Modelos de procesos del software

Los estándares establecen los diferentes procesos implicados a la hora de desarrollar y mantener un sistema desde que surge la idea o necesidad de desarrollar las aplicaciones hasta que éstas se retiran de explotación. Sin embargo, ninguno impone un modelo de procesos concreto (modelo de ciclo de vida) ni cómo realizar las diferentes actividades incluidas en cada proceso, por lo que cada empresa deberá utilizar los métodos, técnicas y herramientas que considere oportuno.
Por su naturaleza, los modelos son simplificaciones; por lo tanto, un modelo de procesos del software es una simplificación o abstracción de un proceso real.
Podemos definir un modelo de procesos del software como una representación abstracta de alto nivel de un proceso software.
Cada modelo es una descripción de un proceso software que se presenta desde una perspectiva particular. Alternativamente, a veces se usan los términos ciclo de vida y Modelo de ciclo de vida.
Cada modelo describe una sucesión de fases y un encadenamiento entre ellas. Según las fases y el modo en que se produzca este encadenamiento, tenemos diferentes modelos de proceso. Un modelo es más adecuado que otro para desarrollar un proyecto dependiendo de un conjunto de características de éste.
Existe una gran variedad de modelos diferentes entre los que tenemos los que se describen a continuación.

Modelo en cascada o lineal secuencial

Características del modelo:
Primer modelo empleado (Royce, 1970), también denominado ciclo de vida clásico y modelo lineal secuencial.
Consiste en la ejecución secuencial de una serie de fases que se suceden, lo que da nombre al modelo.
Cada fase genera documentación para la siguiente. Esta documentación debe ser aprobada.
Una fase no comienza hasta que la anterior ha terminado.
Requiere disponer de unos requisitos completos y precisos al principio del desarrollo.

Presenta una serie de ventajas:

Se debe tener en cuenta que fue el primer modelo empleado, y por lo tanto es mejor que ninguno.
Facilita la gestión del desarrollo.

Así como una serie de inconvenientes:

En general, establecer todos los requisitos al principio del proceso de desarrollo es un mito inalcanzable: Los usuarios no pueden imaginarse lo que quieren hasta que no ven un sistema funcionando.
Los requisitos no se pueden congelar mientras dura el desarrollo. El mercado cambia, todo cambia.
El usuario debe esperar mucho tiempo hasta ver los resultados: Se tarda mucho tiempo en pasar por todo el ciclo (hasta que no termina una fase no empieza la siguiente) y el sistema en funcionamiento no estará disponible hasta el final del proceso, eso sí, será el sistema completo.
Los errores de análisis y diseño son costosos de eliminar, y se propagan a las fases siguientes con un efecto conocido como bola de nieve.
Se genera mucho mantenimiento inicial debido al período de congelación de requisitos y éste recae, en su mayor parte, sobre el código fuente, que en consecuencia se va deteriorando y resultando cada vez más difícil de mantener.

Las características del proyecto que hacen adecuado el uso de este modelo son que:

Se disponga de unos requisitos completos y consistentes al principio del desarrollo.
Sea un proyectos pequeño, en el que el período de congelación de los requisitos es corto, o un proyecto con unos requisitos bastante estables.

Modelo en cascada con prototipado desechable

Trata de resolver algunos de los inconvenientes que presenta el modelo en cascada, fundamentalmente el problema que representa disponer de unos requisitos completos y consistentes al principio del desarrollo y la detección de errores en la fase de integración provenientes de la fase de análisis.
Características del modelo: Divide el ciclo de vida en dos partes.
En la parte A, se construye un prototipo rápido o desechable, que ayudará a refinar y validar los requerimientos.
En la parte B, el desarrollo posterior prosigue en cascada.

Presenta una serie de ventajas:

Se dispone desde muy temprano de unos requerimientos completos y consistentes.
Facilita el desarrollo en lo que respecta a la interfaz de usuario.
Ayuda a mitigar el efecto bola de nieve al reducir el mantenimiento como consecuencia de disponer de unas especificaciones completas y correctas, aunque no lo elimina al continuar el desarrollo en cascada.

Así como de inconvenientes:

Es frecuente arrastrar malas decisiones (de diseño, de planificación, etc.) que sólo eran apropiadas para la obtención rápida del prototipo y cuya implementación real puede ser muy costosa.
El prototipo sólo puede ser aprovechado en su aspecto externo. Los aspectos funcionales son muy reducidos.
El tiempo invertido en la construcción del prototipo y el coste adicional de la inversión que supone la creación de un producto desechable.
Las características del proyecto que hacen adecuado el uso de este modelo son que:
El usuario no tenga un buen conocimiento del dominio.
Sea un proyecto corto.
Haya pocos cambios en los requisitos.