14 del 2 al 5 de Diciembre de 2008
CONVENCION CIENTIFICA
DE INGENIERÍA Y ARQUITECTURA

44 ANIVERSARIO
cujae
2008

METODOLOGÍAS ÁGILES.¿CÓMO DESARROLLO UTILIZANDO XP?

Autor: Ing.Danay Pérez Ramírez
Coautores: Ing.Yoanna Oliveros Guntín, Ing.Yanniel Alvarez Alonso, Lic.Jorge Coello Mena

Para el desarrollo de software en la actualidad existen diversas metodologías, las que se agrupan en ágiles y tradicionales. Las metodologías ágiles proponen una forma de desarrollo de software diferente a las metodologías tradicionales. Éstas promueven las relaciones interpersonales, una estrecha colaboración entre el cliente y el equipo de desarrollo, son adaptativas y se enfocan en las personas y los resultados obtenidos. Entre las metodologías ágiles más utilizadas se encuentra Extreme Programming (XP).

En el documento se abordan las características de las metodologías ágiles; mencionándose brevemente aspectos importantes de alguna de éstas como: Scrum, Crystal Methodologies, Dynamic Systems Development Method (DSDM), Adaptive Software Development (ASD) y Feature-Driven Development (FDD), además se profundiza en XP. El ciclo de vida ideal de la metodología ágil XP es descrito y se detallan los roles y prácticas de ésta. Con el objetivo de facilitar el trabajo aplicando XP se propone una guía detallada, la cual describe paso a paso cómo se debe aplicar XP en el desarrollo de software.

Palabras Claves: ágil, aplicación, desarrollo, guía, metodología, XP.

INTRODUCCIÓN

La industria del software ha adquirido gran auge en las últimas décadas. Cada día son más las demandas de diferentes tipos de software, menos el tiempo disponible para desarrollarlos y más cambiantes los requerimientos iniciales de éstos. Para hacer frente a esta demanda es necesario producir software de calidad en el menor tiempo posible.

Como toda industria, la de software, también tiene normas y pasos a seguir para ayudar a los desarrolladores en la producción: las metodologías de desarrollo de software.
A través de los años han surgido diferentes metodologías. Algunas hacen más énfasis en el control del proceso, estableciendo rigurosamente las actividades y los artefactos que se deben producir, las herramientas y notaciones a utilizar, las cuales se denominan metodologías tradicionales; y otras denominadas metodologías ágiles se centran más en las personas que en el proceso, dan mayor valor al individuo, al equipo de desarrollo y a su colaboración con el cliente; proponiendo un desarrollo incremental del software con iteraciones muy cortas y la documentación necesaria [1]. Surgen a mediados de los años 90, ante la necesidad de una nueva forma de desarrollar software diferente a las que hasta el momento se habían utilizado (metodologías tradicionales).

Las metodologías ágiles de desarrollo de software proponen una estrecha relación entre el cliente y el equipo de desarrollo, hacer entregas funcionales del software continuamente, se aplican a proyectos con requerimientos cambiantes o imprecisos, y están dirigidas fundamentalmente para equipos pequeños.

Diversas han sido las metodologías ágiles que han surgido en los últimos años, y como es de esperar algunas más populares que otras. En una encuesta realizada en el 2006 por CM Crossroads en Estados Unidos se obtuvo como resultado que la metodología más utilizada era XP, seguida por FDD, Scrum, Crystal y DSDM [2].

En otra encuesta más reciente, realizada por Methods & Tools en febrero de este año 2008, acerca del desarrollo ágil de software en 512 compañías, y comparándola con la misma encuesta realizada en el 2005 a 232 compañías del mismo lugar, se obtuvo como resultado que el nivel de desconocimiento acerca del desarrollo ágil disminuyó en la mitad, solo el 13 % de las organizaciones no conocía de la existencia de las metodologías ágiles en el 2008. El número de organizaciones donde todos los nuevos proyectos se desarrollan con metodologías ágiles se duplicó y el 56 % de las compañías las utiliza (ya sea en proyectos pilotos, implementación parcial, desarrollo parcial o total); valor que se incrementó desde el 2005 cuando solo el 41% lo hacía [3].

En la investigación desarrollada, de las fuentes bibliográficas consultadas cerca del 17% son del año en curso (2008), mientras que casi el 42% abarca el período de los últimos tres años, por lo que el estudio presentado es de actualidad.

METODOLOGÍAS ÁGILES

Las metodologías ágiles para el desarrollo de software son adaptativas, enfocándose en las personas y los resultados. Todas las metodologías ágiles proponen una estrecha relación entre el equipo de desarrollo y los clientes; estableciendo como base la comunicación cara a cara; lo cual resulta más eficiente que cualquier documentación escrita que se pueda obtener de los clientes en la etapa de captura de requisitos, a través de entrevistas y encuestas, u otra documentación generada por investigación de documentos u observación directa.

Según la mayoría de los autores consultados, las metodologías ágiles están orientadas para ser utilizadas en proyectos de poca envergadura, equipos pequeños (hasta diez miembros) y no distribuidos. No obstante existen opiniones de que se puede utilizar una variante pesada de una metodología ágil para equipos de mayor tamaño que se encuentren distribuidos. Un estudio presentado por Jeff Sutherland (creador de la metodología ágil Scrum junto a Ken Schwaber y Mike Vedle) en la Conferencia Internacional en Sistemas Complejos en el 2006, evidencia resultados positivos en la aplicación de una variante de Scrum integrada con prácticas de la metodología ágil Extreme Programming (XP) con equipos de desarrollo grandes y distribuidos [4]. Por otra parte la metodología ágil Crystal fue concebida para ser adoptada por equipos de desarrollo de diferentes tamaños. Posee variantes que dependen de la cantidad de miembros del equipo de desarrollo, el que puede tener hasta 100 miembros [5].

Algo distintivo de las metodologías ágiles es que son adaptativas, porque están pensadas para contextos cambiantes, proyectos con requerimientos inestables, cambios tecnológicos y cambios de personal. Están orientadas a las personas, porque necesitan gran participación y compromiso de todos, incluyendo al cliente, en el desarrollo del software.

Las metodologías ágiles siguen una combinación del modelo de desarrollo de software evolutivo y el modelo incremental mediante ciclos de desarrollo cortos (cada uno depende de la metodología que se utilice) y entregas continuas de software funcional.

METODOLOGÍAS ÁGILES ACTUALES

Varias han sido las metodologías ágiles que han surgido desde los años 90. Cada una con sus particularidades para dar solución a diferentes problemas, pero con un objetivo común de mejorar la producción de software. Algunas se han destacado, por sus características, en los últimos años.

Extreme Programming (XP). Creada por Kent Beck en 1996. Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en la realimentación continua entre el cliente y el equipo de desarrollo, la comunicación fluida entre todos los participantes, la simplicidad en las soluciones implementadas y el coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico [6].

Scrum. Desarrollada por Ken Schwaber, Jeff Sutherland y Mike Vedle en 1989. Define un marco para la gestión de proyectos. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos: el desarrollo de software se realiza mediante iteraciones, denominadas sprint, con una duración de 30 días, el resultado de cada sprint es un incremento ejecutable que se muestra al cliente y la segunda característica son las reuniones a lo largo del proyecto, entre ellas se destaca la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración [7].

Crystal Methodologies. Se trata de un conjunto de metodologías para el desarrollo de software caracterizadas por estar centradas en las personas que componen el equipo y la reducción al máximo del número de artefactos producidos. Han sido desarrolladas por Alistair Cockburn desde inicio de los 90. El desarrollo de software se considera un juego cooperativo de invención y comunicación, limitado por los recursos a utilizar. El equipo de desarrollo es un factor clave, por lo que se deben invertir esfuerzos en mejorar sus habilidades y destrezas, así como tener definidas políticas de trabajo en equipo. Estas políticas dependerán del tamaño del equipo, estableciéndose una clasificación por colores, por ejemplo Crystal Clear (3 a 8 miembros), Crystal Yellow (8 a 20 miembros), Crystal Orange (20 a 50 miembros) y Crystal Red (50 a 100 miembros) [5].

Dynamic Systems Development Method (DSDM). Se fundó en Inglaterra en 1994. Define el marco para desarrollar un proceso de producción de software. Sus principales características son que es un proceso iterativo e incremental y que el equipo de desarrollo y el usuario trabajan juntos. Propone cinco fases: estudio de viabilidad, estudio del negocio, modelado funcional, diseño y construcción, y finalmente implementación. Las tres últimas son iterativas, además de existir realimentación en todas las fases [8].

Adaptive Software Development (ASD). Su impulsor es Jim Highsmith. Sus principales características son: iterativo, orientado a los componentes del software más que a las tareas y tolerante a los cambios. El ciclo de vida que propone tiene tres fases esenciales: especulación, colaboración y aprendizaje. En la primera de ellas se inicia el proyecto y se

planifican las características del software; en la segunda desarrollan las características y finalmente en la tercera se revisa su calidad, y se entrega al cliente. La revisión de los componentes sirve para aprender de los errores y volver a iniciar el ciclo de desarrollo [9].

Feature-Driven Development (FDD). Sus impulsores son Jeff De Luca y Peter Coad. Define un proceso iterativo que consta de 5 pasos. Las iteraciones son cortas (hasta 2 semanas). Se centra en las fases de diseño e implementación del sistema partiendo de una lista de características que debe reunir el software [10].

EXTREME PROGRAMMING (XP)

Entre las metodologías ágiles más utilizadas se encuentra Extreme Programming, más conocida por sus siglas XP; la cual surge en 1996. Tiene sus antecedentes desde mediados de los años 80 cuando Kent Beck y Ward Cunningham probaron formas de desarrollar software diferentes a las que se habían utilizado hasta el momento, trabajando en un grupo de investigación de Tektronix. Posteriormente, en los 90, Beck comenzó un proyecto en DaimlerChrysler, conocido como C3 (Chrysler Comprehensive Compensation). Este proyecto fue la cuna de XP [11].

XP es definida por su autor, Kent Beck, como una metodología ágil para el desarrollo de software destinada a ser utilizada por equipos de desarrollo pequeños y medianos (de 2 a 10 miembros) que se enfrenten a proyectos con requerimientos imprecisos o cambiantes. Las relaciones desarrollador-desarrollador y desarrolladores-cliente son fundamentales en esta metodología [6]. La adopción del cliente como un miembro más del equipo de desarrollo es la clave del éxito.

ROLES Y SUS RESPONSABILIDADES

XP, como toda metodología para el desarrollo de software, propone varios roles. Cada uno acarrea consigo ciertas responsabilidades; algunas son menos complejas que otras.

Programador: es una pieza clave en XP. Su responsabilidad no se limita a implementar cierta funcionalidad del sistema; él también debe comunicarse, ya sea con otros miembros del equipo de desarrollo o con el cliente, elaborar pruebas unitarias y llevar a cabo las integraciones del sistema.

Cliente: "El cliente es la otra mitad de la importante dualidad de XP. El programador sabe cómo programar. El cliente sabe que programar" [1] [6].

[1] The customer is the other half of the essential duality of extreme programming. The programmer knows how to program. The customer knows what to program.

Ser cliente de XP no es tarea fácil. El cliente debe escribir las historias de usuario y las pruebas funcionales del sistema, asignarle prioridad a las historias de usuarios y tomar decisiones acerca de cuál se debe implementar en cada iteración [6]; en fin convertirse en un miembro más del equipo de desarrollo.

Encargado de pruebas: verifica que el sistema esté funcionando correctamente. Entre los deberes del encargado de pruebas se encuentran ejecutar regularmente todos los casos de prueba, informar al equipo los resultados obtenidos y ayudar al cliente a escribir las pruebas funcionales del sistema. En caso de utilizarse alguna herramienta de soporte para pruebas, ésta se encontrará bajo su responsabilidad.

Encargado de seguimiento: es la conciencia del equipo de desarrollo. Debe verificar el cumplimiento del plan de entrega y del plan de iteración, e informar si las estimaciones realizadas fueron correctas, se subestimó o sobrestimó; con el objetivo de que el equipo sea más preciso en futuras estimaciones.

Entrenador: es quien advierte si ocurre una desviación en el proceso, mantiene la calma cuando todos se encuentran "aterrados", en fin, guía al equipo de desarrollo para que se siga el proceso XP correctamente.

Consultor: es responsable de guiar al equipo de desarrollo para resolver los problemas que se les presente en un tema específico. El equipo de desarrollo en ocasiones necesita conocimientos de un tema específico, el cual no domina ninguno de sus miembros. Es un miembro externo al equipo con conocimientos específicos en algún tema necesario para la construcción del sistema.

Gestor: es el la máxima autoridad del equipo de desarrollo. Él debe poseer ciertas cualidades como coraje, confianza y en ocasiones insistencia sobre los miembros del equipo para que se realice el trabajo, pero no debe agobiarlos [6].

CICLO DE VIDA DE XP

El ciclo de vida ideal de XP consta de 6 fases: exploración, planificación iteraciones, producción, mantenimiento y muerte del proyecto. Se utiliza la palabra ideal porque se reconoce que es muy baja la probabilidad de que el desarrollo de dos proyectos sea exactamente igual [6].

Un proyecto se inicia con una fase de exploración donde se sientan las bases para que sea exitoso su desarrollo. El plan de entrega a seguir es concebido durante la planificación, con la participación del cliente y los desarrolladores. A continuación tienen lugar una serie de iteraciones que no concluyen hasta obtener una primera versión del sistema, o una primera entrega que es lo mismo. Se pone en producción esta versión y comienza el mantenimiento donde se implementan nuevas funcionalidades y se mantiene el sistema funcionando. En la fase de mantenimiento tiene lugar la implementación de

nuevas versiones del sistema; cada nueva entrega, debe comenzar por una fase de exploración y es ahí cuando se cierra el ciclo. La única manera de interrumpir el ciclo de XP es que ocurra la muerte del proyecto.

GUÍA DE APLICACIÓN DE XP

Si se desea desarrollar utilizando XP es muy útil contar con una guía que explique paso a paso qué hacer. En ninguna de la bibliografía consultada se encontró documento alguno de este tipo, por tanto se propone la siguiente guía de aplicación, la cual fue probada mediante el desarrollo de un caso de estudio [12].

Exploración

1. Los desarrolladores y el cliente elaboran la metáfora.
2. El cliente redacta las historias de usuario.
3. Los desarrolladores dividen las historias de usuario en tareas de programación y calculan los puntos estimados de cada historia de usuario.
4. Los desarrolladores estudian las tecnologías a utilizar.
5. Los desarrolladores construyen uno o varios prototipos del sistema.
6. Los desarrolladores realizan una propuesta inicial de la arquitectura del sistema.
7. El cliente redacta los casos de prueba ayudado por el encargado de pruebas.

Planificación

8. El cliente asigna prioridad a las historias de usuario.
9. Los desarrolladores estiman el esfuerzo total.
10. Los desarrolladores estiman la velocidad del equipo.
11. El cliente y los desarrolladores elaboran el plan de entrega.
12. Actualizar documentación.

Iteraciones

13. El cliente reasigna prioridad a las historias de usuario.
14. El cliente y los desarrolladores elaboran el plan de iteración.
15. Los desarrolladores definen el diseño preliminar.
16. El cliente y los desarrolladores mejoran los casos de prueba.
17. Los programadores comienzan la implementación.
18. Los desarrolladores mejoran el diseño.
19. El cliente y los desarrolladores completan y actualizan el plan de iteración.
20. Actualizar documentación.

Producción

21. Completar y actualizar el plan de entrega.
22. El cliente y los desarrolladores se reúnen diariamente para informar en que continúa trabajando cada cual.
23. El cliente y los desarrolladores realizan continuas pruebas al sistema.

Mantenimiento

24. El cliente y los desarrolladores comienzan una nueva entrega, con lo que se regresa a la fase de exploración.

Muerte

25. Muerte del proyecto.

CONCLUSIONES

Desde mediados de los años 90 surgen formas de desarrollar software diferentes a las tradicionales, que en el año 2001 se oficializan bajo el nombre de metodologías ágiles.

Las metodologías ágiles son promueven prácticas adaptativas en vez de predictivas, se centran en las personas. Son iterativas, orientadas hacia la entrega, de comunicación intensiva entre el cliente y los desarrolladores, los que trabajan juntos con una constante comunicación. Recomendadas para proyectos con los requisitos imprecisos y/o cambiantes.

XP es una metodología ágil para el desarrollo de software que se recomienda utilizar para proyectos con requisitos imprecisos o cambiantes por equipos de desarrollo, pequeños y medianos.

En XP existen 7 roles, cada uno con una función específica dentro del proceso de desarrollo. Una misma persona puede encarar diferentes roles.

Debe respetarse el ciclo de vida de XP, que consta de 6 fases, pero de considerarse por el equipo de desarrollo éste se puede alterar.

El diseño y la planificación tienen lugar a lo largo de todo el proceso XP.

No en todos los entornos ni a todo tipo de proyecto se puede aplicar XP.

La guía de aplicación propuesta describe detalladamente que hacer en cada fase del ciclo de XP; por tanto, facilita el desarrollo utilizando la metodología y constituye un aporte debido a que no se encontró documento alguno de este tipo.

REFERENCIAS

[1] André Ampuero, M., V. D. Muñoz Castillo. "*¿Metodologías tradicionales o metodologías ágiles?",* 2008.
[2] CM Crossroads. (2006). Disponible en: http://www.cmcrossroads.com/h.html [Consulta 7 de diciembre de 2007].
[3] Methods & Tools. (2008). Disponible en: http://www.methodsandtools.com/dynpoll/oldpoll.php?Agile2 [Consulta 14 de marzo de 2008].
[4] Sutherland, J., A. Victorov, y J. Blount. "Adaptive Engineering of Large Software Projects with Distributed/Outsourced Teams". 2006.
[5] Cockbun, A. "*Agile Software Development"*. Addison-Wesley, 2001.
[6] Beck, K. "*Extreme Programming Explained. Embrace Change"*. Addison-Wesley, 1999.
[7] Schwaber, K. y M. Beedle. "*Agile Software Development with SCRUM"*. 2001.
[8] Stapleton, J. "*Dsdm Dynamic Systems Development Method: The Method in Practic"e*. Addison-Wesley, 1997.

[9]Highsmith, J.A. "*Adaptive Software Development: A Collaborative Approach to Managing Complex Systems"*. Dorset House, 2000.
[10] Coad, P., Jeff De Luca, E. Lefebvre. "*Java Modeling In Color With UML: Enterprise Components and Proces"s*. Prentice Hall, 1999.
[11] Reynoso, C. "*Métodos Heterodoxos en Desarrollo de Software"*. 2004.
[12] Pérez. Ramírez D. *"Investigación de la metodología ágil Extreme Programming y su aplicación a un caso de estudio"*. Y Guntín Oliveros Y., Coello Mena J. (Tutor). Tesis de Grado. La Habana, Instituto Superior Politécnico José Antonio Echeverría. 2008.