Enl@ce: Revista Venezolana de Información, Tecnología y Conocimiento
ISSN: 1690-7515
Depósito legal pp 200402ZU1624
Año 7: No. 3, Septiembre-Diciembre 2010, pp. 11-26

Cómo citar el artículo (Normas APA):
Omaña, M. y Cadenas, J. (2010). Manufactura Esbelta: una contribución para el desarrollo de software con calidad. *Enl@ce Revista Venezolana de Información, Tecnología y Conocimiento*, 7 (3), 11-26

Manufactura Esbelta: una contribución para el desarrollo de software con calidad

Macringer Omaña[1]
José Cadenas[2]

Resumen

Esta investigación es una contribución a la mejora de la calidad del software que tiene como premisa que los desarrollos de software efectuados a la fecha no satisfacen las expectativas de tiempo de desarrollo, fiabilidad, *mantenibilidad*, portabilidad y calidad. Se trata de una investigación de campo, apoyada en una revisión documental de tipo no experimental, descriptiva y transeccional con el fin de evaluar la versión 4 del SQLfi. Para ello, se empleó el modelo sistémico de calidad del software (MOSCA) con una población de 26 personas miembros del equipo de investigación, de los cuales se tomó una muestra intencional de 11, los cuales evaluaron el producto SQLfi versión 4. Se obtuvo un nivel sistémico de calidad nulo, y como conclusión se propone la adopción de un modelo de desarrollo para la construcción de software de calidad basado en estándares establecidos de manufactura esbelta (*Lean Manufacturing*), complementado con los aportes derivados de la evaluación a través de MOSCA y la experiencia de los investigadores. La contribución corresponde a una mejora de la calidad sistémica de desarrollo software que permite obtener productos en forma ágil, a un costo razonable y con los recursos presupuestados. Se recomienda evaluar si hay aumento en el nivel de madurez en los procesos de desarrollo de un grupo en un entorno científico y académico.

Palabras clave: calidad del software, modelo sistémico de calidad del software (MOSCA), manufactura esbelta, desarrollo de software esbelto

Recibido: 02-10-10 Aceptado: 18-11-10

[1] Ingeniero Industrial. Profesora Universidad Simón Bolívar, La Guaira, Venezuela. Jefe de sección de Organización empresarial y transporte. Profesora del Consejo Asesor del Departamento de tecnología de servicios.
Correo electrónico: macringer@usb.ve

[2] Ingeniero en Computación. Profesor Universidad Simón Bolívar, Venezuela. Departamento de Computación.
Correo electrónico: jtcadenas@usb.ve

Lean Manufacturing: A Contribution for Software Development with Quality

Abstract

This research is a contribution to improving the quality of software that is premised on software developments made to date, which do not meet the expectations of development time, reliability, maintainability, portability and quality. This is a field research, supported by a documentary review, non-experimental, descriptive and transversal. We used the Systemic Model of Software Quality (MOSCA) with a population of 26 persons, was taken a intentional sample of 11 team members, which evaluated the product SQLfi version 4. We obtain a systemic quality level "null", is therefore proposed to adopt a development model based on standards established in lean manufacturing, complemented with the assistance obtained from the assessment through MOSCA and experience of researchers. The contribution is in obtaining systemic quality in software development to obtain products in agile form, at reasonable cost with the budgeted resources. We recommend evaluating if there is a higher level of maturity in the development process of a project group in a scientific academic environment.

Key words: Software Quality, Systemic Model of Software Quality (MOSCA), Lean Manufacturing, Lean Software Development

Introducción

En la sociedad de la información y el conocimiento, el software es considerado un factor crítico de éxito, por ello, las empresas continuamente utilizan herramientas de gestión del conocimiento para ser más eficientes, al igual que los gobiernos mejoran su presencia en Internet para prestar servicios a los ciudadanos; por su parte los usuarios emplean las herramientas para sus relaciones interpersonales a través de redes sociales; en ello, la Web 2.0 ha desarrollado una nueva cultura virtual donde el eje fundamental es la información.

La calidad es una especialidad de Ingeniería del Software que ha sido objeto de mucho interés debido a su importancia en la sociedad actual, entre sus finalidades está mejorar el desarrollo de productos sin que esto signifique un incremento en el uso del tiempo, ni un mayor costo. Además, debido al uso generalizado y la confianza de las personas en los sistemas informáticos, se hace necesario garantizar que cumplan con las expectativas de calidad y confiabilidad.

En el año 1991, el Instituto de Ingeniería de Software (SEI) propone el Modelo Integrado de Capacidad de Madurez (*CMMI: Capability Maturity Model Integrated*) para proporcionar a las organizaciones de desarrollo de software una referencia de cómo controlar sus procesos de desarrollo y mantenimiento con calidad, además de obtener una cultura de ingeniería de

software en forma incremental de acuerdo a lo expresado por Chrissis, Konrad y Shrum (2007). CMMI proporciona cinco niveles de madurez de una organización en el desarrollo de procesos de software con el fin de que cada organización se evalúe usando como modelo de referencia estos niveles, además de que se sepan los pasos a seguir para conseguir llegar al siguiente nivel. Sólo que esto es propuesto para empresas desarrolladoras de software y no grupos de desarrollo dentro de un entorno científico y académico.

Por otro lado, en Venezuela se ha observado un crecimiento en la demanda del software, además del interés que existe en el desarrollo con calidad como lo demuestran iniciativas de varias universidades y entes venezolanos en pro de fortalecer el desarrollo del software, como lo es el Centro de Excelencia en Ingeniería del Software, CEISOFT (2010). En este proceso hay que destacar la obligatoriedad del uso del software libre a nivel de organismos del Estado (decreto No. 3390 del 23-12-2004, gaceta oficial No. 38095), además de políticas para el impulso del mismo tal como lo muestra la Fundación CENDITEL (2010). De acuerdo a lo estudiado por investigadores del Laboratorio de Sistemas de Información de la Universidad Simón Bolívar, LISI (2010), esta demanda ha sido cubierta con una gama de productos que no satisfacen estándares de calidad; además, las organizaciones desarrolladoras carecen de un nivel de madurez adecuado.

Actualmente se desarrolla el proyecto "Creación y Aplicación de Manejadores de Base de Datos Difusas" liderado por investigadores del Grupo de Base de Datos de la Universidad Simón Bolívar (USB) en conjunto con profesores de la Universidad de Carabobo (UC) y del Instituto Universitario de Tecnología Federico Rivero Palacios (IUTFRP), con el apoyo del Fondo Nacional de Ciencia y Tecnología, tal como se muestra en FONACIT (2010). Este proyecto tiene como misión el desarrollo de software en el área de base de datos difusas entre los que destaca el producto **SQLfi** versión 4 (V4), el cual es un sistema web de consultas difusas a bases de datos relacionales descrito por Goncalves y Tineo (2008). Este grupo de investigadores, consciente de la importancia de producir software de calidad, se planteó la utilización del **M**odelo **S**istémico de **C**alidad del **S**oftware (MOSCA), formulado por Mendoza, Pérez, y Grimán (2005); para establecer el nivel sistémico de calidad del producto.

La metodología utilizada fue un estudio de campo, apoyado con una revisión documental, en una investigación de tipo no experimental, descriptiva y *transeccióna*l. Se empleó MOSCA tomando en consideración una población de 26 personas que intervinieron en el proyecto, seleccionando una muestra intencional de 11 desarrolladores y líderes que evaluaron el producto SQLfi V4. Luego de verificar que el nivel de calidad resultó ser nulo, se planteó como objetivo elaborar una propuesta para la construcción de software de calidad en el grupo de proyecto utilizando un enfoque de manufactura esbelta (*Lean Manufacturing*), con la contribución de los aportes del resultado de la evaluación utilizando MOSCA y la experiencia de los investigadores. En términos se busca que el grupo adopte este modelo para futuros proyectos y se evalúe si hay un aumento en el nivel de madurez

en los procesos de desarrollo dentro de un entorno científico y académico.

A continuación en el desarrollo del trabajo se presenta el planteamiento del problema, la metodología utilizada, un breve marco teórico que incluye: Modelo sistémico de calidad (MOSCA), manufactura esbelta y desarrollo de software esbelto; luego se presentan los resultados obtenidos al utilizar las métricas de MOSCA, la propuesta para la construcción de software de calidad basado en manufactura esbelta; y, finalmente las conclusiones y recomendaciones de la investigación.

Evaluación de la calidad del software SQLfi V4

El SQLfi V4 es un sistema web de consultas difusas a bases de datos relacionales que emplea el lenguaje de programación Java con un estilo arquitectónico de capas y componentes, con la intención de tener una interfaz de programación de aplicaciones (API), cuyo propósito es proporcionar un conjunto de funciones generales que los programadores pueden usar evitando así el trabajo de programar todo desde el principio.

Al tratar de utilizar el SQLfi V4 dentro de otro entorno operativo, por ejemplo servidores de la Universidad de Carabobo en lugar de los de la USB donde están instalados, se enfrentaron con muchas dificultades, de lo cual se deducen problemas de portabilidad del software; aunque una de las premisas, de acuerdo al lenguaje de programación utilizado y el estilo arquitectónico del mismo, era su capacidad de ser portable.

Por otra parte, al utilizar el software con otro sistema gestor de base de datos (*DBMS*), por ejemplo MySQL en lugar de Oracle, se han detectado diversas fallas en la implementación, debido a errores de código e interpretación de la teoría en que se basa, levantándose dudas acerca del cumplimiento funcional del producto y su flexibilidad para adaptarlo a otros DBMS.

También se pudo corroborar la carencia de una documentación adecuada, se cuenta con la publicación de Crespo (2006) quien hizo una descripción del SQLfi V4 con fines académicos, un manual orientado al usuario y presenta el código utilizado; lo cual ha sido de poca ayuda para personas que quieren continuar el desarrollo del software para mejorarlo, por lo que el mantenimiento del producto está en dudas.

La experiencia hasta ahora es que la implementación de diferentes versiones del producto fueron realizadas por estudiantes que luego de cumplir sus metas académicas, se retiraron a trabajar fuera del entorno universitario. En este sentido los conocimientos adquiridos por este personal durante el desarrollo del proyecto son considerados de poco alcance, porque no pudieron ser transferidos a otros miembros del equipo en forma satisfactoria.

También se corroboró el incumplimiento de los tiempos estimados en el proyecto, extendiéndose el desarrollo mucho más de lo planificado, los productos finales presentaron gran cantidad de defectos aún no resueltos, hubo alta rotación de personal y se perdió mucho tiempo tratando de comprender lo hecho por otras personas.

Otro hallazgo significativo fue que los programadores responsables de seguir con el desarrollo no estaban acostumbrados a leer códigos de software elaborado por otras personas, esto produjo una variabilidad en el estilo de implementación debido a la falta de control de versiones y estándares de codificación.

Todo lo anterior repercute en la calidad del software, toda vez que se deben asignar grupos de personas para que continuamente revisen los desarrollos y rehagan el trabajo, repitiendo actividades que debieron efectuarse con anterioridad; modificando y adaptando el código de programación de acuerdo a las directrices de los líderes del grupo.

Los investigadores del presente estudio infieren que la posible causa de esta problemática de deficiencias en la productividad observada en el desarrollo de software y la baja calidad del producto resultante, de acuerdo al análisis efectuado y a la propia experiencia como investigadores, es la carencia de un modelo que sistematice el proceso de desarrollo a seguir por las personas involucradas en el proyecto, en un entorno científico y académico. Esta situación coloca en evidencia la falta de habilidad del grupo de proyecto de gestionar sus procesos, lo cual resulta en repetición de trabajos y una excesiva utilización de los recursos planificados.

Como consecuencia de la extensión de los tiempos de desarrollo, la productividad del grupo de investigación se vio afectada considerablemente, incidiendo negativamente en la posibilidad de obtener financiamientos de los entes interesados para financiar nuevos proyectos del grupo.

Debido la problemática antes descrita se planteó hacer esta investigación para evaluar la calidad del software desarrollado por el grupo de proyecto, específicamente del SQLFfi V4, para ello se utilizó el Modelo Sistémico de Calidad del Software (MOSCA) elaborado por el Laboratorio de Investigación en Sistemas de Información (LISI) de la USB, el cual fue formulado por Mendoza, Pérez, y Grimán (2005); este modelo ya ha sido probado, validada sus métricas y algoritmo de evaluación en diversas investigaciones que pueden revisarse en LISI (2010). Luego se plantea una propuesta para la construcción de software de calidad basado en manufactura esbelta.

Desde el punto de vista metodológico se planteó una investigación de campo, no experimental transeccional, apoyada en una revisión documental. La población objeto de estudio fueron veintiséis personas (26) que trabajaron en el proyecto "Creación y aplicación de Sistemas manejadores de Bases de Datos Difusas", de los cuales se tomó una muestra intencional de once (11) líderes y desarrolladores, ya que se escogieron los que tuvieran conocimiento del producto SQLfi V4. En cuanto a la originalidad del estudio esta se refleja fundamentalmente en el enfoque que los investigadores generaron a partir de los resultados obtenidos, así como en la formulación de una propuesta para el grupo de desarrollo, basada en los postulados del desarrollo de software esbelto, sus conclusiones y recomendaciones, tomándose en cuenta la experiencia propia y el aporte de otros especialistas en el área.

En el transcurrir de la investigación se utilizaron una serie de técnicas que permitieron

recolectar la información (observación, entrevistas, encuestas), la cual fue analizada y procesada conforme a lo planteado en los objetivos del estudio. Se utilizó el algoritmo propuesto por Mendoza, Pérez, Grimán y Rojas (2002) para la evaluación de MOSCA; se instanció el modelo, es decir, se adoptaron sólo las métricas que fueran pertinentes a un grupo de investigación en un entorno científico académico sin fines de lucro, esto con el apoyo de expertos del área pertenecientes al grupo LISI (2010).

Fueron publicadas las encuestas a través de una herramienta web (http://www.e-encuesta.com) de acuerdo a la instanciación del modelo en cada característica, donde se eliminaron preguntas referentes a una empresa comercial, dejando las institucionales y las que aplicaban al tipo de desarrollo.

De acuerdo al algoritmo de MOSCA, se escogieron tres características para la evaluación de la perspectiva del producto SQLfi V4 las cuales fueron: funcionalidad, mantenibilidad y portabilidad. Las métricas de funcionalidad fueron adaptadas para que pudieran ser contestadas tanto para los líderes del proyecto (investigadores de la USB, UC y del IUTFRP) y desarrolladores que conocieran el producto en cuestión. En total once (11) personas contestaron las métricas, haciendo la salvedad que las respuestas no eran obligatorias porque algunas métricas eran muy específicas que podían ser contestadas sólo por los programadores.

Basado en el análisis de los resultados se formuló una propuesta para la construcción de software de calidad para el proyecto "Creación y Aplicación de Manejadores de Base de Datos Difusas" utilizando un enfoque de manufactura esbelta (*Lean Manufacturing*), con la contribución de los aportes de la evaluación utilizando MOSCA y la experiencia de los investigadores.

En otro orden de ideas el Modelo Sistémico de Calidad (MOSCA) plantea, sobre la base de las seis (6) características de calidad del estándar internacional ISO/IEC 9126 (1991), un conjunto de categorías, características y métricas asociadas a la calidad y hacen del modelo un instrumento de evaluación de gran valor, ya que cubre aspectos imprescindibles para medir la calidad del producto de software. En cuanto a la perspectiva del proceso, se formuló sobre la base de las cinco (5) características de calidad del estándar internacional ISO/IEC 15504 (1991), un conjunto de categorías, características y métricas asociadas a la calidad de un proceso de software con un enfoque sistémico. El mejoramiento del sistema no sólo implica asegurar que éste opere de acuerdo a las expectativas de los usuarios; sino además, identificar las desviaciones e investigar cómo se puede mejorar (aplicación del principio de mejoramiento continuo). MOSCA consta de 4 niveles que se describen a continuación.

Nivel 0: *Dimensiones*. Aspecto Interno y Contextual de la Perspectiva: del Producto, del Proceso y Humana; sólo un balance y una buena interrelación entre ellas permitirá garantizar la calidad sistémica de una organización.

Nivel 1: *Categorías*. Producto: Funcionalidad (FUN), Fiabilidad (FIA), Usabilidad (USA), Eficiencia (EFI), Mantenibilidad (MAB) y Portabilidad (POR). Proceso: Cliente - Proveedor (CUS), Ingeniería (ENG), Soporte (SUP), Gestión

(MAN) y Organizacional (ORG). Humana: Individual (IND), Equipo (EQU), Entorno Empresarial (ENT).

Nivel 2: *Características***.** Cada categoría tiene asociado un conjunto de características (56 asociadas al producto y 27 al proceso de desarrollo y 15 a la perspectiva humana), las cuales definen las áreas claves a satisfacer para lograr, asegurar y controlar la calidad tanto en el producto, el proceso como en las personas.

Nivel 3: *Métricas*. Consta de un total de 715 métricas para medir la calidad sistémica.

Para un estudio más profundo de MOSCA puede revisarse a Mendoza, Pérez y Grimán (2005), su algoritmo en Mendoza, Pérez, Grimán y Rojas (2002) y las diferentes publicaciones del grupo de investigación LISI (2010) sobre el tema.

La Manufactura Esbelta según Womack y Jones (2003) constituye la evolución de los principios de calidad desarrollados por la empresa Toyota desde principios de los años 50, donde se desplegaron un conjunto de técnicas de gestión asociadas al sistema de producción *Toyota Production System* (*TPS*) descrito por Ohno (1988).

La idea fundamental del TPS es eliminar toda clase de desperdicio, resaltar las actividades que añaden valor al producto, manufacturar de acuerdo a la demanda de los clientes (minimizando inventarios) y enfocarse en las personas que agregan valor. Según Feld (2000), esta filosofía permite desarrollar no sólo productos sino procesos de calidad; además, empodera a las personas que tienen que ver con el proceso de manufactura, con el objetivo primordial de satisfacer al cliente con productos de alta calidad, entregados a tiempo y a un costo razonable.

El TPS se basa en la utilización del método científico de identificación y resolución de problemas reflejado en el ciclo: planificar – hacer – verificar – actuar (*PDCA*) de Deming descrito por Scholtes, Joiner y Streibel (2003); además, según Parker (2008) en el hecho de que las personas más cercanas a la agregación de valor al producto tienen la capacidad para tomar decisiones con autonomía.

Para ayudar en la tarea de la eliminación del desperdicio de acuerdo a Womack, Jones y Roos (1991) existen diversas herramientas tales como: el mapa de la cadena de valor, las 5S (seiri: organización, seiton: orden, seiso: limpieza, seiketsu: estandarizar, shitsuke: disciplina), la tarjeta visual (Kanban), las herramientas asociadas al mantenimiento total productivo (TPM), el despliegue de la función de calidad, herramientas CASES (MsProject en red), u otras aplicaciones que permiten consolidar más eficientemente los informes de avances, entre otras.

Según Ward (2007), existen en el TPS dos pilares fundamentales; uno organizativo que hace referencia al proceso de despliegue de políticas (*Hoshin Kanri*), estandarización, proceso de desarrollo de nuevos productos y procesos (3P); otro más técnico que se refiere a conceptos como el justo a tiempo (*JIT*), la automatización inteligente (*Jidoka*) o la nivelación de cargas de producción (*Heijunka*).

Por otro lado, la Manufactura Esbelta de acuerdo a Santos, Wysk y Torres (2006), permite a los fabricantes hacer menos partes al mismo tiem-

po y a menor costo, ya sea en masa o de producción artesanal. Los métodos de producción esbelta tienen la finalidad de acelerar el rendimiento, reducir los inventarios, aumentar la personalización y mejorar la calidad.

Ahora bien esta filosofía en el ámbito de la Ingeniería del Software se ha denominado desarrollo de software esbelto de gran utilidad para empresas desarrolladoras de software que deben sobrellevar múltiples presiones para mejorar la calidad de sus productos y efectuar desarrollos en forma ágil. Algunos autores tales como Muller (2000) hacen la analogía que el desarrollo de un proyecto de software es como un barco que uno sabe la fecha en que zarpa pero no la fecha en que va a llegar a puerto seguro.

La complejidad que supone la tarea de programar y los cambios del software para ser continuamente adaptado a los requerimientos de los usuarios según Presmann (2002), hacen que en la creación de productos informáticos no se obtengan los resultados deseados, además de su alto costo y poca flexibilidad; de acuerdo a Zeller (2005), esto reside en la dificultad de escribir programas libres de defectos, que sean fácilmente comprensibles y verificables; Glass (2006) ha utilizado el término "crisis del software" para describir esta problemática y sus consecuencias.

Aunque las estimaciones que realizan los responsables de proyectos no son siempre próximas a la realidad, existen herramientas para estimar duraciones y esfuerzos para la planificación de proyectos (PERT-CPM, MSProject). Sin embargo, para Sommerville (2002) no existen herramientas que permitan estimar de una manera fiable, antes de comenzar un proyecto de desarrollo de software, cuál es el esfuerzo que se necesitará para el mismo. Este hecho provoca que la mayoría de las veces no sea posible estimar acertadamente cuánto tiempo llevará el proyecto, ni cuánto personal será necesario.

Del mismo modo, en muchas ocasiones se incrementa el personal asignado a un proyecto para tratar de disminuir el plazo de ejecución de acuerdo a lo observado por Cohn (2005). Además, Spinellis (2006) afirma que los desarrolladores no están acostumbrados a leer código, comenzando normalmente los desarrollos desde cero, por lo que se deja de aprovechar la creciente cantidad de software libre disponible y se corre el riesgo de "reinventar la rueda". Asevera Tian (2005) que en muchas oportunidades se obtienen productos de baja calidad, que no cumplen las especificaciones y cuyo código es difícil de mantener.

Autores como Poppendieck y Poppendieck (2009) y Larman (2008) han estudiado la aplicación de la manufactura esbelta al desarrollo de software. Los componentes del software son altamente personalizados y deben ser construidos bajo demanda, es más un trabajo artesanal que una producción en masa; debe estandarizarse los procesos de producción y permitir intercambiar las personas en las diversas actividades del proceso.

Análisis de resultados

De acuerdo al algoritmo de MOSCA, una métrica se considera aprobada si su evaluación está ubicada entre las tres primeras de cinco al-

ternativas de la escala tipo Likert. De igual modo para considerar una característica aprobada debe cumplirse al menos con el 75% de las métricas, este criterio también se aplica al nivel superior (Categoría). En la **Tabla 1** se resume la evaluación de las métricas instanciadas de las características de la categoría "funcionalidad".

Del total de siete (7) características medidas de la categoría funcionalidad del producto sólo tres (3) fueron aprobadas, las cuales resultaron: precisión (FUN.2), correctitud (FUN.5) y estructurado (FUN.6) lo que representa el 42,86% de los mismos; resultando rechazadas: ajuste a los propósitos (FUN.1), interoperabilidad (FUN.3), encapsulado (FUN.7) y especificado (FUN.8) que constituye el 57,2% de las características. Para ver en detalle la encuesta realizada, cada una de las métricas que fue instanciada (aprobada y rechazada), puede revisarse Cadenas (2010).

Los resultados indican que el producto SQLfi V4 no cumple con la categoría Funcionalidad por lo que, de acuerdo al algoritmo utilizado, la evaluación termina en este punto. El resto del sub-modelo: mantenibilidad y portabilidad del

Tabla 1

Evaluación de métricas y características de "funcionalidad"

Característica	Número de métricas aprobadas	Número de métricas rechazadas	%	Característica aprobada
FUN.1 Ajuste a los propósitos	3	2	60%	NO
FUN.2 Precisión	3	1	75%	SI
FUN.3 Interoperabilidad	1	3	25%	NO
FUN.5 Correctitud	7	1	88%	SI
FUN.6 Estructurado	1	0	100%	SI
FUN.7 Encapsulado	0	1	0%	NO
FUN.8 Especificado	0	1	0%	NO
% Aprobación Categoría Funcionalidad				42.86%

producto; así como las demás perspectivas (proceso y humana) no fueron evaluadas, dado que la categoría funcionalidad es considerada la más importante, toda vez que identifica la capacidad del software para cumplir las funciones para lo que fue elaborado y por lo tanto el nivel de calidad sistémico resultó ser "Nulo".

Propuesta

La contribución principal del presente estudio es adaptar los postulados del desarrollo de software esbelto dentro de un proyecto en un entorno científico académico tal como el de "Creación y Aplicación de Sistemas Manejadores de Bases de datos Difusas", de acuerdo a la experiencia de los autores, análisis derivado del estudio de campo y práctica de los investigadores asociados al grupo.

Para lograr que el modelo propuesto de mejora sea factible es necesario sistematizar el desarrollo de software donde se involucre el producto, la disciplina en el proceso y se establezca un compromiso de las personas involucradas de forma que compartan la visión con los líderes de proyecto, conformando equipos de trabajo colaborativos y motivados. A continuación se describen los puntos principales del modelo.

El proceso de desarrollo de software debe centrarse en las personas.

La elaboración de software depende en gran medida de las personas que intervienen en el mismo, así como la manufactura esbelta faculta a las personas que hacen el trabajo para que sean las que determinen la mejor forma de hacer las cosas, los líderes del proyecto objeto de esta investigación deben comprender que la forma de mejorar la calidad de los productos es facultar a las personas que hacen el trabajo de desarrollo: darles entrenamiento, los recursos tecnológicos, disciplina, herramientas y el apoyo para resolver los problemas, lo que permitirá mejorar continuamente los procesos.

Para la conformación de equipos de trabajo se recomienda los basados en características (*feature team*) descritos por Larman (2008) y utilizados en metodologías de desarrollo ágil como *Scrum* descrita por Schwaber (2004). Estos equipos se concentran en características, dando pequeños pasos a la vez (análisis, diseño, implementación, prueba y distribución) utilizando la filosofía del proceso de desarrollo iterativo e incremental.

Es de hacer notar que los roles de las personas que conforman los equipos pueden variar de acuerdo a la característica desarrollada; se clasifican a las personas según sus habilidades principales y secundarias, pero las personas deben continuar aprendiendo otras habilidades a través de la interacción con los otros integrantes del equipo de desarrollo y por la rotación en los diversos roles al completar cada iteración.

El desarrollo debe hacerse con una lista priorizada de características deseables a un alto nivel (*backlog*); antes de implementar alguna característica (*feature*) es analizada por los miembros de un equipo quienes conocen el dominio del cliente (contexto donde se establecen las reglas del negocio) y la tecnología. Estas características deben ser divididas en historias

(*stories*) las cuales son unidades de desarrollo que pueden ser estimadas en forma confiable y pueden ser implementadas en pocos días.

En reuniones de planificación el equipo determina como pueden implementarse varias historias en la próxima iteración basados en estadísticas y el compromiso para completar las mismas. Durante la iteración el equipo se reúne en forma breve diariamente para hablar sobre cómo va la planificación, monitorear el compromiso de implementación y ayudarse mutuamente si existen obstáculos para continuar.

Al final de la iteración las historias deben ser integradas, probadas, documentadas y distribuidas para su utilización. Se efectúa una reunión de revisión para demostrar el progreso y obtener retroalimentación, la cual puede ser capturada como un registro o un cambio en lo pendiente. Después de pocas iteraciones se completa un conjunto de características útiles listas para distribuirse.

Una consideración importante, dentro de esta propuesta es facultar a los desarrolladores para encontrar soluciones a los problemas y no hacer tanto énfasis en la documentación, ésta última debería convertirse rápidamente en obsoleta si las personas están pensando continuamente en cómo cambiar las cosas para mejorar. En otras palabras, cambios constantes en la documentación más bien demuestran que una organización ha aprendido a pensar.

Las personas también deben encargarse de reducir los ciclos de tiempo en el mapa de la cadena de valor concentrándose en actividades que añaden valor al proceso que, en el caso del software, son las características requeridas por el usuario; esto se puede lograr mediante la colaboración, efectuando procesos de calidad a través de las seis disciplinas básicas para el flujo del desarrollo de software: organizar el área de trabajo, utilización de estándares, instalar control de versiones, aligerar procesos de construcción, efectuar integración constantemente y establecer políticas de pruebas.

El equipo del proyecto se debe enfocar en cumplir con los plazos, la cantidad de características añadidas al software, poder realizar cambios en los requerimientos del cliente (flexibilidad en el desarrollo), lo que producirá una reducción en el ciclo de vida, sin concentrarse en optimizar los subsistemas de medición e inspección posterior.

En cuanto a los líderes del proyecto, estos deben ser permanentes motivadores de los desarrolladores, ya que en el estudio realizado los equipos de trabajo se conforman con estudiantes de diversas instituciones, cuya meta principal es graduarse en el tiempo menor posible y no se comprometen a largo plazo con el proyecto. Luego, los objetivos de los desarrolladores a veces no coinciden con la visión de los líderes del proyecto, ya que a estos últimos requieren que se logren productos a más largo plazo y con calidad de acuerdo a la planificación del proyecto.

Al aplicar el modelo propuesto se puede lograr que se tenga una visión compartida donde se obtengan productos de calidad, en los plazos esperados y a un costo razonable; pero es muy importante que los líderes induzcan esa motivación al logro a través de una permanente comunicación y estimulando la perseverancia en los estudiantes.

Utilizar herramientas de manufactura esbelta para el desarrollo de software de calidad.

Tomando en consideración que el equipo de proyecto está conformado por estudiantes y profesores de distintas universidades, lo cual contrasta con el personal de empresas de desarrollo de software; en esta investigación se adecuaron las técnicas de manufactura esbelta aplicables directamente al software, seguidamente se explican cada una de ellas.

Kanban con un gran pizarrón para monitorear que está haciendo en cada característica (diseño, codificación, prueba, compilación y distribución) los grupos de desarrolladores y que está pendiente por hacerse. Esto puede ser apoyado por herramientas automatizadas de control de proyectos o aplicaciones que permitan consolidar informes de avances.

Andon utilizando un diagrama que muestra el progreso global del proyecto (denominado *burndown*) e indicando el trabajo restante en el tiempo que denota de un solo vistazo si el proyecto va a tiempo o no.

*Las cinco **S***: organizar el trabajo, sistematizar, estandarizar, mantener orden y limpieza en el sitio de trabajo, para luego mantener esta disciplina en el desarrollo de los proyectos.

Heijunka, nivelar el trabajo de los estudiantes y establecer un balance entre las labores en la academia (tanto estudiantes como profesores) y las inherentes al proyecto. Aquí de nuevo es importante la utilización de herramientas de control de proyecto.

Poka Yoke, a través de la especificación del código de los programas que permite implementar la verificación de los errores antes de que ocurran en lugar de hacer énfasis en la inspección (control posterior), además de utilizar restricciones en los sistemas gestores de base de datos para forzar la validación y limpieza de los datos ingresados.

Método de mejoramiento continuo.

Luego de establecer un desarrollo centrado en las personas y aplicar herramientas de manufactura esbelta al desarrollo de software se deben utilizar técnicas de medición para saber si está resultando la metodología aplicada. De acuerdo a lo observado por los investigadores de este estudio para lograr una calidad sistémica, se deben establecer técnicas que permitan hacer en forma diferente las cosas a cómo se están haciendo hasta el momento en el proyecto estudiado.

Para ello se propone el método de mejoramiento continuo resultante de una aplicación directa del ciclo de mejora de calidad de Deming el cual consiste en un ciclo iterativo e incremental de cuatro pasos: Planificar, Hacer, Verificar y Actuar. A continuación se detalla su aplicación: a) *Planificar* a través de un proceso de desarrollo centrado en las personas (tal como se expresó en el punto 1, acerca del proceso de desarrollo de software centrado en las personas); b) *Hacer* aplicando herramientas de manufactura esbelta (tal como se expresó en el punto 2 acerca de la utilización de herramientas para el desarrollo de software de calidad); c) *Verificar,* mediante la aplicación del modelo sistémico de calidad (MOSCA) a los diversos productos que se desarrollen dentro del proyecto; d) *Actuar* en base

a la evaluación obtenida en las métricas, los líderes del proyecto tienen que aplicar correctivos.

Los cuatro pasos del ciclo de calidad de Deming deben aplicarse en forma iterativa e incremental para ir acercándose poco a poco a la meta, la cual es mejorar en forma continua las personas, el proceso de desarrollo y la calidad global de los productos obtenidos en el proyecto. De esta forma se logrará implementar los principales postulados perseguidos por la manufactura esbelta:

> La perfección no es posible pero se puede llegar cerca con aproximaciones graduales continuas.
>
> Hacer las cosas con calidad desde la primera vez cometiendo los menos errores posibles.
>
> Pensar siempre que para lograr alcanzar un gran trayecto debemos comenzar con un paso y cada paso que demos debemos estar seguros que es en el camino correcto.

Lista de actividades para los líderes del proyecto.

Una lista de actividades que deben ser permanentemente vigiladas, enmarcadas dentro de esta propuesta, que puede servir de guía para los líderes del proyecto se enumera a continuación:

1. Efectuar desarrollos centrados en las personas.
2. Implementar equipos enfocados en características.
3. Complementar los conocimientos de las personas involucradas en el proyecto en metodologías ágiles, como desarrollo de software esbelto; además de herramientas para el desarrollo de software de calidad, pruebas de verificación (de aceptación y de código), utilización de estándares y controlador de versiones.
4. Hacer énfasis en la utilización de especificaciones en los programas y en el caso de base de datos aplicar las restricciones a través de las herramientas que provee el DBMS.
5. Validación de interoperabilidad de aplicaciones web (CSS y XHTML).
6. Analizar el mapa de valor para el desarrollo de software, identificando aquellas actividades que no agregan valor, es decir, aquellas que no incidan en el cumplimiento de los requerimientos de los usuarios.
7. Utilizar herramientas generadoras de documentación para código fuente tal como *Doxygen* (http://www.doxygen.org) y de colaboración (Wiki).
8. Asignar las personas al desarrollo de una característica a la vez; evitando en lo posible la asignación a múltiples desarrollos, lo cual causa más interrupciones que beneficios.
9. Efectuar la programación por pares.
10. Evitar diseños detallados prematuros ya que restringen el aprendizaje, predispone el impacto de los defectos e incrementa el costo del cambio.
11. Asegurar que los desarrolladores tengan acceso a los líderes del proyecto para obtener respuestas a sus preguntas tan pronto como sea posible; se recomienda efectuar reuniones periódicas (diarias y semanales)

de acuerdo a lo recomendado por *Scrum* e implementar herramientas de comunicación que sean efectivas.

12. Utilizar técnicas de programación como la fijación de estándares de codificación y control de versiones para una adecuada gestión del código fuente.
13. Aplicar el ciclo de mejora de calidad de Deming (Planificar, hacer, verificar y actuar) en forma iterativa e incremental.

Conclusiones y recomendaciones

La evaluación de la calidad del desarrollo del software, caso de estudio SQLfi V4, en el marco del proyecto "Creación y Aplicación de Sistemas Manejadores de Bases de Datos Difusas" conformado por un grupo de investigadores de la USB, UC e IUTFRP; dio como resultado un nivel de calidad sistémica "Nulo", equivalente a un grado de madurez de la organización "inicial" según el modelo CMMI. Por otra parte, se observó deficiencias en la productividad debido a que los tiempos se extendieron más de lo planificado y se utilizaron más recursos de los previstos.

Se presentó una propuesta para la construcción de software de calidad para el proyecto en estudio se fundamenta en el enfoque de manufactura esbelta, con los aportes derivados del resultado de la evaluación utilizando MOSCA, así como la experiencia de los investigadores. La contribución es un modelo que permite el desarrollo de productos de mayor calidad sistémica, a bajo costo y con entregas ágiles en un entorno científico académico.

Los puntos principales del modelo son: el proceso de desarrollo de software debe centrarse en las personas, utilizar herramientas de manufactura esbelta para el desarrollo de software con calidad, aplicación del método de mejoramiento continuo y chequeo de una lista de actividades enmarcadas en la propuesta como guía para los líderes de proyecto.

Las herramientas específicas de manufactura esbelta consideradas para el desarrollo de software con calidad mencionan a continuación: *Kanban*, *Andon*, las cinco **S**, *Heijunka*, *Poka Yoke*; las cuales deben aplicarse tomando como premisa que su objetivo es perfeccionar el producto, el proceso de desarrollo y las personas que intervienen en dicho proceso. Se recomienda la utilización del modelo MOSCA para medir la calidad sistémica de los productos y de esta forma verificar que el modelo propuesto está surtiendo efecto, además de permitir corregir las fallas dando origen a nuevas técnicas que permitan cambiar para mejorar hasta lograr minimizar las posibles fallas o errores.

Es importante hacer énfasis que el grupo de investigadores asuman del proyecto "Creación y Aplicación de Manejadores de Bases de Datos Difusas" asuman el modelo propuesto para la mejora de la calidad sistémica de los productos desarrollados, tanto para éste como para futuros proyectos que emprendan; de esta manera será posible evaluar el impacto de la propuesta en un entorno científico académico.

Con miras a que las universidades que imparten carreas en el área de computación, sistemas o informática, puedan formar profesionales capaces de responder a las necesidades del entorno se

recomienda estudiar la posibilidad de incluir en el pensum de estudio metodologías de desarrollo de software ágil, herramientas de automatización de disciplinas básicas y lecturas de código de calidad; para así lograr amplificar el aprendizaje en el desarrollo de software con calidad.

Bibliografía

Cadenas, J. (2010). *Propuesta de Mejora de la Calidad de Software bajo el Enfoque de Manufactura Esbelta*. Trabajo de Grado presentado para Título de Msc en Ingeniería Industrial. Universidad de Carabobo, Valencia, Venezuela.

CEISOFT (2010). Centro de Excelencia en Ingeniería del Software**.** Recuperado el 27-11-2010 del sitio web http://www.ceisoft.org.

Chrissis, M., Konrad, M. y Shrum, S. (2007). *CMMI Guidelines for Process Integration and Product Improvement***.** 2nd Edition. SEI Series in Software Engineering. USA: Addison-Wesley.

Cohn, M. (2005). *Agile Estimating and Planning***.** USA: Addison-Wesley.

Crespo, V. (2006). *Reingeniería del Sistema de Consultas Difusas a Bases de Datos SQLfi***.** Trabajo de Grado para título de Ingeniero en Computación, Universidad Simón Bolívar, Sartenejas, Venezuela.

Feld, W. (2000). Lean Manufacturing: tools, techniques, and how to use them. *Series on Resource Management*. USA: The St. Lucie Press/APICS.

FONACIT (2010). Mediante aplicaciones SQLFI y PostgreSQL Investigadores proponen uso de lógica difusa. Recuperado el 27-11-2010 del sitio web http://www.fonacit.gov.ve/noticias.asp?id=490.

Fundación CENDITEL (2010). Fundación Centro Nacional de Desarrollo e Investigación en Tecnologías Libres. Recuperado el 27-11-2010 del sitio web Ministerio Popular para Ciencia, Tecnología e Industrias Intermedias, Gobierno Bolivariano de Venezuela: http://www.cenditel.gob.ve/.

Glass, R. (2006). The Standish Report: Does it Really Describe a Software Crisis? *Communications of the ACM*. Vol. 49. No. 8.

Goncalves M., Tineo L. (2008). SQLfi and its Applications. *Avances en Sistemas e Informática,* Vol 5 No. 2. Medellin, Colombia. ISSN 1657-7663.

ISO/IEC 9126 (1991). JTC 1/SC 7. Information technology - Software product evaluation - Quality characteristics and guidelines for their use. *JTC 1 Organization*, Montreal, Quebec.

ISO/IEC 15504 (1991) JTC 1/SC 7. Software Process Assessment. TR 15504. *WG 10: Software Process Assessment*. ISO/IEC Organization.

Larman, C. (2008). *Scaling Lean & Agile Development: Thinking and Organizational Tools for Large-Scale Scrum.* USA: Addison Wesley.

LISI (2010). Laboratorio de Investigación en Sistemas de Información. Recuperado el 27-11-2010 del sitio web de la Universidad Simón Bolívar, Sartenejas, Venezuela: http://www.lisi.usb.ve.Mendoza L., Pérez, M. y Grimán, A. (2005). Prototipo de Modelo Sistémico de Calidad (MOSCA) del Software. *Computación y Sistemas*, Vol. 8, No. 3, pp. 196-217, México.

Mendoza, L., Pérez, M., Grimán, A. y Rojas, T. (2002). Algoritmo para la Evaluación de la Calidad Sistémica del Software. *2das. Jornadas Iberoamericanas de Ingeniería del Software e Ingeniería del Conocimiento*. Salvador, Brasil.

Muller, P. (2000). *Modelado de Objetos con UML*. Barcelona, España: Editorial Gestión 2000 S.A.

Ohno, T. (1988). *Toyota Production System. Beyond Large-Scale Production*. USA: Productivity. Inc.

Parker, J. (2008). *Do the Right Thing: How Dedicated Employees Create Loyal Customers and Large Profits*. USA: Pearson Education.

Poppendieck, M. y Poppendieck, T. (2009). *Leading Lean Software Development*. USA: Addison Wesley.

Santos, J., Wysk, R. y Torres, J. (2006). *Improving Production with Lean Thinking*. USA: John Wiley & Sons.

Scholtes, P., Joiner, B. y Streibel, B. (2003). *The Team Handbook*. Third Edition. USA: Oriel Inc.

Sommerville, I. (2002). *Ingeniería del Software*. Sexta Edición, México: Addison Wesley.

Spinellis, D. (2006) Code Quality: The Open Source Perspective. *Effective Software Development Series*. USA: Addison-Wesley.

Tian, J. (2005). *Software Quality Engineering. Testing, Quality Assurance and Quantifiable Improvement*. IEEE Computer Society Press. USA: John Wiley & Sons.

Ward, A. (2007). *Lean Product and Process Development*. USA: The Lean Enterprise Institute.

Womack, J. y Jones, D. (2003). *Lean Thinking*. Simon & Schuster. Second Edition. U.K.: Free Press.

Womack, J., Jones, D. y Roos, D. (1991). *The Machine that Changed the World. The Story of Lean Production*. NY, USA: Harper Perennial.

Zeller, A. (2005). *Why Programs Fail: A Guide to Systematic Debugging*. Elseiver. USA: Morgan Kaufmann.