Metodologías para el desarrollo de software en PYMES

Marcelo A. Colombani, Martín M. Pérez, Cristian D. Pacífico

Lenguajes de Programación e Ingeniería en Software / Facultad de Ciencias de la Administración / Universidad Nacional de Entre Ríos (UNER)
Monseñor Tavella 1424, (0345) 4231400
marcol@fcad.uner.edu.ar, marper@fcad.uner.edu.ar, cripac@fcad.uner.edu.ar

Resumen

Uno de los principales objetivos de este proyecto de investigación en el cual está inmerso el presente trabajo, es especificar modelos y guías de certificación para productos de Software Libre y procesos de ciclo de vida que involucren plataformas open source.

Para alcanzar los objetivos propuestos se abordaron los diferentes modelos de proceso de software, desarrollando sus características, bondades y deficiencias, con el fin de detectar qué metodología se encuadra o adapta mejor al escenario de las Pequeñas y Medianas Empresas productoras de software o empresas que en su estructura albergan un equipo de desarrollo no demasiado numeroso.

Se realizará un relevamiento en PYMES en la Región de Salto Grande, que tengan grupos de trabajos dedicados al proceso de ciclo de vida de software.

Palabras clave: PYMES, proceso de desarrollo de software, modelos de proceso de software, modelos y guías de certificación.

Contexto

El trabajo forma parte de la línea de investigación del Proyecto de Investigación y Desarrollo (PID) denominado "*Modelos de Certificación para los procesos de Ingeniería de Software en productos desarrollados con Lenguajes de Programación Open Source: relevamiento y aplicación en PYMES de la zona de influencia de la UNER Concordia*" presentado en la Fac. Cs Administración de la UNER.

El PID busca establecer las bases del ámbito científico-académico para el desarrollo y redacción de guías y normas para la validación y certificación de productos y procesos de Software Libre y Tecnologías Abiertas.

Dentro del marco de la presente línea de investigación se está desarrollando una tesis de maestría, en la carrera *Maestría en Sistemas de Información* dictada en la Fac. Cs Administración, UNER en el área de Ingeniería del Software.

Introducción

El software de computadoras se ha convertido en un producto fundamental en todas las empresas y organizaciones, es un factor clave que diferencia a las que lo posean. Actualmente existen sistemas en casi todo tipo de empresa y ellas poseen una gran variedad de los mismos, como ser: transporte, gobierno, medicina, agro, telecomunicaciones, herramientas de oficina, entre tantos, y la lista se hace interminable [Som15].

El impacto del software en nuestra sociedad y en la cultura continúa siendo profundo. Al mismo tiempo que crece su importancia, la comunidad del software trata continuamente de desarrollar tecnologías que hagan más sencilla, rápida y menos costosa la construcción de programas de computadora de alta calidad [PM15].

La utilización masiva de las computadoras y por ende de los sistemas en los negocios ha generado una cultura de "sistematización de los procesos". Sumado a esto, las herramientas de desarrollo han permitido la generación de software de una manera mucho más fácil, rápida y menos costosa, lo que ha sustentado la cultura de sistematización. Así se logró insertar sistemas software en empresas o negocios pequeños, que de otra manera no hubiesen podido encarar un proyecto de este tipo.

A raíz de esto cada vez se requieren más y mejores sistemas y en tiempos considerablemente menores, pero la construcción de software no es tarea fácil por varios motivos. Normalmente se requiere que el software cubra todas las necesidades de la empresa, que se desarrolle en un tiempo limitado y, además, existen innumerables metodologías de desarrollo de software que sustentan este proceso de construcción. Existen diferentes propuestas tradicionales [PM15], [Som15] que se direccionan especialmente en el control del proceso, definiendo rigurosamente las actividades a desarrollar, los resultados a producir, las notaciones y las herramientas a utilizar. Como alternativa para mejorar el desarrollo se pueden incluir más actividades [Som15], más diagramas [Som15], pero esto sólo puede llevar a un desarrollo más complejo y a sobrecargar al equipo de trabajo.

Una alternativa a este proceso de desarrollo se basa en centrarse en otros factores, como son el humano o el producto software a construir. Esta idea se conoce como Metodologías Ágiles [Bee09], que dan mayor valor al individuo, mayor participación al cliente, y tienen como objetivo realizar entregas incrementales del software.

Estos métodos son utilizados en proyectos en los cuales los requerimientos o expectativas del clientes/usuario son cambiantes o difusos. También suele exigirse un tiempo de entrega reducido con un producto que respete estándares de calidad [Som15].

En experiencias en desarrollo de software surge una problemática recurrente: la falta de adecuación de los requerimientos establecidos de un software a desarrollar con las expectativas que el cliente/usuario tiene del mismo [ISO08], [Som15]. Esto tiene su origen en que los métodos tradicionales que exigen que los requerimientos sean establecidos rigurosamente; en contraposición con las expectativas del cliente/usuario las cuales son dinámicas a lo largo del proyecto. De hecho, tales expectativas se modifican a medida que se visualizan las potencialidades del software desarrollado en las entregas parciales del mismo; entregas que por otro lado, son necesarias para verificar tempranamente el producto de software y renovar el compromiso hacia el proyecto [PM15], [Bee09], [Wel09]. A medida que el proyecto se desarrolla, es posible que otros interesados (*stakeholders*) intervengan en el mismo, y vean potencialidades en el dominio de aplicación no contempladas en los requerimientos. En este escenario,

es habitual que la empresa cliente vea que algunas expectativas no se cumplen con la versión final del producto; por más que éste cumpla plenamente los requerimientos. Por ende, resulta necesario para la aceptación del producto (y la impresión positiva del equipo de desarrollo), reflejar estas expectativas no cubiertas en los requerimientos de alguna manera, para que se materialicen en la versión final del producto.

Esta problemática ha llevado a la necesidad de utilizar diferentes modelos de proceso de software adaptados y combinados de diferente manera para lograr un proceso óptimo sin perder de vista uno de los objetivos que es lograr un software de calidad en todos los aspectos, y confiable en todas sus dimensiones tanto en disponibilidad, fiabilidad, seguridad y protección.

Modelos de proceso

Los modelos son simplificaciones de los procesos de software; por lo tanto, un modelo de procesos del software es una simplificación o abstracción de un proceso real [Som15]. Se puede definir a un modelo de procesos del software como una representación abstracta de alto nivel de un proceso de software [Som15].

Revisando los enfoques propuestos ([Som15], [PM15], [Wells09], [JBR99]) es posible identificar una variedad de modelos de desarrollo de software, como así también modelos similares mencionados de diferente manera, pero realizando un análisis minucioso de su conformación y manera de organizar el proceso, éstos realmente reflejan el mismo funcionamiento.

Por lo anteriormente expuesto, en este trabajo se enumeran y describen los diferentes modelos que servirán de guía para clarificar qué modelo de desarrollo puede ajustarse mejor a las Pequeñas y Medianas Empresas de nuestra zona (Concordia, Entre Ríos, Argentina), las cuales no están ajenas al problema de los requerimientos confusos, incompletos y cambiantes.

Cada modelo es una descripción de un proceso software que se presenta desde una perspectiva particular [Som15], describiendo una sucesión de fases y una relación entre ellas. Según las fases y el modo en que se produzca esa relación, tenemos diferentes modelos de proceso. Es por ello que un modelo es más adecuado que otro para desarrollar un proyecto dependiendo de un conjunto de características de éste [Som15], [PM15]. Análogamente, los artefactos entregables de cada etapa varían conforme su tasa de cambio y validación; en el caso que se acepte la dinámica de requerimientos como una suposición presente en el modelo, estos cambios deben documentarse y trazarse adecuadamente, para justificar posibles desviaciones no contemplada en el proyecto.

Otra faceta que no debe dejarse de lado, es el aseguramiento de la calidad en el proceso y producto de software; en particular si es un requisito para certificar o validar conforme a estándares internacionales de calidad [Flo13]. Como equipo (empresa) de desarrollo resulta adecuado instaurar buenas prácticas en el proceso de desarrollo conforme a guías de certificación aceptadas (ISO/IEC 9001-90003 [ISO15, ISO14], ISO 15504 SPICE [ISO04], ISO/IEC 15288 [ISO08], ISO/IEC/IEEE 12207 y complementarias [ISO08b, ISO11c]). Como empresa usuaria o productora de software sería importante verificar la calidad del producto conforme a esquemas de

certificaciones (ISO/IEC 25000 [ISO07, ISO11, ISO11b]), permitiendo como efecto deseado tener ventajas competitivas y comerciales.

Resultados Parciales

En base a la investigación realizada se puede asegurar que los cambios son inevitables en todos los proyectos de software, sean éstos pequeños, medianos o grandes. Los requerimientos del sistema cambian cuando el negocio donde está inserto el sistema responde a las presiones externas.

Las prioridades de gestión cambian y, cuando se dispone de nuevas tecnologías, cambian los diseños y la implementación. Esto significa que el proceso del software no es un proceso único; más bien, las actividades del proceso se repiten regularmente conforme el sistema se rehace en respuesta a peticiones de cambios.

Como surge del estudio de las diferentes metodologías, el problema de cada una de ellas siempre gira alrededor de la falta de requerimientos, la gestión de la documentación y la falta de comprensión del cliente con respecto a los requerimientos. Todos estos factores llevan a requerimientos confusos, incompletos y cambiantes.

Además de la metodología que se utilice, será importante definir una buena estructura de documentación, a fin de poder reflejar los requerimientos que dieron origen al sistema, los diferentes componentes del sistema software, sus relaciones y sus efectos en las modificaciones del mismo.

Es importante también detallar una guía de buenas prácticas que aseguren la calidad en el proceso de desarrollo y en el producto software final, siguiendo los esquemas de normas internacionales. En particular se debe puntualizar detalladamente cómo los cambios en los requerimientos impactan en el proceso y en la calidad del producto, y de qué forma medir y registrar dichos efectos, para lograr mejorar continuamente el modelo de ciclo de vida.

Líneas de Investigación, Desarrollo e Innovación

Como trabajo futuro será importante continuar el estudio de los Modelos de Proceso de Software y describir pasos metodológicos para llevar adelante un proyecto de software exitoso utilizando alguna combinación de diferentes metodologías tradicionales y ágiles, ya que cada una posee ventajas y desventajas, logrando así un enfoque híbrido en el cual los métodos ágiles incorporen algunas técnicas de planificación y documentación pero sin perder de vista las iteraciones y los principios del modelo ágil.

Resultados y Objetivos

Este proyecto de investigación se encuentra en su fase de iniciación y, entre los objetivos buscados podemos remarcar la identificación de diversas metodologías de desarrollo de software que pueda ser utilizada en empresas de nuestra zona (Concordia, Entre Ríos, Argentina), que cuenten con un grupo de desarrolladores reducido o en muchos casos unipersonal, y consideren llevar adelante un proceso de desarrollo organizado y de calidad.
Por otro lado, se pretende redactar guías y manuales para la aplicación de tales métodos y que permitan guiar en la adopción de normas y certificación de

estándares del proceso de desarrollo y del producto final de software.

Formación de Recursos Humanos

Como parte de este proyecto de investigación se espera completar una tesis de maestría en la Maestría en Sistemas de Información. Los integrantes están dirigiendo tesis de grado de la Facultad Regional de Concordia, Universidad Tecnológica Nacional y dirigiendo una Tesis de Maestría en la Facultad de Ciencias de la Administración, Universidad Nacional de Entre Ríos.

Referencias

[Bee09] Beedle, M., "Principles behind the Agile Manifesto". [online]. *http:// www.agilemanifesto.org/principles.html*

[Flo13] Florian Schneider, Brian Berenbach, "A Literature Survey on International Standards for Systems Requirements Engineering", Procedia Computer Science, Volume 16, 2013, Pages 796-805.

[ISO04] ISO, "ISO/IEC 15504 Information technology - Process assessment" ISO/IEC, Nov, 2004.

[ISO07] ISO, "ISO/IEC 25030 - Software engineering - Software product Quality Requirements and Evaluation (SQuaRE) - Quality requirements," Jun. 2007.

[ISO08] ISO, "ISO/IEC 15288:2008 - Systems and software engineering - System life cycle processes." Mar. 2008.

[ISO08b] ISO, "ISO/IEC/IEEE 12207:08 Systems and software engineering - Software life cycle processes," Mar. 2008.

[ISO10] ISO, "ISO/IEC/IEEE 24765:10 - Systems and software engineering - Vocabulary", Dec. 2010.

[ISO11] ISO, "ISO/IEC 25040 – Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) - Evaluation process." Feb. 2011.

[ISO11b] ISO, "ISO/IEC 25010 –Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models," Mar. 2011.

[ISO11c] ISO, "ISO/IEC/IEEE 29148: 2011 - Systems and software engineering - Life cycle processes - Requirements engineering," Nov. 2011.

[ISO14] ISO, "ISO/IEC 90003:2014 - Software engineering - Guidelines for the application of ISO 9001:2008 to computer software" Dec, 2014.

[ISO15] ISO, "ISO/IEC 9001:2015 - Quality management systems - Requirements" Sep, 2015.

[JBR99] I. Jacobson, G. Booch, and J. Rumbaugh. "El proceso unificado de desarrollo de software". Addison-Wesley, primera edición, 1999.

[Pal07] Juan Palacio. "Flexibilidad con scrum". Safe Creative, 2007.

[PM14] R.S. Pressman, B.Maxim. "Software Engineering: A Practitioner's Approach". Mc Graw Hill, 7ma Ed, 2014.

[Som15] I. Sommerville. "Software Engineering" (10th Ed). Pearson Addison Wesley, 2015.

[Wel09] D. Wells. "Extreme programming: A gentle introduction". Tech. Report, 2009. [online] *http://www.extremeprogramming.org*