Lectura 5. Metodologías tradicionales vs metodologías ágiles

Alexa Montserrat Rivas Arambula

Lectura 5. Metodologías tradicionales vs metodologías ágiles

Las metodologías tradicionales se especializan en documentación, planificación y procesos. Entre las principales metodologías tradicionales tenemos los RUP y los MSF, que centran su atención en llevar una documentación exhaustiva de todo el proyecto y centran su atención en cumplir con un plan de proyecto, definiendo todo esto, en la fase inicial del desarrollo del proyecto.

Un Rup (RATIONAL UNIFIED PROCESS) es un proceso que provee un acercamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga los requerimientos de los usuarios finales (respetando cronograma y presupuesto).

Es guiado por casos de uso y centrado en la arquitectura, y utiliza UML como lenguaje de notación.

Fases

Las cuatro fases del ciclo de vida son:

Concepción

Elaboración

Construcción

Ventajas

-Evaluación en cada fase que permite cambios de objetivos

-Funciona bien en proyectos de innovación.

-Es sencillo, ya que sigue los pasos intuitivos necesarios a la hora de desarrollar el software.

-Seguimiento detallado en cada una de las fases.

Desventajas

-La evaluación de riesgos es compleja

-Excesiva flexibilidad para algunos proyectos

-Estamos poniendo a nuestro cliente en una situación que puede ser muy incómoda para él.

-Nuestro cliente deberá ser capaz de describir y entender a un gran nivel de detalle para poder acordar un alcance del proyecto con él.

Los MSF (MICROSOFT SOLUTION FRAMEWORK)

MSF es una serie de modelos que puede adaptarse a cualquier proyecto de tecnología de información. El modelo de equipos de MSF tiene seis roles que corresponden a las metas principales de un proyecto y son responsables por las mismas. Cada rol puede estar compuestos por una o más personas, la estructura circular del modelo, con óvalos del mismo tamaño para todos los roles, muestra que no es un modelo jerárquico y que todos los roles son igualmente importantes en su aporte al proyecto.

Todo proyecto es separado en cinco principales fases:

-Visión y Alcances.

-Planificación.

-Desarrollo.

-Estabilización.

-Implantación.

Visión y Alcances:

El equipo debe tener una visión clara de lo que quisiera lograr para el cliente y ser capaz de indicarlo en términos que motivarán a todo el equipo y al cliente.

Planificación:

El equipo prepara las especificaciones funcionales, realiza el proceso de diseño de la solución, y prepara los planes de trabajo, estimaciones de costos y cronogramas de los diferentes entregables del proyecto.

Desarrollo:

Durante esta fase el equipo realice la mayor parte de la construcción de los componentes.

Estabilización:

El equipo se enfoca en priorizar y resolver errores y preparar la solución para el lanzamiento.

Implantación:

Durante esta fase el equipo implanta la tecnología base y los componentes relacionados, estabiliza la instalación, traspasa el proyecto al personal soporte y operaciones, y obtiene la aprobación final del cliente.

METODOLOGÍAS ÁGILES.

Se basa en dos aspectos puntuales, el retrasar las decisiones y la planificación adaptativa; permitiendo potencia aún más el desarrollo de software a gran escala.

Entre los principales métodos ágiles tenemos el XP (eXtreme Programming), Scrum, Iconix, Cristal Methods, AUP. Estas metodologías ponen de relevancia que la capacidad de respuesta a un cambio es más importante que el seguimiento estricto de un plan.

La planificación adaptativa permite estar preparados para el cambio ya que lo hemos introducido en nuestro proceso de desarrollo, además hacer una planificación adaptativa consiste en tomar decisiones a lo largo del proyecto, estaremos transformando el proyecto en un conjunto de proyectos pequeños.

EXTREME PROGRAMMING (XP)

Las características fundamentales del método son:

Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.

Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.

Programación por parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone que la mayor calidad del código escrito de esta manera -el código es revisado y discutido mientras se escribe- es más importante que la posible pérdida de productividad inmediata.

Frecuente interacción del equipo de programación con el cliente o usuario. Se recomienda que un representante del cliente trabaje junto al equipo de desarrollo.

Corrección de todos los errores antes de añadir nueva funcionalidad. Hacer entregas frecuentes.

Refactorización del código, es decir, reescribir ciertas partes del código para aumentar su

legibilidad y mantenibilidad pero sin modificar su comportamiento. Las pruebas han de garantizar que en la refactorización no se ha introducido ningún fallo.

Propiedad del código compartida: en vez de dividir la responsabilidad en el desarrollo de cada módulo en grupos de trabajo distintos, este método promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto. Las frecuentes pruebas de regresión garantizan que los posibles errores serán detectados.

Simplicidad en el código: Cuando todo funcione se podrá añadir funcionalidad si es necesario. La programación extrema apuesta que en más sencillo hacer algo simple y tener un poco de trabajo extra para cambiarlo si se requiere, que realizar algo complicado y quizás nunca utilizarlo.

Ventajas

Apropiado para entornos volátiles

Estar preparados para el cambio, significa reducir su coste.

Planificación más transparente para nuestros clientes, conocen las fechas de entrega de funcionalidades. Vital para su negocio

Permitirá definir en cada iteración cuales son los objetivos de la siguiente

Permite tener realimentación de los usuarios muy útil.

La presión está lo largo de todo el proyecto y no en una entrega final.

Desventajas

Delimitar el alcance del proyecto con nuestro cliente

AUP (AGIL UNIFIED PROCESS)

El AUP es un acercamiento aerodinámico a desarrollo del software basado en el Proceso Unificado Rational de IBM (RUP), basado en disciplinas y entregables incrementales con el tiempo. El ciclo de vida en proyectos grandes es serial mientras que en los pequeños es iterativo.

Las disciplinas son:

-Modelado

-Implementación

-Prueba

-Despliegue

-Administración de la configuración

-Administración o gerencia del Proyecto

-Entorno

Scrum:

Scrum es un proceso ágil y liviano que sirve para administrar y controlar el desarrollo de software. Construir software de calidad y la gestión de un proyecto. Define cuáles son las características que debe tener el producto a construir. Scrum tiene un conjunto de reglas muy pequeño y muy simple y está basado en los principios de inspección continua, adaptación, auto-gestión e innovación.

ICONIX

El proceso de ICONIX maneja casos de uso, como el RUP, pero le falta mucho para llegar al nivel del RUP. También es relativamente pequeño y firme, como XP, pero no desecha el análisis y diseño que hace XP.