Evaluación de Metodologías Ágiles para Desarrollo de Software

Karla Mendes Calo, Elsa Estevez, Pablo Fillottrani[1]
Dpto. de Ciencias e Ingeniería de la Computación, Universidad Nacional del Sur
Av.Alem 1253, (8000) Bahia Blanca, Argentina
+54 291 459-5135
[1]Comisión de Investigaciones Científicas de la Provincia de Buenos Aires, Argentina
{kmca,ece,prf}@cs.uns.edu.ar

RESUMEN

Las metodologías de desarrollo ágiles permiten a los grupos de desarrollo producir software de manera iterativa e incremental. De esta manera facilitan la introducción de cambios a los requerimientos durante el proceso de desarrollo. El objetivo de este trabajo es presentar una línea de investigación sobre el uso de metodologías ágiles, focalizándose en el estudio de un marco de evaluación para dichas metodologías. Se espera que el marco de evaluación propuesto permita evaluar en qué medida las metodologías de desarrollo cumplen con los postulados y principios declarados por el Manifiesto Ágil. Asimismo, se pretende que el marco sirva como herramienta para facilitar la toma de decisión al momento de seleccionar una metodología.

***Palabras Claves*:** Ingeniería de Software, Procesos de Desarrollo, Metodologías Ágiles.

CONTEXTO

Este trabajo de investigación se desarrolla en el Departamento de Ciencias e Ingeniería de la Computación de la Universidad Nacional del Sur (UNS). En particular, como parte de las tareas que se realizan en el Laboratorio de Investigación y Desarrollo en Ingeniería de Software y Sistemas de Información (LISSI). El proyecto se financia parcialmente con fondos asignados por la UNS a proyectos de investigación - PGI 24/N021.

1. INTRODUCCIÓN

La Ingeniería del Software se caracteriza por establecer marcos de trabajo en los cuales los procesos de desarrollo de software adquieren relevancia. Dichos procesos definen artefactos de desarrollo, documentación a producir, herramientas y notaciones a ser utilizadas, y actividades a realizar y su orden de ejecución, entre otras definiciones. Como resultado, los procesos generan gran cantidad de documentación con el objetivo de facilitar compartir el conocimiento entre los integrantes del equipo de trabajo. Si bien existen varios procesos de desarrollo – Proceso Unificado [1], Proceso V [2], etc., la mayoría de estos procesos se derivan del Modelo de Cascada propuesto por Boehm [3]. Estos procesos, denominados tradicionales, han demostrado ser efectivos, particularmente en lo que respecta a la administración de recursos a utilizar y a la planificación de los tiempos de desarrollo, en proyectos de gran tamaño con requerimientos estables. Sin embargo, debido a entornos comerciales más competitivos, conducidos por la rapidez para producir y entregar servicios [4], el enfoque propuesto por estos métodos no resulta el más adecuado. Usualmente, estos nuevos entornos se caracterizan por el desarrollo de proyectos donde los requerimientos del sistema son desconocidos, inestables o muy cambiantes, los tiempos de desarrollo se deben reducir drásticamente, y al mismo tiempo, se espera la producción de un producto de alta calidad, que a su vez garantice mínimo riesgo ante la necesidad de introducir cambios a los requerimientos.

Como alternativa a los métodos tradicionales de desarrollo, surgen las Metodologías Ágiles. Estas metodologías están

especialmente indicadas para productos cuya definición detallada es difícil de obtener desde el comienzo, o que si se definiera, tendría menor valor que si el producto se construye con una retro-alimentación continua durante el proceso de desarrollo. Tal el caso de sistemas donde los usuarios no reconocen sus necesidades reales sin antes ver un componente funcionando. La característica de estos productos y usuarios, sumado a un entorno de cambio acelerado, hacen de los métodos ágiles una alternativa viable para el mejor entendimiento de los requerimientos. Manteniendo prácticas esenciales de las metodologías tradicionales, las metodologías ágiles se centran en otras dimensiones del proyecto, como por ejemplo: intentan trabajar con un mínimo de documentación necesaria, reemplazándola por la comunicación directa y cara a cara entre todos los integrantes del equipo; la colaboración activa de los usuarios durante todas las etapas del proceso de desarrollo; el desarrollo incremental del software con iteraciones muy cortas y que entregan una solución a medida; y la reducción drástica de los tiempos de desarrollo pero a su vez manteniendo una alta calidad del producto.

El objetivo de este trabajo es presentar una línea de investigación sobre el uso de metodologías ágiles para el desarrollo de software, focalizándose en particular en el estudio de un marco de evaluación para dichas metodologías que permita evaluar en qué medida las metodologías cumplen con los postulados y principios declarados por el Manifiesto Ágil. Asimismo, se pretende que el marco de evaluación brinde una herramienta para facilitar la toma de decisión al momento de seleccionar una de estas metodologías.

2. METODOS ÁGILES DE DESARROLLO

Comenzando con la presentación del Manifiesto Ágil (Sección 2.1), esta sección presenta las metodologías ágiles más comúnmente utilizadas (Sección 2.2).

2.1 Manifiesto Ágil

El Manifiesto Ágil es el documento que define los principios rectores de las metodologías ágiles de desarrollo de software. Este documento fue creado por un grupo de académicos y expertos de la industria del software reunidos en Utha, Estados Unidos en febrero de 2001. El objetivo de la reunión era discutir los valores y principios que facilitarían desarrollar software más rápidamente y respondiendo a los cambios que surjan a lo largo del proyecto. La idea era ofrecer una alternativa a los procesos de desarrollo tradicionales, caracterizados por su rigidez y dirigidos por la documentación que se genera en cada etapa. Como resultado de esta reunión, se creó The Agile Alliance [5], una organización, sin fines de lucro dedicada a promover los conceptos relacionados con el desarrollo ágil de software y ayudar a las organizaciones a adoptar dichos conceptos. El resultado de esta reunión fue un documento conocido como el Manifiesto Ágil [6]. El Manifiesto Ágil incluye cuatro postulados y una serie de principios asociados. Sus postulados son:

1) *Valorar al individuo y a las interacciones del equipo de desarrollo por encima del proceso y las herramientas*. Este postulado enuncia que las personas son componentes primordiales en cualquier desarrollo [7]. Tres premisas sustentan este postulado: a) los integrantes del equipo son el factor principal de éxito; b) es más importante construir el equipo de trabajo que construir el entorno; y c) es mejor crear el equipo y que éste configure el entorno en base a sus necesidades.
2) *Valorar el desarrollo de software que funcione por sobre una documentación exhaustiva*. El postulado se basa en la premisa que los documentos no pueden sustituir ni ofrecer el valor agregado que se logra con la comunicación directa entre las personas a través de la interacción con los prototipos.

3) *Valorar la colaboración con el cliente por sobre la negociación contractual*. En el desarrollo ágil el cliente se integra y colabora con el equipo de trabajo. Se asume que el contrato en sí, no aporta valor al producto, sino que es sólo un formalismo que establece líneas de responsabilidad entre las partes.
4) *Valorar la respuesta al cambio por sobre el seguimiento de un plan*. La evolución rápida y continua deben ser factores inherentes al proceso de desarrollo. por sobre la capacidad de seguimiento y aseguramiento de planes pre-establecidos.

Los postulados descriptos anteriormente, inspiraron los doce principios del Manifiesto Ágil. Son las características que diferencian un proceso ágil de uno tradicional. Si bien los dos primeros son principios generales, los demás tienen que ver con los procesos a seguir, el equipo de desarrollo y su organización: (1) la prioridad es satisfacer al cliente mediante entregas de software tempranas y continuas; (2) los cambios en los requerimientos son aceptados; (3) software que funcione se entrega frecuentemente, con el menor intervalo posible entre entregas; (4) el cliente y los desarrolladores deben trabajar juntos a lo largo del proyecto; (5) el proyecto se construye en base a individuos motivados; (6) el dialogo cara a cara es el método mas eficiente y efectivo para comunicar información dentro del equipo; (7) el software que funcione es la medida principal del progreso; (8) los procesos ágiles promueven el desarrollo sostenido; (9) la atención continua a la excelencia técnica y al buen diseño mejora la agilidad; (10) la simplicidad es esencial; (11) las mejores arquitecturas, requerimientos y diseños surgen de equipos auto-organizados, y (12) el equipo reflexiona en cómo ser más efectivos, y ajusta su comportamiento en consecuencia.

2.2 Metodologías Ágiles

El ciclo de desarrollo que aplican las Metodologías Ágiles es iterativo e incremental. Como se referencia en varios trabajos relacionados, el factor humano es fundamental para el éxito de proyecto [7][8][9]. Este modelo permite entregar el software en partes pequeñas y utilizables, conocidas como incrementos. Estas metodologías aportan nuevos métodos de trabajo que requieren una cantidad de procesos, que no se desgastan con cuestiones administrativas – tales como planificación, control, documentación - ni tampoco defienden la postura extremista de la total falta de proceso. Debido a que se tiene conciencia que los cambios indefectiblemente se producirán, el objetivo es reducir el costo de rehacer parte del producto por los cambios introducidos. Se intenta guiar el desarrollo hacia un objetivo que puede no permanecer constante en el tiempo a medida que aumenta el conocimiento de la aplicación a ser construida. Cada iteración se puede considerar como un mini-proyecto en el que las actividades de análisis de requerimiento, diseño, implementación y testing son llevadas a cabo con el fin de producir un subconjunto del sistema final. El proceso se repite varias veces produciendo un nuevo incremento en cada ciclo. Dicho proceso concluye cuando se haya elaborado el producto completo. Si bien todas las metodologías ágiles adoptan este ciclo, cada una presenta sus propias características. A continuación se presentan las más utilizadas.

Programación Extrema (XP) [10]– Fue la que le dio impulso al movimiento actual de metodologías ágiles. Entre los principios más importantes de esta metodología, se pueden mencionar: en cada iteración, se determina el alcance de la próxima iteración combinando prioridades del negocio y estimaciones técnicas, definiéndose la estrategia de planeamiento durante el proceso de desarrollo. Las entregas son frecuentes y continuas. Cada versión liberada debe ponerse en producción rápidamente. El desarrollo es guiado a través de historias de usuario simples. El cliente o un representante del cliente son integrados al equipo de desarrollo, quien está disponible full-

time para responder preguntas del equipo relacionadas con las reglas de negocio. El cliente es responsable por escribir los casos de prueba, demostrando que la funcionalidad está finalizada. Recomienda que el desarrollo de las funciones del producto sea realizado por dos personas en el mismo puesto – programación por pares. Cuando se finaliza la implementación de una funcionalidad, se deben corregir todos los defectos encontrados.

Scrum [11] – Indicada para proyectos con alto ratio de cambio de requerimientos, su principal característica es la definición de *sprints* – cada una de las iteraciones del proceso con una duración máxima de 30 días. El resultado de cada sprint es un incremento ejecutable que se muestra al cliente. Otra característica importante son las reuniones diarias que se llevan a cabo a lo largo del proyecto. Dichas reuniones no requieren más de 15 minutos del equipo de desarrollo y su objetivo son la coordinación e integración del producto a entregar. Al principio del proyecto se define el *Product Backlog*, que contiene todos los requerimientos funcionales y no funcionales que deberá satisfacer el sistema a construir. Los mismos estarán especificados de acuerdo a los procesos definidos en la organización, ya sea mediante: features, casos de uso, diagramas de flujo de datos, incidentes, tareas, etc. El *Product Backlog* será definido durante reuniones de planeamiento con los stakeholders. A partir de ahí se definirán las iteraciones, conocidas como s*print*, en las que se irá desarrollando la aplicación evolutivamente. Cada Sprint tendrá su propio *Sprint Backlog* que será un subconjunto del *Product Backlog* con los requerimientos a ser construidos en dicho *sprint.*

Crystal Methodologies [12] – Se trata de un conjunto de metodologías caracterizadas por la valoración de las personas que componen el equipo de trabajo y la reducción al máximo del número de artefactos producidos. Su nombre se debe a las facetas de una gema: cada faceta es otra versión del proceso, y todas se sitúan en torno a un núcleo idéntico. Enfatiza en esfuerzos para mejorar las habilidades de los integrantes del equipo y para definir políticas de trabajo. Se estableció una clasificación por colores en función del número de integrantes del equipo, por ejemplo Crystal Clear corresponde a equipos de 3 a 8 integrantes, Cristal Yellow en equipos de 10 a 20 integrantes, Crystal Orange en equipos de 25 a 50 integrantes, y así sucesivamente hasta azul. Los principios destacados de la metodología incluyen: Entregas Frecuentes - entregar software a los clientes con una frecuencia diaria, semanal o mensual dependiendo del proyecto; Comunicación - promueve el uso de espacios destinados a que todos puedan ver el avance del trabajo.

Dynamic Systems Development Method (DSDM) [13] – Cumple con las características generales de definir un proceso iterativo e incremental. Propone cinco etapas de desarrollo: (1) Estudio de Viabilidad, (2) Estudio del Negocio, (3) Modelado Funcional, (4) Diseño y Construcción y (5) Implementación. La iteración se produce en las tres últimas etapas, sin embargo prevé retro-alimentación en todas.

Adaptive Software Development (ASD) [14] – Es un proceso iterativo, tolerante a cambios y orientado a los componentes de software. Define tres etapas para el ciclo de vida: (1) Especulación – se inicia el proyecto y se planifican las características del software; (2) Colaboración – se desarrolla el producto; y (3) Aprendizaje – se revisa la calidad del producto y se entrega al cliente. La revisión tiene como objetivo aprender de los errores cometidos y volver a iniciar el ciclo de desarrollo.

Feature-Driven Development (FDD) [15] – Define un proceso iterativo, con iteraciones cortas de dos semanas como máximo. El ciclo de vida consta de cinco pasos: (1) Desarrollo de un modelo global, (2) Construcción de una lista de funcionalidades, (3) Planeación por funcionalidad, (4) Diseño por funcionalidad y (5) Construcción por funcionalidad.

3. RESULTADOS OBTENIDOS/ESPERADOS

Si bien la mayoría de las metodologías ágiles satisfacen los postulados y principios del Manifiesto Ágil, no todas lo hacen de la misma manera. Más aún, el proceso de seleccionar la metodología que mejor se adapta a un problema en particular, se torna dificultoso. Este trabajo de investigación tiene como objetivo definir un marco de evaluación de metodologías ágiles de desarrollo, analizar cómo las mismas cumplen con los postulados y principios del Manifiesto Ágil, así como con otros aspectos del proceso de desarrollo de software. Se espera que el marco propuesto provea información que pueda ser utilizada como input al momento de seleccionar la metodología de desarrollo ágil más adecuada para cada proyecto.

Hasta la fecha, las tareas de investigación realizadas han permitido definir un marco de trabajo que permite evaluar en forma cuantitativa en qué medida las metodologías de desarrollo satisfacen los cuatro postulados básicos del Manifiesto Ágil [16].

Como trabajos futuros de investigación se prevé: extender el framework para considerar los principios del manifiesto ágil y otros aspectos de los procesos de desarrollo; la aplicación del framework a más metodologías; la derivación de recomendaciones para asistir a la toma de decisiones para seleccionar una metodología, teniendo en cuenta aspectos como por ejemplo el tipo de proyecto, reglas de negocio inherentes al desarrollo, personas y tecnología, entre otras.

4. FORMACIÓN DE RECURSOS HUMANOS

Los resultados de investigación serán utilizados para presentar una tesis de Magister en Ciencias de la Computación. Asimismo, servirán para generar courseware para materias optativas a dictarse en el Departamento de Ciencias e Ingeniería de la Computación de la UNS.

5. BIBLIOGRAFÍA

[1] Jacobson, I., et al, *The Unified Software Development Process*, Addison-Wesley.

[2] IABG, *The V-Model*, http://www.v-modell.iabg.de/

[3] Boehm, B., *Software Engineering*, IEEE Transactions on Computers, 1976.

[4] Ridderstrale, J., et al, *Business*:*Talent Makes Capital Dance*. Prentice Hall, EEUU, 2000.

[5] Agile Alliance, http://www.agilealliance.org/.

[6] Beck, K., et.al, *Manifesto for Agile Software Development*, http://agilemanifesto.org/.

[7] Cockburn, A., *Characterizing People as Non-Linear, First-Order Components in Software Development*, octubre 1999.

[8] Mc Connell, S., *Professional Software Development: Shorter Schedules, Higher Quality Products, More Successful Projects, Enhanced Careers*, Addison Wesley, 2003.

[9] Glass, R. L., *Facts and Fallacies of Software Engineering*, 2002.

[10] Beck, K., *Extreme Programming Explained.* Embrace Change, Pearson Education, 1999.

[11] *Scrum Alliance*, http://www.scrumaliance.org

[12] Cockburn, A., *Crystal Clear a Human-Powered Methodology for Small Teams*.

[13] Stapleton J., *DSDM Dynamic Systems Development Method: The Method in Practice*. Addison-Wesley, 1997.

[14] Highsmith J., Orr K., *Adaptive Software Development. A Collaborative Approach to Managing Complex Systems*, 2000.

[15] *Feature Driven Development*, http://www.featuredrivendevelopment.com

[16] Mendes Calo, K., Estevez, E. Fillottrani, P., "*Un Framework para Evaluación de Metodologías Ágiles*", CACIC2010, 2009.