Calidad en la especificación de requerimientos de software aplicado en metodologías ágiles

Daniela Saavedra Menchaca
Postgrado en Informática
Universidad Mayor de San Andrés
La Paz - Bolivia
saavedra.danis@gmail.com

***Resumen*—Los inconvenientes que se presentan en el desarrollo de un proyecto de software repercuten cuando se realiza la entrega del producto al cliente, y precisamente es en la etapa inicial o de planificación donde se originan estos problemas, esto podría ser por la mala estimación de tiempos, falta de definición de riesgos, y requerimientos poco especificados. La especificación de requerimientos es el paso principal que se debe realizar para proseguir con las fases de desarrollo de software. En este artículo se explora la especificación de requerimientos en metodologías agiles y cómo estás se han adaptado al ciclo de vida del software. Se analiza el proceso de especificación de requerimientos en la Metodología Scrum y Metodología Xtreme Programming (XP) para después hacer una comparación de sus procesos y de esa manera obtener una buena práctica que sea aplicable para ambas metodologías.**

Palabras clave—Metodologías de desarrollo, ingeniería de requerimientos, ciclo de vida del software, metodología ágil

I. Introducción

Las especificaciones de los requerimientos son la puerta de entrada a las fases posteriores del proceso de desarrollo de software, por ello la calidad relacionada con la especificación es muy importante para obtener un producto de software adecuado y de esa manera cumplir con los plazos y costos definidos. La definición de los requerimientos no es una actividad trivial por ello se han desarrollado modelos que contienen los procesos definidos donde estos se adecuan de acuerdo con las circunstancias del proyecto y el contexto de la aplicación.

La ingeniería de requerimientos consta de las actividades de elicitación[1], documentación y verificación que permiten conocer la situación del proyecto, sin embargo, la inestabilidad o modificaciones incrementales permite que el software se entregue en pequeñas partes denominadas incrementos. Cada iteración es definida como un subproyecto donde se lleva a cabo la planificación, y se trata a los requerimientos con mucha importancia, estos deben ser concisos y fáciles de entender, una vez culminado el desarrollo se realiza las pruebas con el fin de producir un subconjunto del sistema final. El proceso se repite varias veces hasta que el producto este completo. Actualmente las metodologías ágiles adoptan este ciclo, sin embargo, cada una cuenta con sus propias características.

En base a lo mencionado anteriormente se plantea como objetivo determinar la calidad de especificación de requerimientos revisando bibliografía existente sobre las técnicas de especificación de requerimientos en metodologías ágiles: Scrum y Xtreme Programming.

II. Metodologías Ágiles

En el año 2001 se reunieron varias personas para armar un esquema de trabajo que atacan las problemáticas del desarrollo de software, ya que las metodologías tradicionales ocasionaban ciertos problemas. Las metodologías ágiles se enfocan en procesos incrementales con entregas funcionales, que buscan aumentar la confianza de los clientes al vincularlos en el proceso mediante la cooperación entre ellos y el equipo de trabajo. Las metodologías ágiles se caracterizan por reducir la gran cantidad de documentación que es común en las metodologías tradicionales, están buscan ser adaptables y se basan en modelos de desarrollo incrementales. Estas metodologías se pueden adoptar al problema en cuestión sin importar el tamaño, trabajando por fases definidas y descomponiendo el proyecto para facilitar su gestión. Las metodologías más representativas en la actualidad las detallaremos a continuación:

A. Metodología SCRUM

Scrum (nombre que proviene de cierta jugada que tiene lugar durante un partido de rugby) es un método de desarrollo ágil de software concebido por Jeff Sutherland y su equipo de desarrollo a principios de la década de 1990. En años recientes, Schwaber y Beedle han desarrollado más los métodos Scrum [1].

Los principios Scrum son congruentes con el manifiesto ágil y se utilizan para guiar actividades de desarrollo dentro de un proceso de análisis que incorpora las siguientes actividades estructurales: requerimientos, análisis, diseño, evolución y entrega. Dentro de cada actividad estructural, las tareas del trabajo ocurren con un patrón del proceso llamado sprint. El trabajo realizado dentro de un sprint (el número de éstos que requiere cada actividad estructural variará en función de la complejidad y tamaño del producto) se adapta al problema en cuestión y se define —y con frecuencia se modifica— en tiempo real por parte del equipo Scrum.

Scrum acentúa el uso de un conjunto de patrones de proceso del software que han demostrado ser eficaces para proyectos con

Para referenciar este artículo (IEEE):
[N] D. Saavedra, «Calidad en la especificación de requerimientos de software aplicado en metodologías ágiles», *Revista PGI. Investigación, Ciencia y Tecnología en Informática*, nº 8, pp. 61-64, 2020.

[1] La elicitación de requisitos es el fundamento o base primordial en el desarrollo de proyectos software y es la fase que proporciona el impacto más alto en el diseño y en las demás fases del ciclo de vida del producto. [10]

plazos de entrega muy apretados, requerimientos cambiantes y negocios críticos. Cada uno de estos patrones de proceso define un grupo de acciones de desarrollo.

B. *Metodología XP*

La programación extrema o Xtreme Programming, es un framework de desarrollo de software basado en los métodos ágiles, que evidencia principios tales como el desarrollo incremental, la participación del cliente, el interés en las personas y no en los procesos como elemento principal, y aceptar el cambio y la simplicidad. [1] El proceso está constituido por las siguientes fases: exploración, planificación, iteraciones, producción, mantenimiento y cierre de proyecto. XP propone las siguientes buenas prácticas: planificación incremental, entregas pequeñas, diseño sencillo, desarrollo previamente aprobado, limpieza de código o refactorización, programación en parejas, propiedad colectiva, integración continua, ritmo sostenible y cliente presente.

III. Especificación de Requerimientos en las Metodologías Ágiles

Para mejorar el proceso de gestión de requerimientos el trabajo se centra en las actividades que se realizan en esta etapa, con ese fin se analiza las prácticas ágiles en las metodologías mencionadas anteriormente.

A. *SCRUM*

La metodología Scrum se compone de una reunión de planificación y una reunión diaria de Scrum, la reunión de planificación se lleva a cabo al comienzo de cada sprint. En esta reunión, los miembros del equipo determinan la cantidad de requerimientos que pueden administrarse, es decir, crean una acumulación de Sprints a partir de eso. El backlog de Sprint contiene la lista de todas las tareas que deben realizarse durante un Sprint en particular. Las reuniones diarias de scrum no duran más de 15 minutos, donde el dueño del producto (PO) recibe actualizaciones continuas sobre el proceso de desarrollo y puede proporcionar comentarios sobre las funciones que se incluyen como se explica en la siguiente figura:

Fig. 1. Estructura de Scrum [11]

De esta manera, si un PO decide agregar una nueva función a un sprint, puede discutirlo con el equipo de desarrollo y ahorrar tiempo en lugar de revisarlo al final y exigir un cambio al final.

El equipo lleva a cabo una revisión del sprint al final de cada sprint en el que demuestran nuevas características y funcionalidades al PO u otras partes interesadas que pueden proporcionar cualquier tipo de retroalimentación que podría ser beneficiosa o útil de alguna manera para el próximo sprint. Este ciclo de retroalimentación da como resultado modificaciones a la funcionalidad entregada recientemente, luego, nuevamente, es más probable que se revisen o agreguen nuevos requerimientos a la cartera de productos. Otra actividad en la gestión de proyectos Scrum es la retrospectiva de Sprint. El Scrum Master, PO y el equipo de desarrollo participan en esta reunión. Es la oportunidad de reproducir o revisar el sprint que ha finalizado e identificar nuevas formas de mejorar. Scrum consta de tres artefactos: sprint backlogs, product backlogs y burn down charts.

El Product Backlogs, priorizada por el PO, es una lista completa de la funcionalidad (escrita como historias de usuario) que se agregará eventualmente al producto. Se prioriza para que el equipo siempre pueda trabajar primero en las características más importantes, urgentes y valiosas.

Por otro lado, Sprint Backlog es la lista de todas aquellas tareas que el equipo debe realizar durante el Sprint para entregar la funcionalidad requerida.

La cantidad restante de trabajo ya sea en un sprint o en un lanzamiento, se muestra mediante gráficos "Burn down". Es una herramienta eficaz para concluir si un sprint o lanzamiento está programado para tener todo el trabajo planificado terminado a tiempo.

La ingeniería de requerimientos tradicional consume mucho tiempo y requiere un proceso rápido para satisfacer oportunamente las necesidades del mercado, por lo que la industria del software moderno exige un proceso rápido e iterativo como el desarrollo ágil para hacer frente a los requisitos y el tiempo cambiantes.

Hay muchos factores involucrados en el éxito o el fracaso de un producto, uno de ellos es recopilar y priorizar los requerimientos [2]. La obtención y priorización de requerimientos es una de las tareas más desafiantes durante el desarrollo del producto y es muy poco probable que pueda escribir todos los requerimientos en una etapa temprana, ya que evolucionan continuamente a lo largo del proceso de desarrollo y deben abordarse adecuadamente para satisfacer las necesidades cambiantes. de mercado y tiempo. Dado que scrum es una metodología ágil, permite a los ingenieros manejar los requerimientos cambiantes a medida que evolucionan; sin embargo, aún es una tarea desafiante comprender qué requerimientos previos son lo suficientemente vitales para tener una gran necesidad y ser incorporados en los primeros sprints. La organización de los requerimientos en prioridades ayuda al equipo del proyecto a comprender qué requerimientos son más esenciales y urgentes de implementar y ejecutar. La priorización también es una actividad útil para la toma de decisiones en otras fases de la ingeniería de software. Por lo tanto, debe haber una técnica de priorización de requerimientos bien administrada incluida en los procesos de scrum que mejore su calidad.

B. *XP*

Barry Boehm presentó que el costo del cambio crece exponencialmente a medida que el proyecto avanza en su ciclo de vida [3], Stuart Faulk lo reitera al afirmar que el costo relativo de reparación es 200 veces mayor en la fase de mantenimiento que si se encuentra atrapado en la fase de requerimientos [4]. XP cuestiona que este ya no es el caso.

Si bien es más caro modificar el código que modificar una descripción en prosa, con los lenguajes modernos y las técnicas de desarrollo no es un aumento exponencial.

Para la obtención de requerimientos en XP se agrega un miembro de la organización del cliente al equipo. Este representante del cliente trabaja a tiempo completo con el equipo, escribiendo historias, similar a los casos de uso de Universal Markup Language (UML): desarrollo de pruebas de aceptación del sistema y priorización de requerimientos [5].

La especificación no es un solo documento monolítico; en cambio, es una colección de historias de usuarios, las pruebas de aceptación escritas por el cliente y las pruebas unitarias escritas para cada módulo. Dado que el cliente está presente durante todo el desarrollo, ese cliente puede considerarse parte de la especificación, ya que está disponible para responder preguntas y aclarar ambigüedades.

El ciclo de vida de XP es de naturaleza evolutiva, pero los incrementos se hacen lo más pequeños posible. Esto permite al cliente (y a la gerencia) ver un progreso concreto a lo largo del ciclo de desarrollo y responder a los cambios de requerimientos más rápidamente. Hay menos trabajo involucrado en cada lanzamiento, por lo tanto, las etapas de estabilización que requieren mucho tiempo antes de los lanzamientos toman menos tiempo. Con un tiempo de iteración más largo, puede llevar un año incorporar una nueva idea: con XP esto puede suceder en menos de una semana [6].

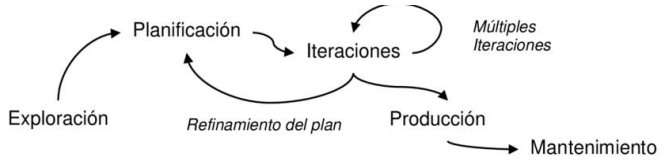

Fig. 2. Ciclo de vida de un proyecto basado en XP [7]

Un aspecto fundamental de XP son las pruebas. El cliente especifica las pruebas del sistema; los desarrolladores escriben pruebas unitarias. Este código de prueba sirve como parte de la definición de requerimientos: un caso de prueba codificado es un medio inequívoco en el que registrar un requerimiento. XP requiere que los casos de prueba se escriban primero, y luego se escriba la cantidad más simple de código para especificar el caso de prueba. Esto significa que los casos de prueba ejercerán toda la funcionalidad relevante del sistema, y la funcionalidad irrelevante no debería ingresar al sistema [8].

Este artículo describe y evalúa los procesos de ingeniería de requerimientos asociados con el paradigma XP.

El proceso de ingeniería de requerimientos de XP, Harwell y col. Divida los requerimientos en dos tipos:

Un parámetro de producto se aplica al producto en desarrollo, mientras que un parámetro de programa se ocupa de los esfuerzos de gestión que permiten que el desarrollo tenga lugar [9].

El cliente que se convierte en miembro del equipo de XP define los parámetros del programa y del producto.

Los parámetros del producto se definen a través de historias y pruebas de aceptación, mientras que los parámetros del programa se tratan en la planificación del lanzamiento y la iteración. Los parámetros del producto se comunican principalmente a través de historias. Estas historias son similares a los casos de uso definidos en UML, pero tienen un alcance mucho más simple [5].

Desarrollar una especificación escrita integral es un proceso muy costoso, por lo que XP utiliza un enfoque menos formal. No es necesario escribir los requerimientos para responder a todas las preguntas posibles, ya que el cliente siempre estará allí para responder a las preguntas que surjan. Esta técnica se saldría rápidamente de control para un gran esfuerzo de desarrollo, pero para equipos pequeños y medianos (los equipos de menos de 20 personas se informan con mayor frecuencia) puede ofrecer un ahorro sustancial de costos. Sin embargo, debe tenerse en cuenta que un representante del cliente sin experiencia pondría en peligro esta propiedad.

Luego, los programadores toman cada historia y estiman cuánto tiempo creen que llevará implementarla. El alcance se controla en este punto, si un programador piensa que la historia, aislada, tomará más de dos semanas para implementar, se le pide al cliente que divida la historia. Si los programadores no comprenden la historia, siempre pueden interactuar directamente con el cliente. Una vez que se estiman las historias, el cliente selecciona qué historias se implementarán para el próximo lanzamiento, impulsando así el desarrollo de los intereses comerciales.

En cada lanzamiento, el cliente puede evaluar si el próximo lanzamiento aportará valor comercial a la organización.

Cada historia que se implementará se divide en tareas. Un par de programadores trabajarán para resolver una tarea a la vez. El primer paso para resolver una tarea (después de comprender, por supuesto) es escribir un caso de prueba para ello. Los casos de prueba definirán exactamente lo que se debe codificar para esta tarea. Una vez que pasan los casos de prueba, la codificación está completa [8]. Por lo tanto, las pruebas unitarias también pueden considerarse una forma de requerimientos. Cada prueba (en todo el sistema) debe pasar antes de que se pueda integrar el nuevo código, por lo que estos requerimientos de prueba unitaria son persistentes. El desarrollo de software basado en pruebas de XP registra los requerimientos específicos de cada tarea en casos de prueba.

IV. Conclusiones

Del análisis de la literatura existente podemos decir que XP realiza la especificación de requerimientos durante todo el ciclo de vida del software en pequeñas etapas informales. El cliente se une al equipo de desarrollo a tiempo completo para escribir las historias de usuario, desarrollar pruebas de aceptación del sistema, establecer prioridades y responder preguntas sobre los requerimientos. Las historias de usuario tienen un alcance más simple para los casos de uso porque el cliente no necesita responder todas las preguntas imaginables. Las historias informales luego se traducen en pruebas de aceptación que tienen algunas propiedades de una especificación ejecutable.

A medida que surgen requisitos a lo largo del proceso de desarrollo de software, es necesario priorizar los requisitos y, por lo tanto, gestionarlos con la máxima prioridad, especialmente cuando el escenario es el del proceso de desarrollo de software ágil. Existen muchas técnicas de priorización de requisitos, metodologías propuestas y seguidas, pero la mayoría de ellas no tienen en cuenta los factores clásicos (métricas) que influyen en la calidad general del producto de

software que se está desarrollando, por ejemplo, ISO (9126, 25000, 830) métricas externas.

Durante el desarrollo de este trabajo podemos indicar que es importante saber diferenciar lo que es requerimiento de software de su especificación. El requerimiento explica el problema que el sistema o producto software debe resolver, sin embargo, la especificación del requerimiento es el registro del requerimiento en una o más formas, como ser: Especificación de Requerimientos de Software, Casos de Uso, Escenarios, Historias de Usuario, etc.

Para la definición de los atributos y características deseables en la especificación de requerimientos en metodologías ágiles, se revisaron los objetivos y las buenas prácticas para el desarrollo de estos y se encontró un modelo llamado INVEST que define las características para la escritura correcta de las historias de usuario.

Tanto en las metodologías ágiles y tradicionales existen mecanismos de obtener, analizar, especificar y verificar los requerimientos de un sistema. Los dueños del producto actúan como la voz del cliente ya que conocen sus necesidades y expectativas y son los responsables de la gestión y priorización de requerimientos. Los requerimientos pueden ser especificados en las historias de usuario y ser validados al finalizar de cada iteración.

De igual modo se pudo observar que para la definición de las historias de usuario es importante la comunicación y participación de los usuarios en el proceso de desarrollo. Expresar los requerimientos de software como historias de usuario resulta muy difícil para quienes comienzan a emplear las metodologías ágiles, los posibles errores cometidos en esta etapa pueden llevar a desarrollar casos de prueba equivocados y a la mala interpretación de requerimientos o en el peor de los casos, desarrollar el producto incorrecto.

REFERENCIAS

[1] I. F. T. e. Ingenieros, «SELECCIÓN DE TÉCNICAS DE INGENIERÍA DE SOFTWARE,» 2020. [En línea]. Available: https://virtual.itca.edu.sv/Mediadores/stis/43____mtodo_scrum.html. [Último acceso: 03 07 2020].

[2] M. Mohd y R. Mohd, «Validation of Requirement Elicitation Framework using Finite State Machine,» IEEE International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014.

[3] D. B. Boehm, Software Engineering Economics, Prentice Hall, 1981.

[4] F. Stuart, Software Requirements: A Tutorial, Software Engineering, IEEE, 1996.

[5] R. Jeffries, Extreme Programming Installed, Boston: Addison Wesley, 2001.

[6] B. Kent, Extreme Programming Explained: Embrace Change, Boston: Addison Wesley, 2000.

[7] Baird, «Tecnologías de información para acercar al ciudadano a los servicios de justicia en Colombia: El caso del mapa de oferta de justicia,» Diciembre 2011. [En línea]. Available: https://www.researchgate.net/figure/Figura-1-Ciclo-de-vida-de-un-proyecto-basado-en-XP-Fuente-Adaptada-de-Baird-2002_fig1_305301468. [Último acceso: 20 12 2020].

[8] U. S. R. Addison-Wesley, «Extreme Programming Explained,» Google Scholar, 2000. [En línea]. Available: http://scholar.google.com/scholar_lookup?title=Extreme%20Programming%20Explained%3A%20Embrace%20change&author=K..%20Beck&publication_year=2000. [Último acceso: 01 06 2020].

[9] R. Harwell, What is a Requirement?, Proceedings, Third Annual International Symposium National Council Systems Engineering, 1993.

[10] F. Cardona y J. Castaño, «Técnicas utilizadas para la toma y elicitación de requerimientos en la ingeniería del software,» 2015. [En línea]. Available: http://repositorio.utp.edu.co/dspace/bitstream/handle/11059/6110/0051C268.pdf?sequence=1&isAllowed=y. [Último acceso: 19 12 2020].

[11] «https://blog.conectart.com/,» [En línea]. Available: https://blog.conectart.com/la-metodologia-scrum-scrum-methodology/. [Último acceso: 19 12 2020].

Breve CV de la autora

Daniela Saavedra Menchaca es Licenciada en Informática por la Universidad Mayor de San Andrés. Actualmente realiza la Maestría en Alta Gerencia en Tecnologías de la Información y las Comunicaciones e Innovación para el Desarrollo (MAG-TIC) en el Postgrado en Informática UMSA. Tiene Certificación Internacional por ISTQB International Software Testing Qualification Board.

Ejerce profesionalmente como Ingeniera de Calidad en la empresa Jalasoft, y desarrolló cargos similares en Coderoad, Inmobiliaria Kantutani y Servicio General de Identificación Personal (SEGIP). Email: saavedra.danis@gmail.com.