damian, D, "Stakeholders in Global Requirements Engineering: Lessons Learned from Practice", *Software IEEE,* vol 24 , nº 2, pp. 21-27, 2007.

[20] Lopez, A., Nicolas, J. y Toval, A, "Risks and Safeguards for the Requirements Engineering Process in Global Software Development", *ICGSE*, pp. 394 – 399, 2009.

[21] Sabahat, Nosheen "An iterative approach for global requirements elicitation: A case study analysis", *Electronics and Information Engineering (ICEIE)*, pp. V1-361 - V1-366, 2010.

[22] Damian, D., Zowghi, Didar, "Requirements Engineering challenges in multi-site software development", *Requirements Engineering Journal*, pp. 149-160, 2003.

[23] Nuseibeh, B., Easterbrook, S., "Requirements engineering: a roadmap", *Proceedings of the Conference on The Future of Software Engineering table of contents*, pp. 35 – 46, 2000.

[24] Beck, K. *Manifesto for Agile Software Development*. http://agilemanifesto.org/. 05/11/10.

[25] Cheng, B., "Research directions in requirements", *Future of Soft. Eng.* pp. 285-303, 2007.

[26] Jiang, L., Eberlein, A., & Far, B., "Combining requirements engineering teclmiques - theory and case study", *12th Conf. on the Computer-Based Systems, ECBS '05*, pp. 105-112, 2005.

[27] Kotonya, G., Sommerville, I., *Requirements Engineering*. John Wiley & Sons. 1997

[28] Damian, D., Zowghi, D., "Requirements Engineering challenges in multi-site software development", *Requirements Engineering Journal*, pp. 149-160, 2003.

[29] Vax, M., Michaud, S., "Distributed Agile: Growing a Practice Together", *Agile Conf.. 2008*.

[30] Lanubile, F., Ebert, C., Prikladnicki, R.,Vizcaíno, A., "Collaboration Tools for Global Software Engineering" *IEEE SOFTWARE*, pp. 52-55, 2010.

[31] Paetsch, F., Eberlein, A., "Requirements Engineering and Agile Software Development", *12º Workshop on Enabling Technologies: Infrastructure for Collaborative Ent.*, pp. 308, 2003.

[32] Racheva, Z., Daneva, M., "How Do Real Options Concepts Fit in Agile Requirements Engineering?" *Eighth ACIS International Conference on Software Engineering Research, Management and Applications.* pp. 231-238, 2010.

[33] Cao, Lan, Ramesh, B., "Agile Requirements Engineering Practices: An Empirical Study" *Software IEEE,* vol 25, nº 1, pp. 60 – 67, 2008.

[34] Gallardo-Valencia, R, Olivera, V.; Sim, S.E., "Are Use Cases Beneficial for Developers Using Agile Requirements?" *Workshop on Comp. Evaluation in Req Eng*. pp. 11-22, 2007.

[35] Ambler, S., "Agile Requirements Best Practices" http://www.agilemodeling.com/essays/ agileRequirementsBestPractices.htm 05/11/10

[36] Fowler, M., "Agile Requirements Best Practices", www.martinfowler.com/articles/ agileOffshore.html 05/11/10

CMMI después de la certificación

María Vanesa Cabral Freije
Practia Consulting
vcabral@practia.es
JuanJo Cukier
Practia Consulting
jcukier@practia.es

Resumen

El estudio presenta los resultados de una encuesta realizada a un conjunto de 40 empresas de España, Argentina y Chile que han llevado adelante un proyecto de mejora de proceso de desarrollo de software basado en el modelo CMMI-DEV del SEI. El estudio se ha realizó en empresas que se han acreditado en los últimos 3 años, alcanzando el Nivel 2 y Nivel 3 del modelo de madurez. El objetivo del estudio es conocer la percepción de los líderes de proyecto en cuanto al resultado obtenido tras la implantación de CMMI para el desarrollo. Es de resaltar que el 85% está satisfecho con el resultado del proyecto, notando mejoras en los tiempos de estimación de los proyectos, en la calidad de los mismos y en la mejora de costes. En ningún caso se percibe un ROI negativo del proyecto de mejora y se destaca que ha posibilitado la ejecución y mantenimiento de clientes. Cabe destacar que las ayudas locales de los gobiernos han posibilitado la ejecución de este tipo de proyectos.

Palabras clave: SEI, CMMI, mejora de procesos

CMMI after the appraisal

Abstract

The study presents the results of a survey performed to a group of 40 companies from Spain, Argentina and Chile that have achieved a project to improve software development process based on the model CMMI-DEV from the SEI. The study was conducted to companies that have been appraised in the past 3 years, reaching maturity levels 2 and 3 in CMMI-DEV. This study aimed at investigating the perceptions of project leaders as to the result obtained after the implementation of CMMI for Development. It is noteworthy that 85% are satisfied with the outcome of the project, noting improvements in project estimation, project quality and costs. In no case there was a perception of negative ROI of the project and it was stressed that it has enabled the implementation and maintenance of customers. Local government aids have allowed the execution of such projects.

Key words: SEI, CMMI, process improvement.

Cabral, V. y Cukier, J.J., "CMMI después de la certificación", REICIS, vol. 6, no.3, 2010, pp.76-83. Recibido: 8-11-2010; revisado: 14-11-2010; aceptado: 23-11-2010

1. Introducción

Los modelos CMMI (*Capability Maturity Model Integration*) son colecciones de las mejores áreas de proceso que ayudan a las organizaciones a mejorar sus procesos. Dichos modelos han sido desarrollados por grupos de trabajo provenientes de la industria, gobierno y del *Software Engineering Institute* (SEI).

El modelo CMMI para desarrollo (CMMI-DEV) provee un conjunto integrado de guías para el desarrollo de productos y servicios. Las áreas de proceso en el modelo se focalizan en aquellas actividades para el desarrollo de productos y servicios con calidad para lograr cumplir con las necesidades de los clientes y usuarios finales [1].

1.1. Objetivo y alcance del estudio

El propósito del estudio realizado es conocer la percepción del resultado obtenido luego de la implantación de CMMI.

La muestra incluyó un total 40 empresas de España, Chile y Argentina que han implantado CMMI en los últimos 3 años (ver figura 1). Para la realización de este estudio se empleó la técnica de encuesta on-line voluntaria y anónima, con una respuesta del 80%.

Las empresas, de diferentes mercados con predominio de factorías de software (ver figura 2), habían obtenido el Nivel 3 de Madurez de CMMI en un 20%, mientras que el 80% restante habían obtenido el Nivel 2 de Madurez de CMMI (ver figura 3). La dotación del personal alcanzada por el proyecto muestra que en 65% cuentan con menos de 100 personas (ver figura 4).

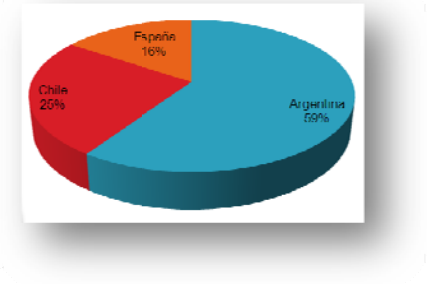


Figura 1 – Muestra por país

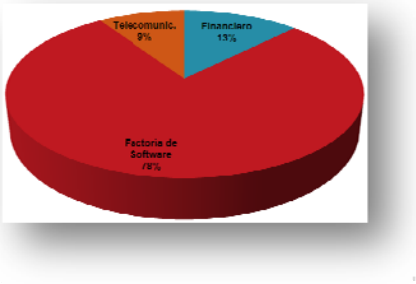


Figura 2 – Muestra por mercado

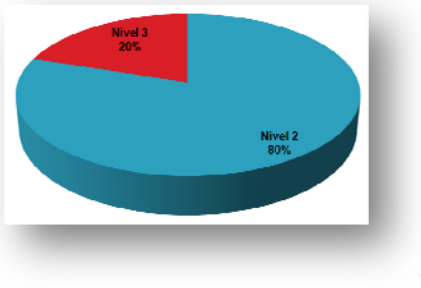


Figura 3 – Muestra por nivel de madurez

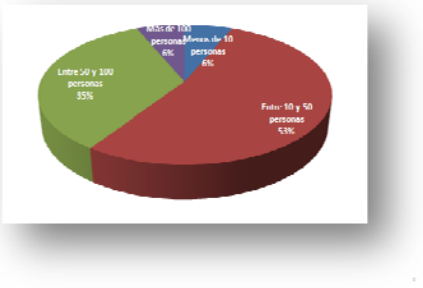


Figura 4 – Muestra por dotación de personal

2. Resultados

Se presentan los resultados del estudio, destacándose los beneficios más importantes. Ya en el 2006, en [2] se menciona que se puede generar un valor importante para el negocio con una implementación exitosa de CMMI.

Uno de los factores más importantes a evaluar en un proyecto de mejora de procesos es conocer la satisfacción con los cambios que se han implantado. El estudio recoge que se percibe una mejora de más del 92% en cuanto a los tiempos de los proyectos, notando las mejoras especialmente en la estimación de tiempos, aunque no en los tiempos de desarrollo del software (figura 6).

Por otro lado, los líderes han valorado una mejora del 100% en la gestión de los proyectos y disminución de los errores por haber implantado el área de proceso REQM (figura 7).

En cuanto a costes, se han percibido mejoras ya que se han estimado tareas invisibles y se dispone del coste completo de los proyectos (figura 8). Como desventaja menciona el incremento de costes debido a la implantación de la metodología.

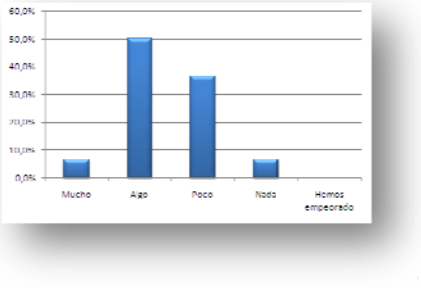


Figura 6 – Mejora de tiempos

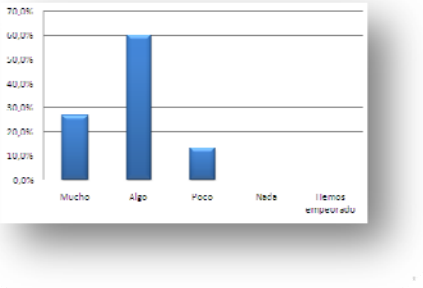


Figura 7 – Mejora de calidad

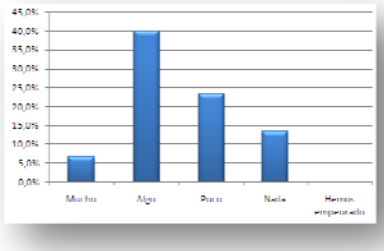


Figura 8 – Mejora de costes

Con respecto a la satisfacción del cliente, los líderes indican que han logrado una mejora importante, especialmente en las áreas de proceso relativas a Gestión de Requisitos y Planificación, remarcando que han obtenido mayor visibilidad de los proyectos de desarrollo, más formalidad por parte del área de desarrollo de software (cumplimiento de fechas, presentaciones, etc.) y orden (figura 9). La percepción de la satisfacción es superior al 80%.

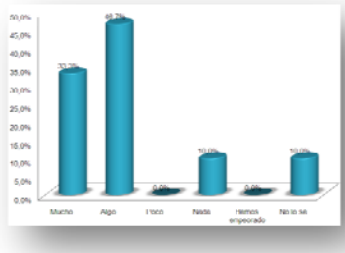


Figura 9. Satisfacción del cliente

Así mismo, más del 85% de los líderes de proyecto ha aplicado la metodología, con un decaimiento en algunos casos atribuido a la falta de tiempo. De todas maneras, más del 60% planea re-acreditar al vencimiento, incluso un 50% de ellos se ha planteado incrementar el nivel de acreditación, o bien, especialmente en organizaciones pequeñas, utilizar la representación continua del modelo para aquellas áreas de proceso que les interesa obtener mejoras significativas y que consideran que pueden ofrecer mayores beneficios a la organización. La principal desventaja que encuentran a lo hora de afrontar este tipo de proyectos es el coste de los mismos, incluyendo el coste de la acreditación. Más del 80% de los líderes han recomendado el modelo en dos sentidos: dentro de la organización y hacia otras empresas

Por otra parte, más del 93% considera que la organización percibe que el proceso de desarrollo de software responde a las necesidades del negocio (tiempo de respuesta, completitud de la solución, oportunidad) (figura 10).

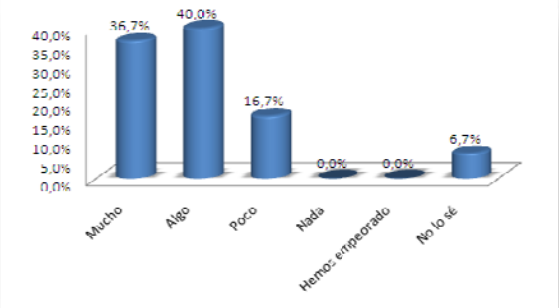


Figura 10 – Alineamiento del desarrollo con los objetivos del negocio

Realizando una comparativa entre los objetivos planteados al inicio del proyecto de mejora de procesos, en su gran mayoría han quedado satisfechos con el logro de los mismos (figura 11).

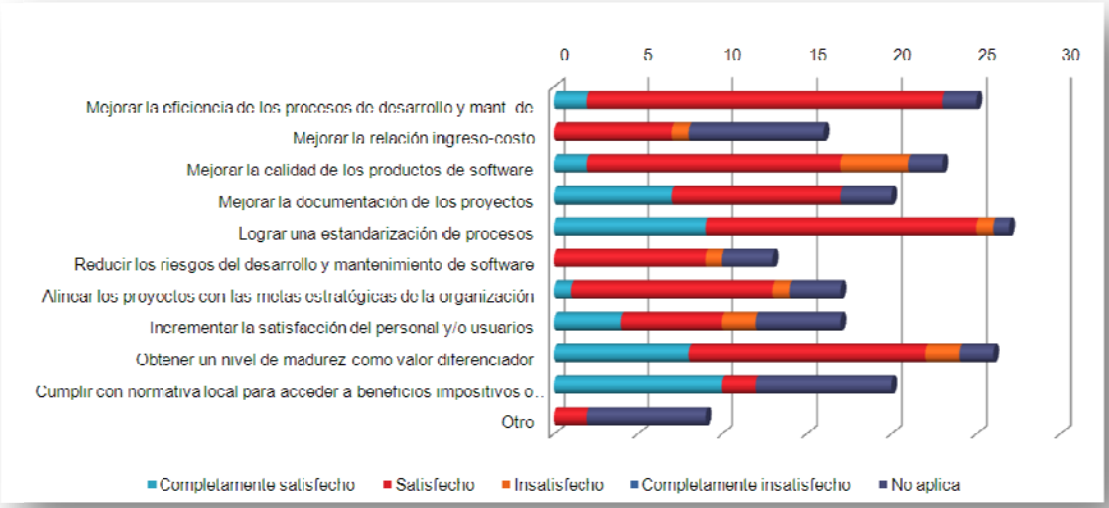


Figura 11. Cumplimiento de objetivos

El análisis de las diferentes áreas de proceso por cada nivel, aporta una clara visión de las principales falencias de los proyectos de desarrollo, y muestra el valor que el modelo puede aportar a las organizaciones.

El análisis muestra que las áreas de proceso mejor valoradas para el Nivel 2 son PP, REQM y PMC (figura 12) así como se agregan CM y VAL para el Nivel 3 (figura 13) del modelo CMMI.

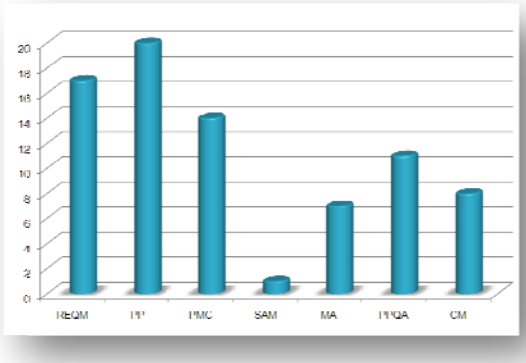


Figura 12. CMMI N2 – PAs que han aportado mayor valor

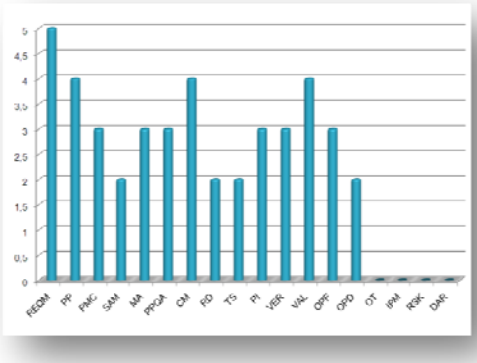


Figura 13. CMMI N3 – PAs que han aportado mayor valor

Los líderes consideran que realizando una comparación del proceso de desarrollo previo al proyecto de mejora de procesos basado en el modelo CMMI con respecto a la situación luego del proyecto, se ha generado una sobrecarga en el trabajo diario del personal (figura 14), pero que no supera en ningún caso el 10%.

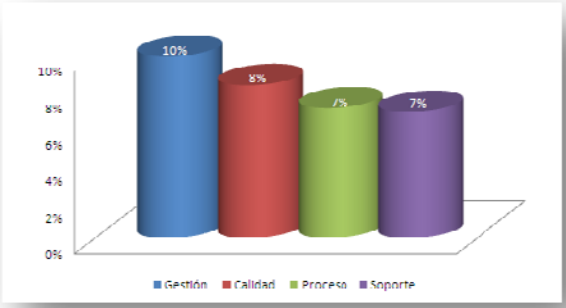


Figura 14. Sobrecarga de trabajo

Asimismo, el 82% de los líderes de proyecto considera que se puede aprovechar la documentación que se ha generado, principalmente aquella referida a estimaciones y métricas (figura 15).

Por otro lado, más del 65% considera que la documentación aporta gran valor para la organización, en el sentido de estandarizar y mantener el orden (figura 16).

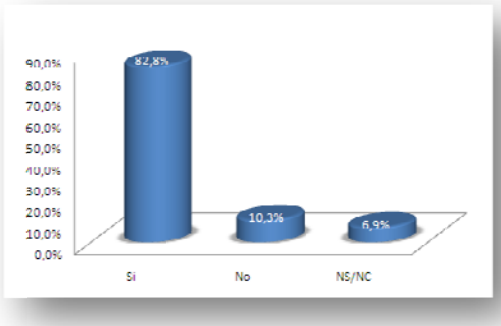


Figura 15. Aprovechamiento de la documentación

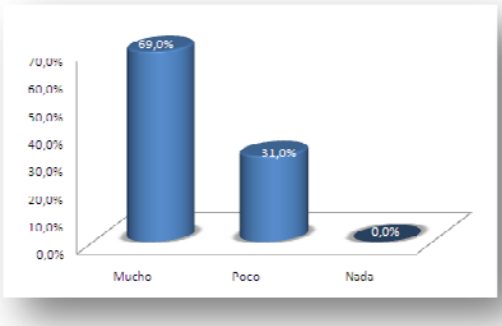


Figura 16. Valor que aporta la documentación

Los proyectos de mejora de procesos se han visto en muchos casos beneficiados con las ayudas provistas por los gobiernos locales, lo que ha impulsado que muchas organizaciones emprendieran proyectos de este tipo. Los líderes de proyecto consideran, en un 70%, que el ROI del proyecto es positivo aunque se trata de una percepción y no un valor que hayan podido medir. Cabe resaltar que no se percibe un ROI negativo (figura 17) en ningún caso y que es un posibilitador de nuevos negocios en el sentido que ha permitido mantener cuentas actuales así como participar en otros mercados (figura 18).

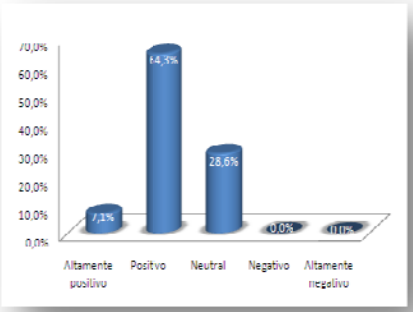


Figura 17 – ROI

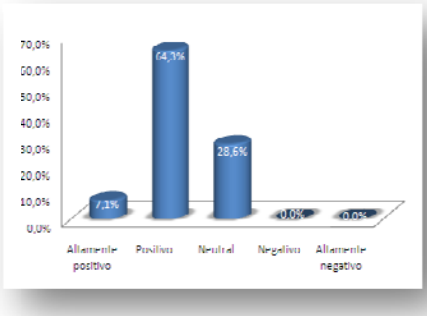


Figura 18 – Generador de nuevos negocios

3. Conclusiones

Llevar adelante un proyecto de mejora de procesos para alcanzar un nivel de madurez del modelo CMMI es un desafío que muchas empresas han emprendido con éxito.

Este estudio revela cuáles han sido los factores más importantes que han aportado valor para lograr la acreditación, y la percepción de las empresas en cuanto a los aportes más valorados del modelo.

El estudio revela que el 85% está satisfecho con el resultado del proyecto, notando mejoras significativas en tiempos (92%), calidad (100%) y costes (70%).

El 40% de las organizaciones han aprovechado las subvenciones otorgadas por el gobierno local para la financiación del proyecto. Los líderes no perciben ROI negativo. Por otro lado, se introduce mayor documentación aunque genera valor y ésta es reutilizada.

Finalmente, el análisis profundiza sobre las claves de éxito para alcanzar los objetivos de este tipo de proyectos, valorando principalmente el compromiso y apoyo de la alta dirección, el involucramiento de las personas que participan, el deseo de trabajar mejor y el tailoring.

Este estudio se complementará en un futuro con una base de empresas mayor, que amplíe las empresas actuales, para validar si con el paso del tiempo las áreas de proceso actuales se mantienen y continúan aportando valor a las empresas.

Referencias

[1] *CMMI for Development, Version 1.2*, Software Engineering Institute, Carnegie Mellon University, 2006

[2] García-Miller, Suzane y Turner, Richard, *CMMI Survival Guide: Just Enough Process Improvement*, 2006.

Comparando UML y OWL en la representación del conocimiento: correspondencia sintáctica

Susana M. Ramírez, Yisel Alonso, Violena Hernández, Arturo Cesar Arias y Dayana La Rosa
Universidad de las Ciencias Informáticas
{smramirez, yisel, violena, arturo, dlarosa}@uci.cu

Resumen

UML y OWL son lenguajes insignias de dos de los paradigmas más importantes que han emergido en los últimos tiempos para dar soporte al desarrollo de software. En la revisión de la literatura afín no se encuentra ampliamente documentada la relación entre ambos, a pesar del creciente interés en la utilización conjunta de UML y OWL. El propósito de este trabajo es proporcionar una comparación objetiva, con ejemplos concretos de la sintaxis de UML y OWL, que permita crear una base sólida para aprovechar las ventajas de cada uno y combinarlos en el proceso de desarrollo de software. Además se realiza una introducción al Ontology Definition Model para la utilización de la metodología, las herramientas y la tecnología UML como soporte para el desarrollo y el mantenimiento de ontologías.

Palabras clave: MOF, ODM, ontología, OWL, UML.

Comparing UML and OWL in knowledge representation: syntactic correspondence

Abstract

UML and OWL are insignia languages of two of the most important paradigms that have emerged in recent times to support software development. In the literature revision it is not widely studied the relation between UML and OWL, in spite of the growing interest on their combined use. The purpose of this paper is to provide an objective comparison with concrete examples of UML and OWL syntax, which would allow to create a solid base for making good use of their advantages, and to combine both languages in nowadays software development processes. Moreover, it makes an introduction to Ontology Definition Model for the use of UML's methodology, tools and technology, as a support for ontologies development and maintenance.

Key words: MOF, ODM, ontology, OWL, UML.

Ramírez,S.M,. Alonso,Y., Hernández,V., Arias, A.C.y La Rosa, D., "Comparando UML y OWL en la representación del conocimiento: correspondencia sintáctica", REICIS, vol. 6, no.3, 2010, pp.84-94. Recibido: 8-11-2010; revisado: 14-11-2010; aceptado: 19-11-2010

1. Introducción

En los últimos años dos importantes paradigmas han emergido para dar soporte al proceso de desarrollo de software. Por un lado se encuentra el paradigma que sitúa a los "modelos" *Model Driven Architecture* (MDA) en el centro del proceso de desarrollo con el *Unified Modeling Language* (UML) [1] del *Object Management Group* (OMG) como lenguaje insignia; y por el otro el paradigma de la ingeniería ontológica que ubica a las "ontologías" como la base del proceso. Esta ciencia, que evolucionó de la inteligencia artificial, cuenta con el *Web Ontology Language* (OWL) [2] de la *World Wide Web Consortium* (W3C) como su lenguaje insignia.

Dado que ambos paradigmas fueron desarrollados desde visiones diferentes, hasta hace muy poco se mantuvieron relativamente separados, cada comunidad evolucionando el lenguaje de acuerdo a sus propias necesidades. Sin embargo, desde que la representación del conocimiento en el desarrollo de software ha cobrado importancia y ambos lenguajes ofrecen la posibilidad de hacerlo en alguna medida, ha crecido el interés tanto de la industria como de la academia. Ambos intentan comprender cómo estos paradigmas se relacionan y cuál ofrece mejores capacidades para expresar el conocimiento bajo ciertas circunstancias, y cómo pueden ser usados de conjunto.

Para poder llevar esto a cabo, es importante establecer una comparación en cuanto a lo que ambas tecnologías ofrecen. No es posible tomar una buena decisión sobre cuál tecnología emplear, cuán efectivo sería usar los espacios de ambas tecnologías de conjunto y cómo integrarlas, sin conocer en qué se asemejan y difieren.

2. Lenguajes UML y OWL. Comparación.

UML se ha convertido en el estándar de facto de la industria. Por su parte OWL es un lenguaje concebido específicamente para la Web Semántica, que sobresale por encima de sus similares. [3]. Sin embargo, hasta el momento no se han explotado las potencialidades de su uso combinado. En esta sección se presentan las principales características de cada uno, y una comparación técnica, fundamentalmente de la sintaxis de ambos lenguajes.

2.1. Unified Modeling Language

UML es definido por sus creadores en [4] como un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

Las versiones iniciales de UML (UML 1) se originaron con los tres principales métodos orientado a objetos (Booch, OMT[3] , y OOSE[4]) del momento. La última revisión de UML (UML 2.0) [5] [6], ha sido realzada con definiciones significativamente más precisas, una estructura del lenguaje más modular, y una altamente mejorada capacidad para el modelado de sistemas de gran escala.

UML capta la información sobre la estructura estática y el comportamiento dinámico de un sistema, no obstante carece de una semántica rigurosa, que permita utilizar razonadores automáticos sobre los modelos. Para realizar la comparación, este trabajo se enfocará en la representación de la información de estructura estática a través de los diagramas de clases.

UML se encuentra conceptualmente embebido dentro de la arquitectura de cuatro niveles de OMG, y es considerado como una instancia del *Meta Object Facility* (MOF) [7]. El estándar MOF define diversos metamodelos, abstrayendo la forma y la estructura que describen. Define los elementos esenciales: sintaxis y estructuras de metamodelos que se utilizan para construir modelos de sistemas. Varias son las tecnologías estandarizadas por OMG que usan MOF y sus tecnologías derivadas para el intercambio y manipulación de metadatos. Dentro de las más recientes iniciativas se encuentra el estándar *Ontology Definition Metamodel* (ODM) para la transformación entre metamodelos MOF. ODM será abordado con posterioridad en este trabajo.

2.2. Web Ontology Language

OWL es un lenguaje de ontologías para la Web Semántica con significado formalmente definido, desarrollado por el *W3C Web Ontology Working Group*. Es desarrollado como un lenguaje extensión del *Resource Description Framework* (RDF) [8] y es derivado del lenguaje *DAML+OIL* [9]. Está diseñado para facilitar el desarrollo de ontologías y el uso compartido por la Web, con el objetivo de hacer más accesibles los contenidos de la Web para las máquinas.

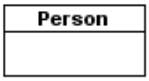
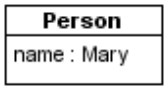
Las ontologías en OWL proveen clases, propiedades, individuos y valores de datos, y son almacenados como documentos Semánticos de la Web. La versión actual de OWL define tres sublenguajes que son presentados según su capacidad expresiva. [10]: OWL –

[3] Object Modeling Technique
[4] Object Oriented Software Engineering

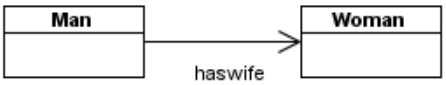
Lite, OWL – DL y OWL-FULL. Para la comparación se utilizará OWL – FULL por ser la versión más general del lenguaje con la que se logra mayor expresividad. Será representado usando *Functional Syntax*, debido a que es la sintaxis más simple para presentar la estructura formal de una ontología.

2.3. Comparación entre UML y OWL

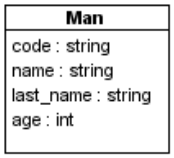
Es necesario destacar que ambos lenguajes fueron construidos bajo propósitos diferentes: OWL para la representación del conocimiento y UML para soportar el desarrollo de software. Sin embargo, ambos permiten la representación del conocimiento y emplean para esto objetos y relaciones entre ellos. A pesar de la similitud en cuanto a su fin, estos difieren en cuanto a cómo el conocimiento es entendido y expresado. Con UML se modelan las restricciones que satisfacen al conjunto de estados permitidos de un sistema. Con OWL se representa el conocimiento y este se emplea para inferir nuevo conocimiento. [11]. A continuación se muestra una comparación (no completa pero si representativa) de las características comunes de ambos lenguajes.

<table>
<tr><th>Elementos UML</th><th>Elementos OWL</th></tr>
<tr><td>class</td><td>class</td></tr>
<tr><td colspan="2">Ambos lenguajes se basan en clases. Una clase en OWL es un set de cero o más instancias. Una clase en UML es una construcción más general, uno de sus usos es como un set de instancias. En OWL hay una clase universal Thing cuya extensión es todos los individuos en un modelo dado, y todas las clases son subclases de Thing.</td></tr>
<tr><td> 
</td><td>Class(: Person)</td></tr>
<tr><td>instance</td><td>individual</td></tr>
<tr><td colspan="2">En OWL la construcción individual es similar a la construcción instance en UML. Sin embargo OWL FULL permite diferentes formas de definición de una clase (como clase y como instancia). En el primer ejemplo se muestra como la clase Person es usado como clase mientras que en el segundo se utiliza como individuo actuando SocialRole como metaclase de Person.</td></tr>
<tr><td> 
</td><td>Individual(:Person :Mary)
Individual(:SocialRole :Person)</td></tr>
<tr><td>binary association, ownedAttribute</td><td>property</td></tr>
<tr><td colspan="2">Las relaciones entre las clases en OWL son llamadas property y representadas por ObjectProperty y DataProperty.
En el ejemplo la clase Man tiene una relación con la clase Woman llamada haswife, que es representada en el modelo UML como la asociación haswife y en OWL como una ObjectProperty del mismo nombre.</td></tr>
</table>

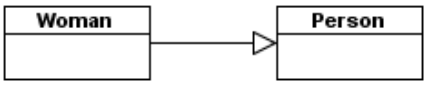
UML tiene la opción para las asociaciones de distinguir la navegabilidad (el final de relación), esta puede ser *navigable* o *non-navigable*. En contraparte a esto las propiedades OWL han distinguido finales designados como: dominio (*domain*) y rango (*range*). En el ejemplo se muestran el dominio y rango de la *ObjectProperty haswife*

	*ObjectProperty(: haswife* domain *(: Man)* *range (: Woman))*

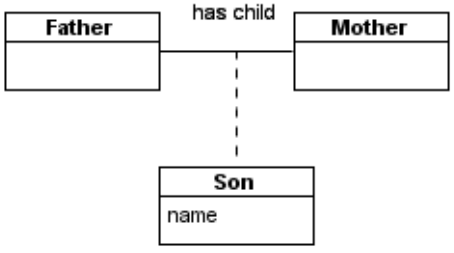
Los *DataProperty* surgen de la asociación llamada *ownedAttribute* entre *Class* y *Property* y son traducidos como *properties* cuyo dominio es una Clase y el rango es el tipo de la *Property*. En el ejemplo se presenta la definición en OWL para el *ownedAttribute* de la clase *Man* : *name*

	*DatatypeProperty (: name* *domain (:name :Person)* *range (:name xsd:string))*
generalization	**subclass subproperty**

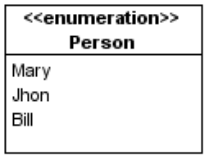
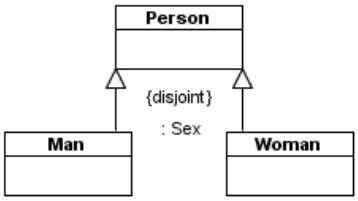
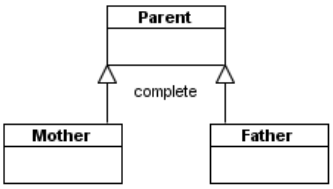
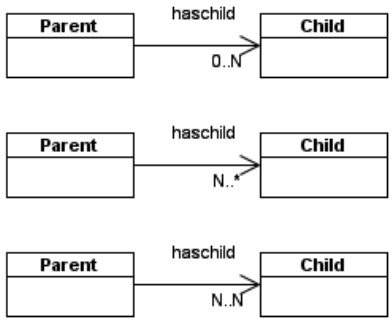
Ambos lenguajes soportan la relación *subclass*. En OWL con *subClassOf* y en UML se presenta como la relación de *generalization* para expresar la inclusión de una clase.

	*subClassOf(:Woman :Person)*
N-ary association, association class	**class, property**

Una asociación en UML puede ser *N-ary*. Una asociación también puede ser una clase (*association class*) que puede participar en más relaciones. En el ejemplo la traducción descrita a OWL no es normativa debido a que no existe una construcción específica, por definición una relación se define entre pares de conceptos. La solución que se presenta es la utilización del patrón de diseño llamado *reification* que consiste en la creación de una nueva clase que conceptualiza la relación e involucra los conceptos que esta envuelve. Se crean n nuevas relaciones funcionales, una por cada participante en la asociación.

	*Class(: Son)* *ObjectProperty(: Father_Son* domain *(: Son)* r*ange (: Father))* *ObjectProperty(: Mother_Son* domain *(: Son)* r*ange (: Mother))*
enumeration	**oneOf**

Ambos soportan una extensión fija definida para una clase, aunque en UML un enumerador es considerado un tipo de datos antes que una clase.

<table>
<tr><td> 
</td><td>Class(: Person
OneOf(:Mary :Jhon :Bill))</td></tr>
<tr><td>disjoint, cover</td><td>disjointWith, unionOf</td></tr>
<tr><td colspan="2">Ambos lenguajes permiten subclases de clases ser declaradas disjuntas. En OWL se utiliza la construcción disjointWith. Además es posible que para una colección de subclases sea declarado que el cubrimiento es completo, es decir cada instancia de la superclase es una instancia de al menos una de las subclases. La construcción correspondiente en OWL para declarar que una superclase es la unión de sus subclases es unionOf. En UML se representa a partir de la definición de las restricciones correspondientes.</td></tr>
<tr><td> 
</td><td>disjointWith (:Woman :Man)</td></tr>
<tr><td> 
</td><td>Class(:Parent
unionOf(:Mother :Father))</td></tr>
<tr><td>multiplicity</td><td>minCardinality, maxCardinality, cardinality</td></tr>
<tr><td colspan="2">En OWL, una restricción de una propiedad aplicada a una clase puede imponer una restricción de cardinalidad dando el número mínimo (minCardinality), máximo (maxCardinality), o exacto (cardinality) de instancias que pueden participar de una relación. En UML una asociación puede tener cardinalidades mínimas y máximas (multiplicity). Se muestran tres ejemplos de las tres restricciones de cardinalidad.</td></tr>
<tr><td> 
</td><td>maxCardinality(N :hasChild :Parent)

minCardinality(N :hasChild :Parent)

cardinality(N :hasChild :Parent)</td></tr>
<tr><td colspan="2">Además, una propiedad OWL puede ser globalmente declarada como funcional (functionalProperty) o inversa funcional (inverseFunctional). Una propiedad funcional tiene una cardinalidad máxima de 1 sobre su rango, mientras una propiedad funcional inversa tiene una cardinalidad máxima de 1 sobre su dominio. En el ejemplo la ObjectProperty haswife es de los tipos.</td></tr>
</table>

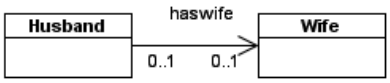
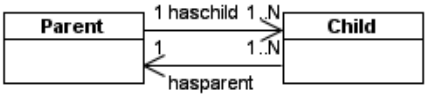
	*FunctionalProperty(:haswife)* *InverseFunctionalProperty(:haswife)*
Si una asociación binaria UML tiene una multiplicidad sobre ambos extremos, entonces la propiedad correspondiente OWL será un par inverso.	
	*inverseOf (:hasparent :haschild)* *maxCardinality(N :haschild :Parent)* *maxCardinality(1 :hasparent :Child)*
package	**ontology**
Ambos soportan el concepto de *namespace* como una construcción empaquetadora, definida como *Model* que es una subclase de la primitiva *Package* en UML y *Ontology* en OWL.	

Tabla 1. Comparación. Características comunes OWL – UML.

Para la comparación se han tenido en cuenta todas las características de UML que de alguna manera tienen algún equivalente en OWL. Sin embargo, existen también otras características sin equivalencia. Estas son las siguientes: *derived, abstract classifier, operations, interface classes, active classes, ports, connectors y complex objects.* Por su parte también existen características de OWL con ningún equivalente en UML: *allValuesFrom, someValuesFrom, SymmetricProperty, TransitiveProperty y complementOf.*

A pesar de las diferencias mostradas, la metodología, las herramientas y la tecnología UML parece ser un enfoque factible para soportar el desarrollo y el mantenimiento de ontologías. Recientemente, se ha comenzado a perfilar la idea del uso directo de UML como un lenguaje de ontologías, considerando la definición de varios estereotipos adicionales, que permitan un mapeo más detallado de UML a las primitivas brindadas por la lógica descriptiva [3]. Consecuentemente, OMG realizó la petición de propuestas para una definición de Ontology Definition Metamodel [12].

3. Ontology Definition Metamodel

ODM ha sido concebido dentro del MOF como cualquier otro metamodelo de OMG. La especificación define una familia de metamodelos independientes, perfiles relacionados, y mapeos entre los metamodelos, correspondiendo a varios estándares internacionales para ontologías, así como la capacidad de soportar paradigmas convencionales de modelado para captar conocimiento conceptual, como UML y *Entity-Relationship Diagram*.

Los metamodelos en ODM son tratados de igual manera, aun cuando son independientes entre sí. No es necesario entender o estar al tanto de los demás para alcanzar el entendimiento total de uno específico. La única excepción es el metamodelo para OWL, que extiende del metamodelo para RDF, tanto como el lenguaje OWL extiende del RDF.

Sin embargo, en un proyecto de desarrollo de una ontología debería ser necesario el uso de varios metamodelos. Teniendo en cuenta el amplio uso de UML, comúnmente el desarrollador se encontrará ante la necesidad de reutilizar artefactos ya existentes, o de aprovechar la experiencia ya acumulada con herramientas asociadas a estas, para realizar al menos una primera aproximación de alineamiento con el modelo OWL. ODM, por tanto, necesita proveer facilidades para establecer relaciones entre instancias de sus metamodelos, incluyendo UML. Existen dos formas para lograr esto: perfiles UML y mapeos.

3.1 Perfiles UML y Mapeo

Un perfil UML de la perspectiva ODM, tiene como objetivo proveer un puente entre UML y los lenguajes de representación del conocimiento sobre una bien fundada, base semántica. Los perfiles facilitan la implementación usando la notación común en herramientas existentes UML. Igualmente proveen a los usuarios capacidades para utilizar UML como la base para el desarrollo de ontologías para un lenguaje especifico de representación del conocimiento como OWL. Por lo tanto se necesita especificar los mapeos de un metamodelo a otro. Para el trabajo con múltiples metamodelos se requiere un elemento de traducción del modelo por cada elemento de los modelos a mapear.

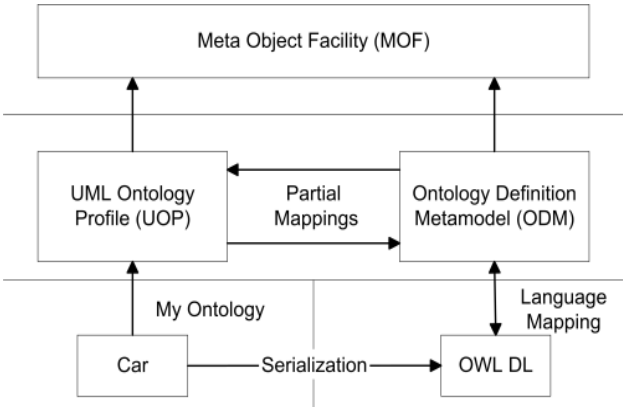


Figura 2. Mapeo a OWL DL [13]

ODM, además de definir una notación visual a través de los perfiles, debe proveer los mapeos parciales en ambas direcciones entre el metamodelo y el perfil definido. Del ODM

se debe poder generar una representación de una ontología en un lenguaje como OWL DL, una serialización usando el XML Metadata Interchange (XMI) y una sintaxis de cambio. Este formato XMI permite intercambiar un metamodelo ODM entre herramientas. La figura anterior muestra este proceso.

Tres perfiles UML han sido desarrollados para el uso con ODM: para RDF, OWL, y Topic Maps (TM). Estos permiten el uso de notación UML (y las herramientas) para el modelado de ontologías y facilitan la generación de las correspondientes descripciones de ontologías en RDF, OWL, y TM, respectivamente. Con estos perfiles se reutilizan construcciones UML que tienen la misma semántica que dichos lenguajes, y, cuando no es posible encontrar construcciones exactamente equivalentes desde el punto de vista semántico se utilizan entonces construcciones que son consistentes y cercanas. Una descripción detallada de cada perfil se encuentra en [12].

ODM encierra los fundamentos de un enfoque para el desarrollo y despliegue de sistemas que ha dado en llamarse *Ontology Driven Architecure* (ODA) [14].

4. Conclusiones

En este trabajo, se ha presentado una comparación detallada de la sintaxis de UML y OWL, que permitan valorar las capacidades de expresión del conocimiento en determinadas circunstancias de ambos lenguajes. Esta comparación puede ser interpretada desde dos perspectivas dependiendo de la intención del lector, resaltando las marcadas diferencias existentes o las innegables similitudes entre ellos.

Las similitudes se evidencian a partir de que ambos lenguajes permiten la representación del conocimiento y emplean para esto objetos y relaciones entre ellos. Adicionalmente, la sintaxis abstracta de ambos presenta un alto grado de similaridad, manifestada en la correspondencia existente en la mayor parte de sus constructores.

Por otra parte, se diferencian en que fueron concebidos bajo diferentes propósitos: OWL para la representación del conocimiento y UML para soportar el desarrollo de software, lo que se constata en diferentes formas. Con UML se pretende modelar las restricciones que satisfacen al conjunto de estados permitidos de un sistema, mientras que con OWL se representa el conocimiento y este se emplea para inferir nuevo conocimiento.

Como resultado, es imposible la traducción de ontologías OWL a modelos UML y viceversa, sin la pérdida o corrupción de la información.

Parte de la solución a dicha problemática se ofrece en la propuesta de integración que ha realizado el OMG con una de sus más recientes iniciativas el *Ontology Definition Metamodel*, dado que ambos lenguajes y las tecnologías relacionadas ocupan un puesto central en la especificación del mismo. Con este estándar se trata de utilizar la notación UML y las tecnologías asociadas a este para el modelado de ontologías, de manera que se aprovechen las ventajas de ambas tecnologías y combinen durante el proceso de desarrollo. Una introducción de las características y tecnologías envueltas en ODM es provista en este trabajo.

Referencias

[1] Object Management Group, *Unified Modeling Language (OMG UML) v1.3*, OMG, 2000.

[2] World Wide Web Consortium, *OWL Web Ontology Language Semantics and Abstract Syntax*, W3C, 2004.

[3] García Noguera, M., *Modelado y Análisis de Sistemas CSCW siguiendo un enfoque de Ingeniería dirigida por Ontologías.* Tesis Doctoral. Universidad de Granada, España, 2009.

[4] Booch, G., Jacobson, I. y Rumbaugh, J., *El Lenguaje Unificado de Modelado. Manual de Referencia,* Addison Wesley, 2000.

[5] Object Management Group, *Unified Modeling Language (OMG UML), Infrastructure Specification, v2.3*, OMG, 2010.

[6] Object Management Group, *Unified Modeling Language (OMG UML), Superstructure Specification v2.3*, OMG, 2010.

[7] Object Management Group, *Meta Object Facility Core Specification v 2.0*, OMG, 2006.

[8] World Wide Web Consortium, *Recommendation. Resource Description Framework (RDF): Concepts and Abstract Syntax*, W3C, 2004.

[9] World Wide Web Consortium, *Note. DAML+OIL Reference Description*, W3C, 2001.

[10] Horrocks, I., Patel-Schneider, P. y van Harmelen, F., "From SHIQ and RDF to OWL: The Making of a Web Ontology Language", *Journal of Web Semantics*, vol. 1, nº 1, pp. 7-26, 2003.

[11] Kiko, K. y Atkinson, C., "A Detailed Comparison of UML and OWL", *University of Mannheim*, Technical Report TR-2008-004, Department for Mathematics and Computer Science, University of Mannheim, 2008.2008.

[12] Object Management Group, *Ontology Definition Metamodel v1.0*, OMG, 2009.

[13] Brockmans, S, Volz, A., Eberhart, A. y Loeffler, P., "Visual Modeling of OWL DL Ontologies Using UML", En: van Harmelen, F., Mcllraith, S. A. y Plexousakis, D. (eds.) *The Semantic Web- ISWC. Hiroshima (Japón), Noviembre*, pp. 198-213, 2004.

[14] World Wide Web Consortium, *Working Draft. Ontology Driven Architectures and Potential Uses of the Semantic Web in Systems and Software Engineering. Editors' Draft,* W3C, 2006.