XP :: EXTREME PROGRAMMING

Extreme Pogramming es una metodología de desarrollo de software ágil que define un proceso de trabajo incremental y colaborativo. La aplicación de esta metodología está centrada en la comunicación e interacción entre los miembros del equipo, y tiene la capacidad de adaptarse fácilmente a los cambios que puedan surgir durante la etapa de desarrollo del producto de software.

Los equipos de trabajo XP incluyen a todos los interesados en el proyecto, es decir, tanto al cliente como a los programadores. Será responsabilidad del cliente decidir el alcance del proyecto, la prioridad de las funciones, la composición de cada una de las entregas y las fechas de finalización de cada etapa. Los programadores en cambio deberán hacer la estimación de tiempos de desarrollo para cada una de las funciones, las consecuencias técnicas y decidirán la forma de trabajo del equipo y programación temporal de las tareas dentro de cada iteración.

Aunque el cliente es parte del equipo, las prácticas de XP se enfocan en mantener la división planteada anteriormente con el objetivo de mantener una clara visión de las consecuencias de cada una de las decisiones que tome, ya que será él quien deberá convivir con el producto final.

Para que un equipo de XP sea productivo, deberán trabajar en un entorno saludable y aconseja no sobrecargar las horas de trabajo. Si el cliente desea un conjunto de funcionalidades para una fecha particular, pero el equipo estima que no puede cumplirla, tendrá las siguientes opciones: disminuir la cantidad de funcionalidades requeridas; aceptar una fecha de entrega posterior; invertir más dinero en la búsqueda de otras alternativas; o buscar otro equipo de desarrollo.

En rasgos generales, XP consta de tres etapas principales:

- ***Planeamiento de una entrega***, donde el cliente informará sus necesidades escribiendo ***historias de usuario***, y los programadores realizarán una evaluación y estimación de tiempos y tareas para cada historia de forma que el cliente pueda decidir la prioridad con que se implementarán.
- ***Iteración***, el cliente escribirá casos de prueba y responderá a las inquietudes de los programadores.
- ***Entrega***, los programadores instalaran la solución y el cliente dará su aprobación.

LAS PRÁCTICAS DE XP

XP define 12 reglas que deben cumplirse todas al mismo tiempo para asegurar el correcto funcionamiento de la metodología. Estas reglas pueden verse desde tres enfoques diferentes según su ámbito de aplicación, aunque algunas se solapen:

PROGRAMACIÓN

- Diseños simples
- Testing constante
- Refactorización
- Uso de convenciones de codificación

Los programadores deben escribir código de forma incremental, y aplicando el método de "*test-first programming*", es decir, escribiendo pequeñas *test-units* y luego código suficiente como para satisfacer el dicho test.

La idea general detrás de XP es: generar un código simple y un sistema flexible manteniéndolo tan simple como se pueda. Para ello el *refactoring* es crucial. Se aconseja hacerlo de forma constante y durante toda la vida del proyecto, con el objetivo de eliminar *bugs* y mejor el rendimiento general del programa.

En cuanto a la documentación, XP es muy flexible. Solo la requiere cuando los miembros del equipo la consideran apropiada. Esto sumado a la política del *refactoring* constante converge en la necesidad de definir convenciones de codificación y apegarse estrictamente a ellas para evitar problemas en la comunicación a través del código fuente.

PRÁCTICAS DE EQUIPO

- Propiedad colectiva
- Integración constante
- Metáfora del sistema
- Uso de convenciones de codificación
- Semana de 40h laborales
- Programación en pares
- Entregas pequeñas

XP es una metodología orientada al trabajo en equipo, y define una serie de estrategias para: mejorar la interacción entre los integrantes del equipo, asegurar su bienestar, y lograr el objetivo de la empresa y sus clientes: "*mayor productividad*".

La política de *propiedad colectiva del código* ayuda a evitar pretextos al momento de retomar código de iteraciones anteriores e introducir modificaciones en él. La responsabilidad es compartida entre todos los integrantes del equipo, y por consiguiente nadie podrá reusarse a modificarlo.

La regla de integración constante brinda un mecanismo verificación continua de *todo* el código producido. Cada vez que un programador termina su tarea,

deberá instalar el nuevo código y asegurarse que el sistema sigue verificando los test anteriores, más los que se definieron para esa tarea en particular. Por esta razón se recomienda además generar test automatizado.

El Proceso

- Cliente in-situ
- Testing constante
- Entregas pequeñas
- Planeación de entregas e Iteraciones

En XP el cliente juega un rol fundamental. El cliente definirá el alcance del proyecto, lo ajustará de acuerdo a los avances del equipo, responderá las inquietudes de los programadores, y tendrá la posibilidad de tomar decisiones que afecten el curso de todo el desarrollo del producto. Teniendo en cuenta esto, un *cliente in-situ* se traduce en una mejorara significativa en la productividad. En vez de tener que esperar horas o incluso días por una respuesta, los programadores reciben respuestas a sus inquietudes en el momento.

Una entrega del sistema es una versión, con suficientes funcionalidades como para funcionar fuera del grupo de desarrollo, es decir, que se puede instalar en la empresa del cliente. Durante el proceso de planeación de las entregas, los programadores estimarán el tiempo de programación de las historias que brinde el cliente, y este a su vez las ordenará según las necesidades de su empresa.

El objetivo principal de esta etapa es lograr que el cliente tenga una visión general del producto de software que se desarrollará y la planeación temporal global para llevarlo a cabo.

La etapa siguiente, es el planeamiento de las iteraciones. En ella los programadores decidirán que historias implementarán, las descompondrán en tareas y asignaran estas a los programadores. En esta fase el cliente puede solicitar que se las historias implementen en el orden que considere conveniente a sus fines e incluso puede introducir nuevas, siempre y cuando le brinde a los programadores el tiempo necesario para hacer las estimaciones pertinentes.

Una vez que la iteración está organizada y las historias divididas en tareas, cada pareja de programadores deberá decidir que cuales implementará. Para esto deberá estimar el tiempo que le demandará cada tarea y seleccionar las que considere apropiadas para ese día. Si al finalizar la jornada no pudiesen terminar alguna tarea, se descarta el código generado para ella y esta regresa a la lista de pendientes. Por otra parte, mientras los programadores hacen su trabajo, el cliente estará disponible para responder a las preguntas de los programadores y escribirá los *test de aceptación* para cada una de las historias en proceso de codificación. Esta tarea la puede realizar solo o con la ayuda de un programador, según lo considere conveniente.

ROLES DE MANAGEMENT

XP define tres roles de suma importancia, que de acuerdo con el tamaño del equipo, pueden ser ejecutados por una o más personas.

MANAGER

El manager es el responsable del equipo. Es la cara visible al exterior, es quien conforma el equipo, el que obtiene los recursos, coordina a los integrantes del equipo y es el mediador en conflictos.

Su deber es proveer a los integrantes del equipo con los materiales necesarios para su trabajo, ya sea hardware, software o incluso la contratación de consultores externos si fuese necesario. Será quien organice las reuniones internas y quien comunicará los resultados del equipo a los interesados externos.

TRACKER

La persona que se encargue de este rol deberá aplicar métricas para evaluar la evolución del equipo y comunicar sus conclusiones al manager. Deberá medir los resultados de la aplicación de los test, el cumplimiento de las proyecciones de cada iteración y el curso general de las entregas.

COACH

El coach, al igual que el cliente, deberá ser *in-situ* y estar disponible para: responder las dudas de los programadores, ejecutar tests, y reajustar las prioridades del equipo. Su función es la de un mentor de los programadores.

Sus actividades se concentran en controlar los progresos del equipo. Es conveniente que no tenga demasiadas tareas de programación asignadas o bien, que no tenga ninguna.

Su responsabilidad será asegurar que se cumplan con los *standars* de calidad del software y velar por la aplicación del proceso o si lo considera conveniente, introducir cambios en la metodología.