Metodología iterativa de desarrollo de *software* para microempresas

Iterative *software* development methodology for micro-enterprises

Luis-Alexander Calvo-Valverde[1]

Fecha de recepción: 1 de octubre del 2014
Fecha de aprobación: 3 de febrero del 2015

Calvo-Valverde, L. Metodología iterativa de desarrollo de *software* para microempresas. *Tecnología en Marcha*. Vol. 28, Nº 3, Julio-Setiembre. Pág 99-115.

1 Centro de Investigaciones en Computación. Escuela de Ingeniería en Computación. Instituto Tecnológico de Costa Rica, Cartago, Costa Rica. Correo electrónico: lcalvo@itcr.ac.cr

Palabras clave

Desarrollo de software; calidad; metodologías.

Resumen

En los últimos años, Costa Rica ha experimentado un creciente aumento en el número de microempresas de desarrollo de software, pero este crecimiento no ha ido acompañado del uso de metodologías apropiadas para el desarrollo de software en este tipo de empresas. Esta situación se debe a varios motivos: carencia de formación en ingeniería de software por parte de los creadores de la compañía, urgencia aparente de producir código a toda costa en detrimento de la planificación, uso inadecuado de metodologías que generan un exceso de trabajo administrativo, entre otros.

Aunado a lo anterior, el mundo empresarial vive vertiginosas transformaciones que demandan soluciones que rápidamente se adapten a los cambios. Las metodologías de desarrollo iterativas, como la propuesta en el presente trabajo, permiten más ágilmente adaptarse a esta variabilidad de requerimientos que vive el sector de TI.

El presente trabajo hace un diagnóstico sobre las características de las microempresas costarricenses de desarrollo de software; seguidamente, determina las mejores prácticas en las metodologías de desarrollo de software, para luego hacer un análisis comparativo y proponer una metodología de desarrollo que se adecúe a este tipo de organizaciones.

Keywords

Development of software; quality; methodologies.

Abstract

In recent years, Costa Rica has experienced a steady increase in the number of micro software development companies, but this growth has not been accompanied by the use of appropriate software development methodologies for such companies. This situation is due to several reasons: lack of knowledge on software engineering, urgency to generate code without much planning, inappropriate use of methodologies that generate excessive administrative work, among others.

In addition, the business world undergoes dizzying transformations that require solutions that quickly adapt to changes. The iterative development methodologies allow more nimbly adapt to this variability of requirements that the sector of IT.

This paper makes an assessment of the characteristics of Costa Rican micro software development companies, then determines the best practices in software development methodologies and then do a comparative analysis and propose a development methodology that suits this type of organizations.

Introducción

Aquellas personas emprendedoras que se han lanzado al reto de constituir una microempresa de desarrollo de *software* encuentran varios obstáculos de entrada al mercado, uno muy importante es que no existe mucha documentación sobre la aplicación de metodologías de

desarrollo de *software* para este tipo de empresas. La mayor parte de la literatura existente está enfocada sobre todo a grandes y medianas empresas.

Comprobar la anterior afirmación fácil: basta buscar en internet, en el mercado o en una biblioteca, qué metodologías de desarrollo de *software* están pensadas para microempresas y se verá que son muy pocas. Muestra que la mayoría de metodologías de desarrollo de *software* no están pensadas en microempresas es que dichas metodologías al definir roles y responsabilidades de los miembros del equipo del proyecto se nota cómo se está pensando en equipos mucho más grandes que los normalmente manejados en una microempresa.

La situación anteriormente mencionada no es extraña al contexto costarricense, en Costa Rica la adopción en las microempresas de metodologías de desarrollo de *software* creadas para medianas y grandes empresas se vuelve un obstáculo importante sobre todo porque las microempresas –por su corta estructura administrativa, no más de cinco empleados– urgen de reducir los costos administrativos en los proyectos y entregar productos de alta calidad pues normalmente se ejecutan muy pocos proyectos a la vez y por tanto los ingresos dependen de uno o dos proyectos en ejecución.

El presente trabajo constituye un aporte a estos microempresarios al presentarles una propuesta metodológica que vele por la calidad en el desarrollo de *software* adecuada a su tipo de organización.

Situación del problema

En Costa Rica hay una gran cantidad de microempresas interesadas en desarrollar *software* de calidad, pero al incursionar en este campo, muchas veces, no cuentan con una metodología apropiada para hacerlo de un modo ordenado.

Según un estudio realizado por CAMTIC, Costa Rica cuenta con unas 805 empresas de en Tecnologías Digitales (TD); y, entre el 2003 y el 2008, hubo un aumento del 101%, del que el Subsector del *Software* fue el que más creció en número de empresas, exportando mayoritariamente sus servicios (CAMTIC, 2006).

CAMTIC indica que los principales mercados de exportación son América Central y los Estados Unidos. También se colocan productos en México, Suramérica y, en menor medida, en Europa y Asia. Al 2008, el Sector de las TD en Costa Rica reportaba ventas de aproximadamente US$3.500 millones anuales, un 10, 6% del PIB; US$2.800 millones en exportaciones (28.8% del total); y, un total de 54.700 empleos (3,4% de la fuerza laboral) (CAMTIC, 2006).

Ese mismo estudio indica que el Sector de las Tecnologías Digitales no solamente representa una importante proporción de la producción y de las exportaciones del país, sino que ha sido clave para que Costa Rica se vaya insertando en la Sociedad de la Información y el Conocimiento (CAMTIC, 2006).

Aspectos Metodológicos

Objetivos

Objetivo general

Proponer una metodología iterativa de desarrollo de *software* para microempresas costarricenses.

Objetivos específicos

- Diagnosticar las características de las microempresas costarricenses de desarrollo de *software*.
- Determinar mejores prácticas en las metodologías de desarrollo de *software*.
- Analizar de manera comparativa las características de las microempresas costarricenses de desarrollo de *software* y las mejores prácticas de calidad.
- Diseñar una metodología iterativa de desarrollo de *software* para microempresas costarricenses.

Variables

Para la consecución de los objetivos se desarrollaron las siguientes variables, las cuales, a su vez, marcan el orden del proceso seguido en la presente investigación:

- Diagnóstico de las características de las microempresas en costarricenses de desarrollo de *software*.
- Estudio sobre modelos iterativos para el desarrollo de *software*.
- Estudio sobre métricas para el aseguramiento de la calidad en el desarrollo de *software*.
- Estudio sobre mejores prácticas para mejorar la calidad en el proceso de desarrollo de *software*.
- Análisis comparativo entre las características de las microempresas costarricenses de desarrollo de *software* y las mejores prácticas de calidad.
- Propuesta de una metodología iterativa de desarrollo de *software* para microempresas costarricense

Aclaración de conceptos

Generalidades sobre la calidad en el desarrollo de software

La calidad en un producto de *software* se analiza desde tres perspectivas (Piattini, 2007):

- Calidad interna: medible en las características intrínsecas del producto como el código fuente.
- Calidad externa: medible en el comportamiento del producto como en una prueba.
- Calidad en uso: medible en la utilización efectiva por parte de los usuarios del producto.

La Norma ISO/IEC 9126

La Noma ISO/IEC 9126 es un estándar internacional para la evaluación de la calidad del *software*. Esta norma identifica seis características de calidad y cada una de ellas las subdivide en sub-características. Vela por la calidad interna y externa. Los criterios que propone son (ISO/IEC 9126, 2001):

- Funcionalidad: capacidad del producto de *software* para ofrecer funciones que reúnan una serie de condiciones y necesidades implícitas cuando el *software* se utiliza en determinadas condiciones.
- Confiabilidad: capacidad del producto de *software* para mantener un determinado nivel de rendimiento cuando se utiliza en condiciones especificadas.

- Usabilidad: capacidad del producto de *software* de ser comprendido, aprendido, usado y atractivo para el usuario, cuando utilizados en condiciones específicas.
- Eficiencia: capacidad del producto de *software* para proporcionar el rendimiento apropiado, relativo a la cantidad de recursos utilizados, bajo condiciones expuestas.
- Mantenibilidad: capacidad del producto de *software* para ser modificado. Las modificaciones pueden incluir correcciones, mejora o adaptación del *software* a los cambios en el medio ambiente, y de las necesidades y las especificaciones funcionales.
- Portabilidad: capacidad del producto de *software* que se han de transferir de un medio a otro. El medio ambiente puede incluir la organización, de hardware o *software*.

La Norma ISO/IEC 25000

En calidad del producto recientemente ha aparecido una nueva versión de la norma ISO/IEC 9126: la norma ISO/IEC 25000. Esta presenta una guía para el uso de las nuevas series de estándares internacionales, llamados Requisitos y Evaluación de Calidad de Productos de Software (SQuaRE) (ISO/IEC 25000, 2005). Constituyen una serie de normas basadas en la ISO 9126 y en la ISO 14598 (Evaluación del *Software*), y su objetivo principal es guiar el desarrollo de los productos de *software* con la especificación y evaluación de requisitos de calidad. Establece criterios para la especificación de requisitos de calidad de productos *software*, sus métricas y su evaluación. Los criterios a considerar son (ISO/IEC 25000, 2005):

- Funcionabilidad
- Rendimiento
- Compatibilidad
- Usabilidad
- Madurez
- Seguridad
- Mantenibilidad
- Portabilidad

Pruebas de software

En cuanto a las pruebas lo primero que se debe realizar es identificar los tipos de pruebas y los métodos y criterios para realizar las actividades de pruebas; cada nivel de pruebas requerirá un instrumento de medición de los entregables.

El objetivo primario de estas pruebas es descubrir las limitaciones del sistema y medir sus capacidades. Entre los tipos de pruebas se tienen:

- Pruebas de aceptación
- Pruebas automatizadas
- Pruebas beta
- Pruebas de conversión
- Pruebas de la documentación
- Pruebas del *hardware*
- Pruebas de interface
- Pruebas de integración

- Pruebas paralelas
- Pruebas de seguridad
- Pruebas de volumen y de estrés
- Pruebas de rendimiento
- Pruebas de regresión
- Pruebas unitarias
- Pruebas de aceptación del usuario

Modelos iterativos para el desarrollo de software

Sobre los modelos iterativos se dice que en una visión genérica, el proceso se divide en 4 partes: Análisis, Diseño, Código y Prueba. Sin embargo, para la producción del *software*, se usa el principio de trabajo en cadena o "Pipeline", utilizado en muchas otras formas de programación. Con esto se mantiene al cliente en constante contacto con los resultados obtenidos en cada incremento. Es el mismo cliente el que incluye o desecha elementos al final de cada incremento a fin de que el *software* se adapte mejor a sus necesidades reales. El proceso se repite hasta que se elabore el producto completo (Soto, 2010).

Ciclo de vida en el desarrollo de software

La ISO 12207 indica que un modelo de ciclo de vida es un marco de referencia que contiene los procesos, las actividades y las tareas involucradas en el desarrollo, la explotación y el mantenimiento de un producto de *software*, abarcando la vida del sistema desde la definición de los requisitos hasta la finalización del mismo. El ciclo de vida abarca toda la vida del sistema, desde su concepción hasta cuando no se utiliza. Además, a veces se habla de ciclo de desarrollo, el cual es un subconjunto del anterior pues éste empieza en el análisis y finaliza con la entrega del sistema al usuario (ISO/IEC 12207, 2008).

Estándares de calidad en software

Hay dos tipos de estándares para gestionar la calidad del *software* (Sommerville, 2006):

- Estándares del producto: Estos estándares se aplican sobre el producto de *software* que se está desarrollando.
- Estándares del proceso: Estos estándares definen el proceso que deben seguirse para desarrollar el *software*.

Métricas para el aseguramiento de la calidad

Una métrica de *software* es cualquier tipo de medida relacionada con un sistema, proceso o documento de *software*. Algunos ejemplos son las medidas que se utilizan para calcular el tamaño de un producto en líneas de código el índice de Fog que mide la claridad de un párrafo en un texto, el número de fallos encontrados en un producto de *software* entregado, y el número de personas/día requeridas para desarrollar un componente del sistema (Sommerville, 2006).

Principios del Manifiesto Ágil

No toda metodología iterativa es necesariamente una metodología ágil; por lo tanto, es importante aclarar los principios que se encuentran detrás del llamado Manifiesto Ágil(Highsmith, 2001):

- Primero: nuestra mayor prioridad es satisfacer al cliente mediante la implementación temprana y continua de *software* valioso.

- Segundo: bienvenidos los requerimientos cambiantes, incluso tarde en el desarrollo. Los procesos ágiles aprovechan el cambio para obtener una ventaja competitiva del cliente.
- Tercero: entregar frecuentemente *software* que funciona, de un par de semanas a un par de meses, con una preferencia por el plazo más corto.
- Cuarto: la gente del negocio y los desarrolladores deben trabajar juntos diariamente a lo largo del proyecto.
- Quinto: los proyectos deben ser generados con individuos motivados. Darles el medio ambiente y el apoyo que necesitan y confiar en ellos para realizar el trabajo.
- Sexto: el método más eficiente y eficaz de comunicación de la información dentro del equipo de desarrollo es la conversación cara a cara.
- Séptimo: el *software* que funciona es la medida principal del progreso.
- Octavo: los procesos ágiles promueven el desarrollo sostenible. Los patrocinadores, los desarrolladores y usuarios deberían ser capaces de mantener un paso constante indefinidamente.
- Noveno: atención continua a la excelencia técnica y al buen diseño, mejora la agilidad.
- Décimo: simplicidad - el arte de maximizar la cantidad de trabajo no hecho es esencial.
- Undécimo: las mejores arquitecturas, requisitos y diseños surgen de equipos auto-organizados.
- Duodécimo: a intervalos regulares el equipo reflexiona cómo ser más eficaces, luego afina y ajusta su comportamiento consecuentemente.

Microempresas en Costa Rica

¿Qué es una microempresa en Costa Rica? El Ministerio de Economía Industria y Comercio (MEIC) de Costa Rica se vale de la clasificación propuesta por la Caja Costarricense del Seguro Social para definir qué es una microempresa. En Costa Rica se clasifican las empresas según tamaño en los siguientes grupos (Ministerio de Economía Industria y Comercio, 2009):

- Microempresas: de 1 a 5 trabajadores.
- Pequeñas: de 6 a 30 trabajadores.
- Medianas: de 31 a 100 trabajadores.
- Grandes: más de 100 trabajadores.

En cuanto a la cantidad de microempresas se tiene (Ministerio de Economía Industria y Comercio, 2009):

- Sector agropecuario: 4650.
- Sector industria: 2355.
- Sector comercio: 9176.
- Sector servicios: 20 140.

Las empresas de desarrollo de *software* se ubican en el sector servicios. Sin embargo, es importante aclarar que no se dispone del desglose de este dato para saber cuántas de las 20140 corresponden a microempresas de desarrollo de *software*.

Empresas de desarrollo de software en Costa Rica

Respecto al desarrollo de *software* en Costa Rica, el estudio más completo y reciente que se encontró fue el desarrollado por CAMTIC, en el 2005, en colaboración con el Banco Central de Costa Rica y el Instituto Centroamericano de Administración de Empresas (CAMTIC, 2006).

CAMTIC ha clasificado las empresas de tecnologías de información y comunicación (TIC) en cuatro sectores:

- Productores: empresas que se encargan del diseño y desarrollo de distintos productos de *software* estandarizado.
- Servicios directos: empresas que se encargan de la producción de *software* a la medida, servicios de instalación, integración y soporte de sistemas.
- Servicios habilitados por TIC: empresas que dependen de las TIC para su negocio.
- Componentes: empresas que se encargan de producir los componentes necesarios para desarrollar las tecnologías de información y comunicación.
- Es necesario resaltar que no existe un inventario de cuáles de estas empresas son microempresas, solo se cuenta con un estudio general que involucra micro-, pequeñas, medianas y grandes empresas de desarrollo de *software*.

Análisis de Resultados

Variable: diagnóstico de las características de las microempresas costarricenses de desarrollo de software

Los resultados se resumen a continuación.

Número de empleados

En Costa Rica, el Ministerio de Economía Industria y Comercio (MEIC), valiéndose de la clasificación propuesta por la Caja Costarricense del Seguro Social, define microempresa como la compuesta de entre 1 a 5 trabajadores (Ministerio de Economía Industria y Comercio, 2009).

En la Fig. 1 se resumen los resultados obtenidos en la encuesta con respecto a este criterio.

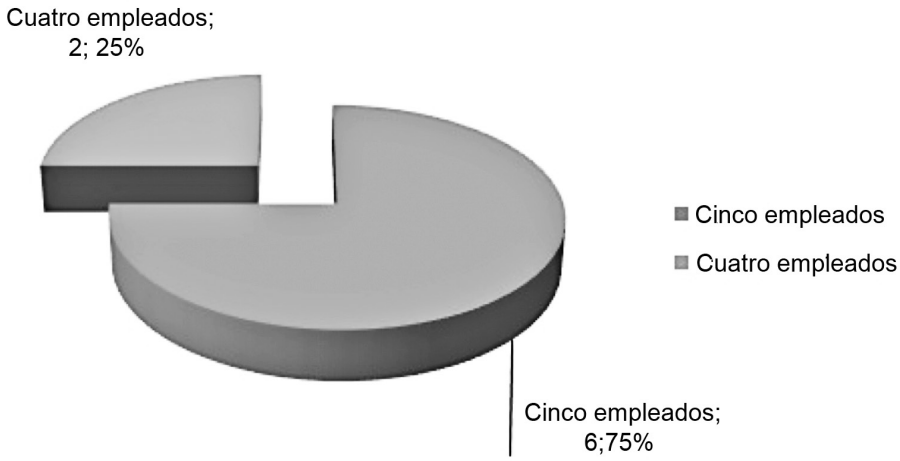

Figura 1. Cantidad de empleados en las empresas encuestadas

Como podemos constatar en la figura anterior, el 75% de las empresas encuestadas tienen 5 trabajadores y un 25% 4 trabajadores. Esto para un total de 8 empresas en la muestra.

Vale indicar que no aparecen microempresas con 1, 2 o 3 empleados, situación que no afecta significativamente el presente estudio pues el número de empleados que tienen las empresas encuestadas está dentro del criterio definido como microempresa.

Estructura organizativa

Otro criterio importante por considerar en la encuesta era cómo se estructuran organizacionalmente las microempresas de desarrollo de *software*.

Necesariamente, deben tener estructuras muy planas y con duplicación de funciones en varias personas, esto por tener no más de cinco empleados. Es decir, todos o casi todos los empleados deben hacer de todo. Máxime que, normalmente, los dueños de la microempresa son los principales trabajadores de su empresa.

Por el pequeño número de personal que conforma a las microempresas, la estructura organizativa tiende a ser muy sencilla:

- Una persona que dirige: llamado gerente, administrador o dueño.
- Encargado, administrador o líder de proyecto: una o dos personas.
- Programadores: dos o tres personas.

Vale resaltar que la misma persona puede fungir en varios roles. Por ejemplo, un encargado de proyecto puede ser, a la vez, programador; el que dirige la microempresa normalmente es, también, encargado de uno o más proyectos.

Fortalezas de este tipo de empresas

En esta sección se trata de definir qué fortalezas tienen este tipo de empresas. Esto será útil en la elaboración de la metodología pues ayuda a decidir que procedimientos y formularios se acomodan mejor al tipo de organización que son las microempresas de desarrollo de *software*.

Una microempresa de desarrollo de *software* tiene en resumen las siguientes fortalezas a explotar:

- Facilita el control organizacional.
- Organización flexible.
- Organización no compleja.
- Organización pequeña.
- Recurso humano comprometido

Debilidades de este tipo de empresas

Similarmente a lo indicado en la sección anterior, la determinación de las debilidades de este tipo de empresa será útil en la elaboración de la metodología pues ayuda a decidir qué procedimientos y formularios se acomodan mejor al tipo de organización que son las microempresas de desarrollo de *software*.

Una microempresa de desarrollo de *software* tiene en resumen las siguientes debilidades a cuidar:

- Limitada financieramente.
- Organización pequeña.

Metodología de desarrollo utilizada

Se puede concluir lo siguiente con respecto a la metodología:

- Se desea una metodología que tenga flexibilidad y que incorpore elementos de agilidad.
- Por lo general, entre Scrum o Cascada.
- Resumen del proceso de desarrollo
- Se consultó a los encuestados sobre el proceso seguido en el desarrollo de *software*, la siguiente tabla presenta las respuestas obtenidas.

Variable: Estudio sobre métricas para el aseguramiento de la calidad en el desarrollo de software

Se presenta seguidamente el resultado del estudio sobre métricas para el aseguramiento de la calidad en el desarrollo de *software*. Según lo planificado se tomó una muestra de la población total de métricas, es una muestra no probabilística y el criterio usado es afinidad con el tema de estudio (Sommerville, 2006)(ISO/IEC TR 9126-3, 2011).

- Longitud de código. Es una medida del tamaño del programa. Generalmente cuanto más grande sea el tamaño del código de un componente, es más complejo susceptible a errores.
- Longitud de los indicadores. Es una medida de la longitud promedio de los indicadores en un programa. Cuanto más grande sea la longitud de los identificadores, más probable será que tengan significado y el programa sería más comprensible. Da claridad a los programas.
- Profundidad del anidamiento de las condicionales. Esta es una medida de la profundidad de anidamiento de las instrucciones condicionales (p.ej. if) en un programa. Muchas condiciones anidadas son difíciles de comprender y son potencialmente susceptible de errores.
- Completitud de la documentación del usuario y/o facilidad de ayuda. Indica qué proporción de las funciones son descritas en la documentación del usuario o en la ayuda del sistema. Logra una buena documentación de los programas.
- Promedio de desviación en estimados de tiempo de desarrollo. Se calcula como la diferencia entre el tiempo estimado para un módulo y el tiempo real requerido. Luego se obtiene el promedio para el proyecto. Permite acumular experiencia que luego puede servir para estimar mejor otros módulos y proyectos.

Particularmente, se seleccionaron métricas que no requieren una gran inversión en tiempo y recursos para su cálculo, esto por cuanto, normalmente, el presupuesto de las microempresas es bajo.

Variable: Estudio de las mejores prácticas para evaluar la calidad del producto de software

Se presenta seguidamente el resultado del estudio sobre mejores prácticas en cuanto a calidad propuestas por el PMI en el PMBOK y la ISO/IEC 25000. Según lo planificado, se tomó una muestra de la población total de mejores prácticas, es una muestra no probabilística y el criterio usado es afinidad con el tema de estudio.

- Análisis Costo-Beneficio: evaluar si el costo de implementar la actividad de calidad es inferior al beneficio esperado.
- Costo de calidad: el costo de la calidad incluye tanto los costos para evitar fallos como los costos ocasionados por los fallos.

- Cartas de control: sirve para determinar si un proceso está estable. Se establece un límite superior y uno inferior de la especificación para tener un rango permitido.
- Benchmarking: se trata de comparar las prácticas actuales del proyecto con las de otros proyectos comparables para determinar las mejores prácticas.
- Diseño de experimentos: eEs un método estadístico para determinar qué factores pueden influir en variables específicas de un producto o proceso, ya sea en desarrollo o en producción.
- Muestreo estadístico: involucra escoger parte de una población de interés para inspeccionarla.
- Diagramas de flujo: es una representación gráfica de un proceso mostrando las relaciones entre los diferentes pasos del proceso.
- Metodologías de administración de la calidad propietarias: tales como Six Sigma, CMMI.
- Auditorías de calidad: es una revisión independiente y estructurada para determinar si las actividades del proyecto cumplen con las políticas, procesos y procedimientos, tanto de la organización como del proyecto.
- Diagrama causa-efecto: ilustra cómo varios factores producen ciertos efectos.
- Histograma: es un gráfico de barras vertical que muestra qué tan frecuente se da cierto hecho (por ejemplo, el valor de una variable). Cada columna representa un atributo o característica de un problema o situación en estudio. La altura de la columna representa la frecuencia relativa de esa característica.
- Diagrama de Pareto: es un tipo de histograma ordenado por frecuencia de ocurrencia.
- Carta de corrida: similar a las cartas de control sin mostrar límites. La carta de corrida muestra la historia y patrón de una variación.
- Diagrama de dispersión: sirve para mostrar la relación entre dos variables. Se tiene una variable independiente y una dependiente.
- Inspección: una inspección es el examen de un producto de trabajo para determinar si este está conforme a los estándares documentados.

Variable: Propuesta de una metodología iterativa de desarrollo de software de calidad para microempresas costarricenses

Para el desarrollo de esta sección se realizó el siguiente proceso:

- Analizar los modelos propuestos buscando el más apropiado para el caso en cuestión.
- Seleccionar un subconjunto de métricas de calidad para ser utilizadas en la metodología propuesta.
- Seleccionar un subconjunto de las mejores prácticas en cuanto a calidad.
- Determinar las principales características de las microempresas a tomar en cuenta para la preparación de la metodología.
- Preparar la metodología

Con respecto a los modelos de desarrollo de software

- RAD – Descartada. Se descarta pues para utilizarla se debe contar con las herramientas CASE, que no tiene la microempresa y está fuera de su capacidad de compra.

- FDD – Descartada. Se descarta pues es recomendada para proyectos grandes lo cual no es común que desarrollen las microempresas.
- XP – Descartada. Se descarta pues en este modelo el cliente debe estar dispuesto a dedicar mucho tiempo al proyecto, lo cual no es común en los proyectos de las microempresas. Además, es empleada en proyectos innovadores.
- Scrum – Descartada. Se descarta pues en este modelo se depende mucho de un buen Scrum Master. Para una microempresa capacitar o contratar un Scrum Master con experiencia es muy caro.
- Prototipado – Aceptada pero con modificaciones. Es el más adecuado para una microempresa por cuanto: Es recomendable cuando el cliente no está muy seguro de lo que quiere. Cuando el cliente desea ir haciendo pruebas del sistema y conocer la interfaz. Modificación, se procede a codificar parar ir generando versiones funcionales del programa. Esto es tomado de las metodologías ágiles.

Con respecto a las métricas de calidad

- Longitud de código– Aceptada. Adecuada en proyectos en que el programador es quien directamente escribe el código. Aceptada pues aplica para el caso.
- Longitud de los indicadores – Descartada. Adecuada en proyectos o muy pequeños o en los que se cuente con una herramienta automática que lleve las estadísticas de los indicadores. Rechazada pues no se cuenta con dicha herramienta automática.
- Profundidad del anidamiento de las condicionales – Descartada. En proyectos muy pequeños, o en los que se cuente con una herramienta automática que lleve las estadísticas de los indicadores. Rechazada pues no se cuenta con dicha herramienta automática.
- Completitud de la documentación del usuario y/o facilidad de ayuda - Aceptada. En general para todo tipo de proyectos. Aceptada pues aplica para el caso.
- Promedio de desviación en estimados de tiempo de desarrollo - Aceptada. En general, para todo tipo de proyectos. Aceptada pues aplica para el caso.

Con respecto a las mejores prácticas de calidad

Con el fin de no saturar la metodología se seleccionaron dos prácticas de las más recomendadas por su generalidad:

- Auditorías de calidad. Permite velar por la verificación del proceso de desarrollo del producto. Ejerciendo el aseguramiento de la calidad.
- Inspección. Permite velar por la validación del producto. Ejerciendo control de la calidad.

Las otras métricas si bien son útiles, el mismo costo-beneficio indica que no es aconsejable cargar la metodología de una microempresa de muchas actividades; por tanto, no se incluyen de momento y según necesidad se podrían incorporar más adelante:

Con respecto a las características de las microempresas en Costa Rica

Se presentas las características más relevantes de las microempresas costarricenses de desarrollo de *software* que fueron consideradas:

- Empresas pequeñas de menos de 5 empleados.
- Estructura organizativa muy horizontal y sencilla.
- Quien administra el proyecto normalmente es desarrollador y, por tanto, debe ser una

metodología sencilla de aplicar, que no le quite mucho tiempo en labores administrativas y de registro de información.

- La microempresa no tiene mucho presupuesto y, por ende, la metodología no debe implicar herramientas costosas.
- Es una metodología dirigida a proyectos pequeños o mediados pues la empresa no tiene más capacidad que esto (presupuesto menor a $100 000,00 y plazo no mayor a 6 meses).
- Normalmente no se cuenta con una cartera amplia de clientes, por eso la metodología debe velar porque el cliente sienta que su proyecto va dando los resultados esperados y así no quiera abandonar el servicio de la microempresa.

Propuesta Metodológica

La metodología propuesta se acompaña de Formularios y de un Diagrama del Proceso general, los cuales puede consultar en la siguiente dirección (http://goo.gl/sJZfT5).

Seguidamente, se presenta la metodología propuesta.

Toda reunión sostenida durante el proyecto será documentada a través de una Minuta de Reunión (Formularios F16). La minuta será confeccionada durante la reunión, y al finalizar deberá ser firmada por los asistentes como signo de aceptación. En ella además de registrar los asistentes, se llevará control de los puntos tratados y de los acuerdos alcanzados (en este caso indicando claramente quienes son los encargados de ejecutar dichos acuerdos).

Definiciones previas:

- Gerente del proyecto: es el responsable por parte de la empresa del proyecto y, a la vez, administra el mismo.
- Patrocinador: es la persona que contrata el proyecto.
- Cliente: Se refiere a la persona que el patrocinador designe como encargado de estar en el día a día del proyecto. Nada impide que, en algunos casos, patrocinador y cliente sean el mismo.
- Otros miembros del equipo: son los funcionarios de la empresa desarrolladora que participan en el proyecto, no incluye al gerente del proyecto.
- Equipo del proyecto: Se refiere al gerente del proyecto más los otros miembros del equipo del proyecto.

Proceso

Se presenta seguidamente el proceso a seguir. Vale aclarar que también se desarrollaron los formularios relacionados:

Realizar el Acta Constitutiva (Formularios F01)

- Se refiere al contrato inicial que marca los grandes acuerdos entre las partes.
- El número de proyecto es lo que le da unicidad a cada documento; por tanto, no deben existir dos Actas Constitutivas con un mismo número de proyecto.
- Debe ser firmado, al menos, por el gerente del proyecto y el patrocinador.
- Todo cambio que afecte los acuerdos tomados en el Acta Constitutiva debe ser aprobado con un documento de control de cambios debidamente firmado por los involucrados (Formularios F14), particularmente el gerente del proyecto y el patrocinador.

Planificación de las iteraciones (Formularios F02)

- Previo al inicio de las iteraciones, el equipo del proyecto y el cliente establecen las características generales del proceso iterativo como lo son el número de iteraciones, la duración de cada iteración, los roles y responsabilidades en el proceso y una planificación inicial del contenido de cada iteración. Este acuerdo inicial es solo para tener un plan general pues, por el tipo de metodología, será al inicio de cada iteración que cliente y equipo del proyecto determinen qué requerimientos se atenderán.
- El número de iteraciones dependerá, sobre todo, del número y complejidad de los requerimientos.
- La duración de cada iteración se recomienda que esté entre 10 a 20 días hábiles de trabajo por parte del equipo del proyecto. La decisión dependerá del esfuerzo que conlleven los requerimientos y de la disponibilidad del cliente para estar reuniéndose.

Preparar Plan de Inspección (Formularios F13) y Plan de Auditoría (Formularios F15),

- Como parte de la gestión de la calidad del proyecto se prepara un plan de inspección que ayudará en la validación del producto —control de calidad— y un plan de auditoría que ayudará en la verificación del proceso de construcción del producto —aseguramiento de la calidad—.

Mientras haya requerimientos a considerar:

- Toma detallada de requerimientos (Formularios F03)
  - Cada iteración tendrá un número único 01, 02, 03 y, así, sucesivamente.
  - A la vez se llevará una codificación de la versión de este documento: A, B, C,… Esta codificación servirá cuando el prototipo, en esa iteración, no sea aprobado y, por tanto, se volverán a tomar los requerimientos, en cuyo caso se tendría el mismo número de iteración, pero una nueva versión del documento (en la segunda vez “B”). La última versión de la iteración contendrá todos los requerimientos que se están desarrollando.
  - Se crean o modifican los Casos de Uso necesarios (Formulario F04). La primera vez que se prepara un caso de uso se le asigna el siguiente número disponible CU-01, CU-02, etc. Si se está creando ese caso de uso, se le asigna versión v1; si se está modificando por primera v2, y así sucesivamente.
- Construcción del prototipo (Formularios F05)
  - Los prototipos serán numerados en forma consecutiva 01, 02, 03,… por proyecto. Este será el identificador de cada uno de ellos.
  - Todo prototipo responde a unos requerimientos específicos a satisfacer.
- Presentación del Prototipo (Formularios F06)
  - La presentación del prototipo al cliente se hace el día, hora y lugar acordados previamente.
  - El prototipo es presentado y explicado ampliamente por el gerente del proyecto al cliente con el fin de conocer si cumple con las expectativas requeridas.
  - El prototipo se considera aprobado si un 75% o más de los requerimientos deseados son cubiertos satisfactoriamente a consideración del cliente; de lo contrario, se considera rechazado.
  - En caso de ser aprobado pero tenerse observaciones menores, estas deben ser registradas en el documento.

- En caso de ser rechazado, se toma nota de los requerimientos que se consideran aprobados y la razón por la que se rechazaron los requerimientos no aceptados.

- Si es aprobado el prototipo
  - Actualización de requerimientos (Formularios F07)
    - Este documento lo que contiene son los requerimientos aprobados, y que serán diseñados y codificados en la iteración.
    - En caso de existir pequeñas observaciones obtenidas con el prototipo se consignarán estos cambios menores.
    - En este documento queda la versión definitiva de los requerimientos a implementar en la iteración.
    - Además se adjuntan los casos de uso definitivos.
  - Diseño (Formularios F08)
    - Se genera el documento de diseño para los requerimientos de la iteración en ejecución.
    - El listado de requerimientos está actualizado en el documento "Actualización de requerimientos" de la actual iteración y tiene los casos de uso asociados adjuntos.
  - Codificación (Formularios F09)
    - Este formulario y sus adjuntos dejan registrado el código generado y el cumplimiento de los estándares de calidad definidos.
  - Pruebas (Formularios F10)
    - Para cada caso de uso de la iteración se genera un caso de prueba que sirva para comprobar la correctitud de la aplicación generada.
    - Los casos de prueba tienen la numeración P01, P02, P03, etc., y este identificador es único por proyecto.
  - Presentación de versión funcional (Formularios F11)
    - La presentación al cliente se hace el día, hora y lugar acordados previamente.
    - La aplicación es presentada y explicada ampliamente por el gerente del proyecto al cliente con el fin de conocer si cumple con las expectativas requeridas.
    - Además, en este momento se hace una revisión del monto en colones invertidos a la fecha en el proyecto con el fin de que el cliente determine si desea y puede iterar más con otros requerimientos.
  - Si la aplicación es aprobada
    - Ir a 3 "Mientras haya requerimientos a considerar"
  - Si no es aprobada
    - Ir a 3.4.3 "Codificación"

- Si no es aprobado
  - Ir a 3.1 "Toma detallada de Requerimientos"

Cierre del proyecto (Formularios F12)

- A partir de la fecha del último requerimiento analizado (Paso 4) se da un plazo de un mes calendario como garantía, y si no hay reclamos, se procede al acta de cierre del proyecto.
- Este documento da por cerrado el proyecto, sin reclamos posteriores.
- Es firmado, al menos, por el gerente del proyecto y el patrocinador.

Conclusiones

La investigación demostró que existen muchos modelos iterativos, sobre todo con el surgimiento de las metodologías ágiles, lo que falta es más literatura sobre la aplicación de estas metodologías en microempresas, normalmente está muy bien documentado de empresas medianas y grandes.

Se logró listar el número de métricas deseadas. Existen muchos tipos de métricas y su selección es compleja pues no hay muchos criterios de cuándo y por qué escoger una u otra. Además, hay que tener cuidado de que la selección de varias métricas no vaya contra el mismo costo-beneficio.

Se realizó un estudio concienzudo del PMBOK del PMI y de la ISO/IEC 25000, y luego se inventariaron las mejores prácticas. Se debe tener cuidado en las microempresas que la selección de técnicas y herramientas para la administración de proyectos no debe crear un costo administrativo tan alto que la microempresa no lo pueda absorber.

Para el cumplimiento pleno de este objetivo se encontró un problema, por el tamaño de este tipo empresas en Costa Rica (5 o menos trabajadores) normalmente no están bien organizadas y muy pocas están agremiadas. Son sobre todo las pequeñas, medianas y grandes empresas, las que se agremian en organizaciones como CAMTIC o se inscriben formalmente. Por lo anterior, conseguir la muestra para la encuesta fue complicado y difícil: se logró, sobre todo, usando redes de amigos y conocidos.

La metodología propuesta contiene los procedimientos e instrumentos necesarios para utilizarla. La presente investigación demuestra que para microempresas de desarrollo de *software* el uso de este tipo de metodologías es una gran oportunidad de negocio, dado que disminuyen los gastos administrativos que tanto afectan a las microempresas por un tema de economías de escala.

Trabajo futuro

Sería muy provechoso realizar una investigación que se centre solo en la comparación de una mayor cantidad de modelos iterativos, lo cual podría determinar más opciones a considerar.

La aplicación de la metodología propuesta generará estadísticas de su uso y, por tanto, luego de varias iteraciones se podría ver la conveniencia de continuar con las métricas definidas en la presente metodología o cambiarlas por otras.

Si bien para esta metodología se eligió el plan de inspección y el plan de auditoría, sería conveniente que luego de varias aplicaciones de la metodología se valorara la inclusión de alguna práctica más que venga a aportar y a no sobrecargar el proceso. Por ejemplo, la aplicación de alguna técnica de Costo/Beneficio a la hora de introducir nuevas características al *software*.

Reconocimientos

Se agradece a todos los microempresarios que de forma anónima colaboraron completando la encuesta que sirvió como insumo para la presente investigación.

Bibliografía

CAMTIC. (28 de Febrero de 2006). *Estado Nacional del Sector del Software*. Recuperado el 20 de Agosto de 2011, de http://www.camtic.org/ES/camtic/estudios/.

Highsmith, J. (2001). *Manifesto for Agile Software Development*. Recuperado el 04 abril de 2012, de http://www.agilemanifesto.org/.

ISO/IEC 12207. (2008). Systems and *software* engineering Software life cycle processes. EEUU.

ISO/IEC 25000. (2005). *Requisitos y Evaluación de Calidad de Productos de Software (SQuaRE).* Recuperado el 28 de marzo de 2011, de http://webstore.iec.ch/preview/info_isoiec25000%7Bed1.0%7Den.pdf

ISO/IEC 9126. (2001). Software engineering - Product quality. Suiza.

ISO/IEC TR 9126-3. (Marzo de 2011). Software engineering – product quality, part 3 Internal Metrics. Canadá.

Ministerio de Economía Industria y Comercio. (Marzo de 2009). Digepyme - Empresas según tamaño. San José, San José, Costa Rica.

Piattini, M. y. (2007). Calidad de Sistemas Informáticos. En M. y. Piattini, *Calidad de Sistemas Informáticos* (pág. 206). México: AlfaOmega.

Sommerville, I. (2006). *Ingeniería del Software.* España: Pearson.

Soto, L. (2010). *Modelo Incremental*. Recuperado el 27 de agosto de 2011, de http://www.mitecnologico.com/Main/ModeloIncremental