Informe de investigación de la Metododología SXP

María E. Orozco Vaillant

"Año 55 de la Revolución"

RESUMEN

El presente documento recoge los resultados de la investigación que se realizó acerca de la metodología SXP, metodología ágil desarrollada en la Universidad de las Ciencias Informáticas en el año 2007 como parte de la estrategia para proponer una guía de procedimientos ágiles para el proceso de producción de software de la facultad 7 de dicha institución. De la metodología específicamente se caracteriza, las fases con que cuenta, que se debe hacer en cada una de ellas, los artefactos que se generan en estas y lo que recoge cada artefacto. Esta investigación ofrece una guía para la introducción de esta metodología con el objetivo de mejorar del proceso productivo en proyectos pequeños.

Palabras claves: metodología, plantilla, software.

ABSTRACT

This present document the results of research conducted on the SXP methodology agile methodology developed at the University of Computer Sciences in 2007 as part of the strategy to propose a guide to agile methods for the production process software faculty 7 of that institution. Methodology specifically characterizes the phases it has, should be done in each of them, the artifacts that are generated in these and what is in each artifact. This research provides guidance for the introduction of this methodology in order to improve the production process in small projects.

Keywords: methodology, template, software.

Índice

1. INTRODUCCIÓN

Una de las cuestiones más importantes cuando se va a desarrollar o producir un producto de software es sin duda la elección de la metodología de desarrollo de software a seguir para llevar a cabo este proceso, este es un paso muy importante debido a que su uso final está orientado a la estructuración, planificación y control del proceso de desarrollo.

Una metodología de desarrollo se refiere a un marco de trabajo que es usado para estructurar, planear y controlar el proceso de desarrollo en sistemas de información. Se clasifican en dos tipos principales, metodologías ágiles y metodologías robustas o pesadas.

Las metodologías ágiles tienen como filosofía centrarse en el factor humano y el producto haciendo al cliente partícipe del proceso de desarrollo, dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas se recomienda su uso en proyectos pequeños. Ejemplo de metodologías ágiles: XP, SCRUM, Crystal Methodologies, DSDM (Dynamic Systems Development Method), FDD (Feature-Driven Development).

Por su parte las metodologías robustas se centran principalmente en el control de proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir, y las herramientas y notaciones que se usarán. Este tipo de metodología ha demostrado ser efectivas y necesarias para un gran número de proyectos, principalmente proyectos grandes. Ejemplo de metodologías robustas: MSF (Microsoft Solution Framework), Métrica 3, RUP (Proceso Unificado de Desarrollo).[3]

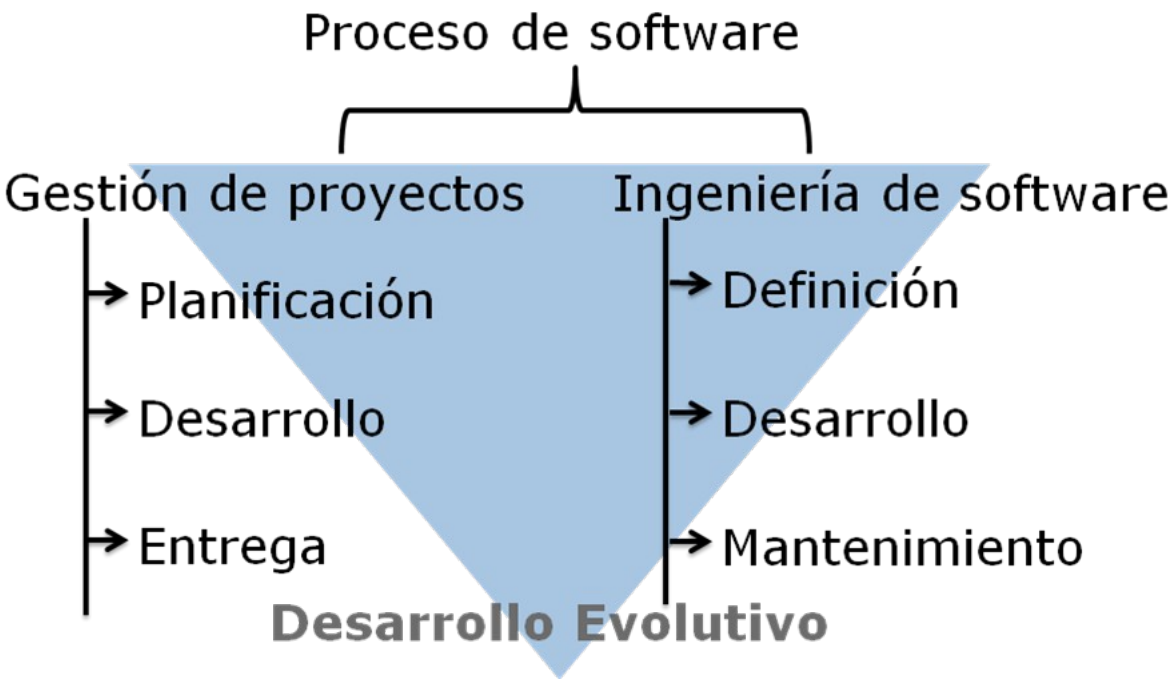
La metodología ágil SXP es la unión de XP y SCRUM. XP (eXtreme Programming - Programación Extrema) es una metodología de desarrollo formulada por Kent Beck, es el más destacado de los procesos ágiles de desarrollo de software, su particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar el éxito del proyecto.[3] Por su parte SCRUM, marco de trabajo también ágil para la gestión y desarrollo de software está basado en un proceso iterativo e incremental, fue desarrollada por Hirotaka Takeuchi e Ikujiro Nonaka, es una forma de gestionar un equipo de manera que trabaje de forma eficiente y de tener siempre medidos los progresos, de forma que sepamos por dónde andamos.[2] SXP por su parte, desarrollada en el 2007 en la Universidad de las Ciencias Informáticas está especialmente indicada para proyectos pequeños, con rápidos cambio de requisitos donde existe un alto riesgo técnico y se orienta a una entrega rápida de resultados y una alta flexibilidad.[5, 4]

Objetivo: Realizar un estudio de la metodología SXP.

2. DESARROLLO

2.1. Esquema de SXP

En la metodología se usará SCRUM para la planificación de los proyectos que utilicen metodologías ágiles, dado que SCRUM en sí no es una metodología de análisis ni de diseño, es una metodología de gestión de trabajo. De esta no se toma todo, solo algunas buenas prácticas. Para llevar a cabo el proceso de desarrollo como tal entonces se utiliza XP, con la idea principal de ir entregando versiones del producto que sean operativas aunque no cuenten con todas las funcionalidades requeridas por el cliente final.[5]



2.2. Fases de la metodología SXP

2.2.1. Planificación ↔ Definición[1][5]

En esta fase es donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.

Actividades

- Entrevista con el cliente (concepción inicial): escribir la visión (concepción del sistema), presupuesto y reserva del producto con estimaciones iniciales.
- Juego de la planificación: es un espacio frecuente de comunicación entre el cliente y los programadores. El equipo técnico realiza una estimación del esfuerzo requerido para la implementación de las historias de usuario y los clientes deciden sobre el ámbito y tiempo de las entregas y de cada iteración.
- Captura de requisitos.
    - Creación de la LRP (Lista de Reserva del Producto).
    - Priorización de la LRP.
    - Definir las historias de usuario.
    - Asignar las historias de usuario.
- Valoración del esfuerzo.
- Valoración de riesgos.
- Diseño con las metáforas: el sistema es definido mediante una metáfora o un conjunto de metáforas compartidas por el cliente y el equipo de desarrollo. Una metáfora es una historia compartida que describe cómo debería funcionar el sistema.
- Refactorización: la refactorización es una actividad constante de reestructuración del código con el objetivo de remover duplicación de código, mejorar su legibilidad, simplificarlo y hacerlo más flexible para facilitar los posteriores cambios.
- Reunión de revisión del diseño.

Artefactos

- Plantilla de concepción del sistema: en esta plantilla se refleja la visión general del producto a implementar, recoge los diferentes roles que intervendrán en el desarrollo del software, así como las tecnologías usadas.
- Plantilla LRP: es una lista priorizada que define el trabajo que se va a realizar en el proyecto. Los posibles elementos de esta lista son requerimientos técnicos y del negocio, funciones, y actualizaciones tecnológicas requeridas. Cuando un proyecto comienza es muy difícil tener claro todos los requerimientos sobre el producto. Sin embargo, suelen surgir los más importantes que casi siempre son más que suficientes para un Sprint (Iteración), un sprint es un ciclo iterativo en el cual se desarrolla o mejora una funcionalidad para producir nuevos incrementos.

- Plantilla Historia de usuario: en esta plantilla se especifican los requisitos del software, las historias de usuarios son escritas por el cliente como las tareas que el sistema debe hacer y su construcción depende principalmente de la habilidad que tenga el cliente para definirlas, escritas en lenguaje natural y sin un formato predeterminado.
- Plantilla Lista de riesgos: en ésta son definidos los posibles riesgos que actuarán sobre el proceso de desarrollo de software, así como la estrategia trazada para mitigarlos.
- Plantilla Modelo de diseño: en esta plantilla se define un esbozo inicial del diseño del sistema, sin entrar en especificaciones, ni detalles, solo lo que el diseñador necesita para hacer una primera versión del sistema.

2.2.2. Desarrollo

Es la fase donde se realiza la implementación del sistema hasta que esté listo para ser entregado.
Actividades

- Junta de planificación.
    - Definir las historias de usuario a implementar.
    - Definir las tareas para lograr la implementación.
- Implementación.
    - Estándar de código
- Junta de seguimiento.
- Taller técnico.
- Junta de revisión.
- Pruebas.

Artefactos

- Plantilla de Glosario de términos: recoge los términos que se relacionan con el sistema y la metodología utilizada que pueden causar dudas al cliente para un mejor entendimiento de todo el proceso de desarrollo de software.
- Plantilla de Tareas de Ingeniería: en esta plantilla se recogen las tareas por historias de usuario a realizar.
- Plantilla Cronograma de producción: recoge las actividades realizadas en el equipo de desarrollo durante la iteración. Se realiza un cronograma para cada iteración planificada durante el proceso.
- Plantilla de Plan de Release: cada una de las iteraciones que se van a desarrollar para la realización del producto, la descripción del objetivo de la misma, el número de historias de usuario que se van a implementar en cada una de las iteraciones por orden de prioridad y la duración total que va a ser el tiempo estimado según las HU propuestas en que demorará su implementación.
- Estilo de código: plantilla con el estilo de código del o los lenguajes de programación con los que se va a desarrollar la aplicación. Esto se hace ya que es importante que los programadores del equipo de desarrollo sigan un determinado estándar de código, lograr la comunicación de los programadores, ya que todos no tienen la misma forma de programar, se pueden seguir estándares del mismo equipo, de la organización a la cual pertenecen u otros estándares reconocidos para los lenguajes de programación que se utilizan, fundamentales cuando los programadores cambian de pareja, con esto se consigue un código legible, con el mismo estilo y homogéneo.
- Código fuente: esto no es una plantilla sino un indicativo del código fuente comprimido y además otro paquete compilado con sus dependencias.
- Plan de Pruebas: contiene el listado de historias de usuarios que serán probadas, el cronograma con las observaciones realizadas a cada tarea de las historias de usuario probadas. Recoge además el resultado de las pruebas unitarias, de las pruebas de instalación y configuración, de las pruebas de seguridad, pruebas de recuperación y tolerancia a fallos y por último la evaluación de las pruebas.
- Plantilla Caso de Prueba de aceptación: en esta plantilla el desarrollador escribe las pruebas realizadas según la historia de usuario seleccionada para realizar la comprobación y validar las funcionalidades del sistema, y de esta forma saber si esta apto para ser liberado. Para la realización de la misma se debe tener en cuenta el tipo de prueba que se va a aplicar, de Caja Negra o de Caja Blanca, contiene los casos de prueba de aceptación que se realizan a diferentes funcionalidades que recogen el código del caso de prueba, con el nombre de la historia de usuario a la que le va a realizar la prueba, nombre de la persona que la realiza, la descripción de la misma, las condiciones de ejecución; que son las condiciones necesarias para poder realizar la prueba, el resultado esperado y la evaluación de la prueba.

Observación: Desde el diseño en la fase de Planificación $\leftrightarrow$ Definición y toda la fase de Desarrollo es un ciclo.

2.2.3. Entrega

Fase donde se pone en marcha el producto desarrollado y se hace la entrega al cliente.

Actividades

- Entrega de la documentación.
- Entrenamiento.
- Marketing.

Artefactos

- Manual de desarrollo: es una plantilla realizada por los desarrolladores, donde se recoge todo lo relacionado con las pautas realizadas para la programación del sistema.
- Instalador. (.deb, .exe y .tar.gz)
- Cursos de capacitación: los cursos de capacitación que se le dan al cliente y otros interesados que utilizarán el producto.

2.2.4. Mantenimiento

En esta se realiza el soporte para el cliente.

Actividades

- Soporte.

Artefactos

- Plantilla de Gestión de cambios: esta plantilla guarda los cambios realizados en el sistema una vez es terminado el producto y se pasa a la fase donde se mantiene dándole soporte al cliente. Recoge los datos para identificar el nombre de la historia de usuario de donde se deriva el cambio, quien lo realiza y el lugar donde ocurre la actualización del código.

No se define ningún flujo en la metodología para el registro legal, pero se aclara para los proyectos estudiados de instituciones en Cuba donde una actividad fundamental es el registro de los proyectos libres a los que las instituciones dan financiamiento y aceptación oficial.

Registro Legal

En la actividad de registro se generan los siguientes artefactos, los cuales deben ser registrados.

- Manual de usuario: en esta plantilla el desarrollador realiza una guía de los pasos necesarios y de conceptos de aspectos importantes con los que cada uno de los usuarios podrá consultar en caso de dudas para interactuar con el sistema.
- Manual de identidad: el diseñador debe hacer una caracterización general de los aspectos que se tomaron en cuanta para la realización del diseño del sistema.
- Registro legal.

3. CONCLUSIONES

En este investigación se realizó una descripción de cada una de las fases de la metodología SXP, las actividades que se realizan en cada una de ella y los artefactos generados en estas, así como una breve explicación del contenido de cada artefacto en cada una de las fases, se logró:

- Conocer más a fondo la metodología de desarrollo de software que se utiliza en la mayoría de los proyectos productivos de nuestra facultad.
- Profundizar en estos conocimientos que pueden servir más adelante, sobre todo para la futura tesis para optar por el título de Ingenieros y para ser aplicados en los proyectos productivos que se llevan a cabo en la facultad, si se emplea esta metodología en estos.

4. REFERENCIAS

Referencias

[1] Ing. Abel Meneses Abd, Ing. Gladys Marsi Peñalver Romero, and Ing. Malay Rodríguez Villar. Guión de la metodología scrum & xp unicornios methodology, February.

[2] Henrik Kniberg. Scrum and xp from the trenches. *Lulu. com*, 2007.

[3] Patricio Letelier. Métodologías ágiles para el desarrollo de software: extreme programming (xp). 2006.

[4] Gladys Marsi Peñalver Romero. *MA-GMPR-UR2 Metodología ágil para proyectos de software libre.* PhD thesis, Universidad de Ciencias Informáticas, 2008.

[5] Malay Rodruíguez Villar. *Introducción de procedimientos ágiles en la producción de software en la Facultad 7 de la Universidad de Ciencias Informáticas.* PhD thesis, Universidad de Ciencias Informáticas, 2007.