Universidad Nacional de Río Cuarto

Facultad de Ciencias Exactas, Físico y Químicas

Departamento de Computación

Análisis y Diseño de Sistemas

“Extreme Programming”

Autores:

Blanco, Susana

Oviedo, Silvia.

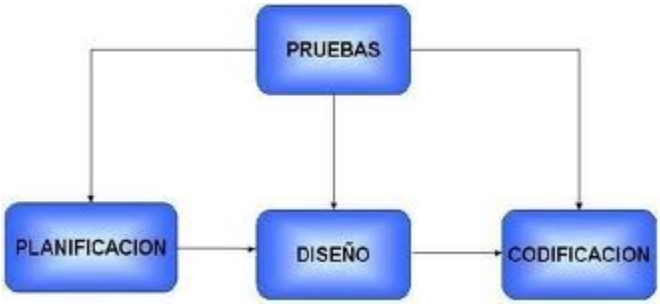
Metodología: Programación Extrema

Introducción

En que consiste esta metodología?

(XP) surge como una nueva manera de encarar proyectos de software, proponiendo una metodología basada esencialmente en la simplicidad y agilidad.

Las metodologías de desarrollo de software tradicionales(ciclo de vida en cascada, evolutivo, en espiral, iterativo, etc.) aparecen,comparados con los nuevos métodos propuestos en XP, como pesados y poco eficientes.
La programación extrema apuesta que es 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. Las características más importantes de este método son:



- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- Pruebas unitarias continuas,
- Programación en parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto.
- Frecuente integración 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.

- Entregas frecuentes.
- Reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad pero sin modificar su comportamiento
- Código compartido: promueve el que todo el personal pueda corregir y extender cualquier parte del proyecto.
- Simplicidad en el código: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario.

La metodología XP define cuatro variables para cualquier proyecto de software: ***costo, tiempo, calidad*** y ***alcance***. Además, se especifica que, de estas cuatro variables, sólo tres de ellas podrán ser fijadas arbitrariamente por actores externos al grupo de desarrolladores (clientes y jefes de proyecto). La variable restante podrá ser establecido por el equipo de desarrollo,

Fase de exploración

En esta fase, el cliente define lo que necesita mediante la redacción de sencillas "historias de usuarios". Los programadores estiman los tiempos de desarrollo en base a esta información. Debe quedar claro que las estimaciones realizadas en esta fase son primarias (ya que estarán basadas en datos de muy alto nivel), y podrían variar cuando se analicen más en detalle en cada iteración.
Esta fase dura típicamente un par de semanas, y el resultado es una visión general del sistema, y un plazo total estimado.

Fase de planificación

La planificación es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario, y, asociadas a éstas, las entregas. Típicamente esta fase consiste en una o varias reuniones grupales de planificación. El resultado de esta fase es un Plan de Entregas, o *"Release Plan"*, como se detallará en la sección "Reglas y Practicas".

Fase de iteraciones

Esta es la fase principal en el ciclo de desarrollo de XP. Las funcionalidades son desarrolladas en esta fase, generando al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración. Como las historias de usuario no tienen suficiente detalle como para permitir su análisis y desarrollo, al principio de cada iteración se realizan las tareas necesarias de análisis, recabando con el cliente todos los datos que sean

necesarios. El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo.
Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance.

Fase de puesta en producción

Si bien al final de cada iteración se entregan módulos funcionales y sin errores, puede ser deseable por parte del cliente no poner el sistema en producción hasta tanto no se tenga la funcionalidad completa.
En esta fase no se realizan más desarrollos funcionales, pero pueden ser necesarias tareas de ajuste ("fine tuning").

Reglas y Practicas

La metodología XP tiene un conjunto importante de reglas y se pueden agrupar en:

- Reglas y prácticas para la Planificación
- Reglas y prácticas para el Diseño
- Reglas y prácticas para el Desarrollo
- Reglas y prácticas para las Pruebas

Planificación

La metodología XP plantea la planificación como un dialogo continuo entre las partes involucradas en el proyecto, incluyendo al cliente, a los programadores y a los coordinadores o gerentes.comienza recopilando "Historias de usuarios", las que sustituyen a los tradicionales "casos de uso".
los programadores evalúan rápidamente el tiempo ,se realizan pequeños programas de prueba (*"spikes"*) se organiza una reunión de planificación, con los diversos actores del proyecto (cliente, desarrolladores, gerentes), a los efectos de establecer un plan o cronograma de entregas (*"Release Plan"*) en los que todos estén de acuerdo. Una vez acordado este cronograma, comienza una fase de iteraciones.

Según Martín Fowler (uno de los firmantes del "Agile Manifesto"), los planes en XP
se diferencian de las metodologías tradicionales en tres aspectos .

- Simplicidad del plan.
- Los planes son realizados por las mismas personas que realizarán el trabajo.
- Los planes no son predicciones del futuro, sino simplemente la mejor estimación de cómo saldrán las cosas. tienen que ser cambiados cuando las circunstancias lo requieren.

Historias de usuarios

Las “Historias de usuarios” (*“User stories”*) sustituyen a los documentos de especificación funcional, y a los “casos de uso”. Estas “historias” son escritas por el cliente, en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. La diferencia más importante entre estas historias y los tradicionales documentos de especificación funcional se encuentra en el nivel de detalle requerido. Cuando llegue el momento de la implementación, los desarrolladores dialogarán directamente con el cliente para obtener todos los detalles necesarios.
Las historias de usuarios deben poder ser programadas en un tiempo entre una y tres semanas. Si la estimación es superior a tres semanas, debe ser dividida en dos o más historias. Si es menos de una semana, se debe combinar con otra historia.

Plan de entregas (“Release Plan”)

El cronograma de entregas establece qué historias de usuario serán agrupadas para conformar una entrega, y el orden de las mismas. Este cronograma será el resultado de una reunión entre todos los actores del proyecto (cliente, desarrolladores, gerentes, etc.).
El cronograma de entregas se realiza en base a las estimaciones de tiempos de desarrollo realizadas por los desarrolladores.
Luego de algunas iteraciones es recomendable realizar nuevamente una reunión con los actores del proyecto, para evaluar nuevamente el plan de entregas y ajustarlo si es necesario.

Plan de iteraciones (“Iteration Plan”)

Cada historia de usuario se traduce en tareas especificas de programación.
se establecen las pruebas de aceptación. Estas pruebas se realizan al final del ciclo en el que se desarrollan, Las pruebas de aceptación que hayan fallado en el ciclo anterior son analizadas para evaluar su corrección, así como para prever que no vuelvan a ocurrir.

Reuniones diarias de seguimiento (“Stand-up meeting”)

El objetivo de tener reuniones diarias es mantener la comunicación entre el equipo, y compartir problemas y soluciones. Para no quitar tiempo innecesario del equipo, se sugiere realizar estas reuniones en círculo y de pie.

Diseño

La metodología XP hace especial énfasis en los diseños simples y claros.

Simplicidad

Un diseño simple se implementa más rápidamente que uno complejo. Por ello XP propone implementar el diseño más simple posible que funcione.

Soluciones "spike"

Cuando aparecen problemas técnicos, o cuando es difícil de estimar el tiempo para implementar una historia de usuario, pueden utilizarse pequeños programas de prueba (llamados "spike"1), para explorar diferentes soluciones.

Recodificación

La recodificación (*"refactoring"*) consiste en escribir nuevamente parte del código de un programa, sin cambiar su funcionalidad, a los efectos de hacerlo más simple, conciso y/o entendible.

Las metodologías de XP sugieren recodificar cada vez que sea necesario. Si bien, puede parecer una pérdida de tiempo innecesaria en el plazo inmediato, los resultados de ésta práctica tienen sus frutos en las siguientes iteraciones, cuando sea necesario ampliar o cambiar la funcionalidad.

Metáforas

Una "metáfora" es algo que todos entienden, sin necesidad de mayores explicaciones. La metodología XP sugiere utilizar este concepto como una manera sencilla de explicar el propósito del proyecto, y guiar la estructura y arquitectura del mismo. Por ejemplo, puede ser una guía para la nomenclatura de los métodos y las clases utilizadas en el diseño del código. Tener nombres claros, que no requieran de mayores explicaciones, redunda en un ahorro de tiempo.

Es muy importante que el cliente y el grupo de desarrolladores estén de acuerdo y compartan esta "metáfora", para que puedan dialogar en un "mismo idioma.

Desarrollo del código

Disponibilidad del cliente

Uno de los requerimientos de XP es tener al cliente disponible durante todo el proyecto. No solamente como apoyo a los desarrolladores, sino formando parte del grupo. El involucramiento del cliente es fundamental para que pueda desarrollarse un proyecto con la metodología XP.

Al comienzo las historias de usuarios. son expresamente cortas y de "alto nivel", no

contienen los detalles necesarios para realizar el desarrollo del código. Estos detalles deben ser proporcionados por el cliente, y discutidos con los desarrolladores, durante la etapa de desarrollo. Cliente "cara a cara" a los desarrolladores.

Si bien esto parece demandar del cliente recursos por un tiempo prolongado, debe

tenerse en cuenta que en otras metodologías este tiempo es insumido por el cliente en realizar los documentos detallados de especificación.

Uso de estándares

Si bien esto no es una idea nueva, XP promueve la programación basada en estándares, de manera que sea fácilmente entendible por todo el equipo, y que facilite la recodificación.

Programación dirigida por las pruebas (*"Test-driven programming"*)

En las metodologías tradicionales, la fase de pruebas, incluyendo la definición de los tests, es usualmente realizada sobre el final del proyecto, o sobre el final del desarrollo de cada módulo. La metodología XP propone un modelo inverso, en el que, lo primero que se escribe son los test que el sistema debe pasar. Las pruebas a los que se refiere esta práctica, son las pruebas unitarias, realizados por los desarrolladores. La definición de estos test al comienzo, condiciona o "dirige" el desarrollo.

Programación en pares

XP propone que se desarrolle en pares de programadores, ambos trabajando juntos en un mismo ordenador. Si bien parece que ésta práctica duplica el tiempo asignado al proyecto (y por ende, los costos en recursos humanos), al trabajar en pares se minimizan los errores y se logran mejores diseños, compensando la inversión en horas. El producto obtenido es por lo general de mejor calidad que cuando el desarrollo se realiza por programadores individuales.
Este sobre costo es rápidamente pagado por la mejor calidad obtenida en el producto final.

la programación en pares tiene las siguientes ventajas:

- La mayoría de los errores se descubren en el momento en que se codifican, ya que el código es permanentemente revisado por dos personas.

- La cantidad de defectos encontrados en las pruebas es estadísticamente menor.

- Los diseños son mejores y el código más corto.

- El equipo resuelve problemas en forma más rápida.

- Las personas aprenden significativamente más, acerca del sistema y acerca de desarrollo de software.

- El proyecto termina con más personas que conocen los detallas de cada parte del código.

- Las personas aprenden a trabajar juntas, generando mejor dinámica de grupo y haciendo que la información fluya rápidamente.

- Las personas disfrutan más de su trabajo.

Integraciones permanentes

Todos los desarrolladores necesitan trabajar siempre con la “última versión”.
Realizar cambios o mejoras sobre versiones antiguas causan graves problemas, y
retrasan al proyecto. Es por eso que XP promueve publicar lo antes posible las nuevas versiones, aunque no sean las últimas, siempre que estén libres de errores. Idealmente, todos los días deben existir nuevas versiones publicadas.
Para evitar errores, solo una pareja de desarrolladores puede integrar su código a
la vez.

Propiedad colectiva del código

En un proyecto XP, todo el equipo puede contribuir con nuevas ideas que apliquen a cualquier parte del proyecto. Asimismo, cualquier pareja de programadores puede cambiar el código que sea necesario para corregir problemas, agregar funciones o recodificar.
En este caso, quienes encuentran un problema, o necesitan desarrollar una nueva función, pueden resolverlo directamente, sin necesidad de “negociar” con el “dueño” o autor del módulo (ya que, de hecho, este concepto no existe en XP).
Si es necesario dialogar y convencer al encargado de cada módulo, posiblemente la solución no se pueda implementar, por lo menos en tiempos razonables. Cualquier pareja de programadores puede tomar la responsabilidad de este cambio. Los testeos permanentes deberían de aseguran que los cambios realizados cumplen con lo
requerido, y además, no afectan al resto de las funcionalidades.

Ritmo sostenido

La metodología XP indica que debe llevarse un ritmo sostenido de trabajo.
se desea establecer con esta práctica es el de planificar el trabajo de manera de mantener un ritmo constante y razonable, sin sobrecargar al equipo.
El trabajo extra desmotiva inmediatamente al grupo e impacta en la calidad del producto.

Pruebas

Pruebas unitarias

Las pruebas unitarias son una de las piedras angulares de XP. Todos los módulos deben de pasar las pruebas unitarias antes de ser liberados o publicados. Que todo código liberado pase correctamente las pruebas unitarias es lo que habilita que funcione la propiedad colectiva del código. En este sentido, el sistema y el conjunto de pruebas debe ser guardado junto con el código, para que pueda ser utilizado por otros desarrolladores, en caso de tener que corregir, cambiar o re codificar parte del mismo.

Detección y corrección de errores

Cuando se encuentra un error (“bug”), éste debe ser corregido inmediatamente, y se deben tener precauciones para que errores similares no vuelvan a ocurrir.
Asimismo, se generan nuevas pruebas para verificar que el error haya sido resuelto.

Pruebas de aceptación

Las pruebas de aceptación son creadas en base a las historias de usuarios.
El cliente debe especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada.
Las pruebas de aceptación son consideradas como “pruebas de caja negra” (*“Black box system tests”*). Los clientes son responsables de verificar que los resultados de éstas pruebas sean correctos.
Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación.

Valores en XP

XP se basa en cuatro valores, que deben estar presentes en el equipo de desarrollo para que el proyecto tenga éxito:

Comunicación

Muchos de los problemas que existen en proyectos de software se deben a problemas de comunicación entre las personas. La comunicación permanente es fundamental en XP. Dado que la documentación es escasa, el diálogo frontal, cara a cara, entre desarrolladores, gerentes y el cliente es el medio básico de comunicación. Una buena comunicación tiene que estar presente durante todo el proyecto.

Simplicidad

XP, como metodología ágil, apuesta a la sencillez, en su máxima expresión.

Sencillez en el diseño, en el código, en los procesos, etc. La sencillez es esencial para que todos puedan entender el código, y se trata de mejorar mediante re codificaciones continuas.

Retroalimentación

La retroalimentación debe funcionar en forma permanente. El cliente debe brindar retroalimentación de las funciones desarrolladas, de manera de poder tomar sus comentarios para la próxima iteración, y para comprender, cada vez más, sus necesidades.
Los resultados de las pruebas unitarias son también una retroalimentación permanente que tienen los desarrolladores acerca de la calidad de su trabajo.

Coraje

Cuando se encuentran problemas serios en el diseño, o en cualquier otro aspecto, se debe tener el coraje suficiente como para encarar su solución, sin importar que tan difícil sea. Si es necesario cambiar completamente parte del código, hay que hacerlo, sin importar cuánto tiempo se ha invertido previamente en el mismo.

Aplicabilidad

Por lo general, cada metodología tiene sus escenarios de aplicabilidad. Ninguna de las metodologías de desarrollo de software son buenas para todos los proyectos. Para proyectos medianos, y en los que las especificaciones no se puedan obtener hasta luego de comenzado el proyecto, XP puede ser la metodología recomendada.
La metodología XP aplica a equipos relativamente pequeños. Si bien no hay un consenso en el número máximo de desarrolladores, todos parecen coincidir en números no mayores a 20.
Por otro lado, el entorno físico en el que se realizan los desarrollos deben ser adecuados a la metodología.
La participación e involucramiento del cliente en el proceso es fundamental. El cliente debe conocer y aprobar la metodología, y destinar los recursos necesarios.
De otra manera, el proyecto no podrá ser llevado a cabo.
Finalmente, los participantes del equipo de desarrollo deben estar compenetrados con el proyecto y su metodología. Deben aceptar compartir sus códigos y ser responsables por el código que escribieron otros.

Críticas

Una de las críticas a XP es la dificultad de estimar cuánto va a costar un proyecto. Dado que el alcance del mismo no está completamente definido al comienzo, y que la metodología XP es expresamente abierta a los cambios

durante todo el proceso, se torna sumamente difícil poder realizar un presupuesto previo.
Por otro lado, muchas de las prácticas sugeridas por XP son compartibles y utilizadas, de una u otra forma, en muchas otras metodología.
Otra posible crítica a XP refiere a la baja documentación, pensando en el posterior mantenimiento del sistema. Si bien durante el proyecto el equipo tiene en mente todas sus particularidades, hay que prever que pasará luego de entregado, cuando el equipo se disuelva, y sea necesario realizar algún cambio o mejora. XP propone la re codificación permanente, para asegurar la claridad y simplicidad del código. Sin embargo, es posible que aún un código simple y claro no baste cuando otro equipo de trabajo tenga que tomar el sistema y cambiarlo. Es posible que sea necesario mantener cierta documentación, aunque es compartible que ésta debe ser la mínima necesaria.
También se critica la "auto-realimentación" de XP, ya que sus prácticas requieren ser aplicadas en su totalidad para que el método sea viable: No disponer de requerimientos detallados escritos lleva a mantener un diseño simple. Un diseño simple lleva a que sea necesario re codificar constantemente. La re codificación necesita de permanentes pruebas unitarias. Las pruebas unitarias fallarán poco, gracias a la programación de a pares, los que requieren del cliente en sitio para poder conocer los detalles a implementar.
Finalmente, tener al cliente en sitio hace irrelevante disponer de requerimientos escritos detallados, cerrando el círculo. De acuerdo a esto, una falla en cualquiera de estas prácticas hará que todo el proyecto fracase.

Conclusiones

En una encuesta realizada sobre 45 proyectos realizados con XP en 2001, se concluye que:

- Casi todos los proyectos se categorizaron como exitosos.
- El 100% de los desarrolladores encuestados afirmaron que volvería a utilizar la metodología XP en el siguiente proyecto si fuera apropiado.
- Las frecuentes ausencias del cliente fueron identificadas como el mayor riesgo en los proyectos.
- Los problemas más comunes fueron "barreras psicológicas", como por ejemplo el escepticismo de la línea gerencial, la filosofía de la empresa desarrolladora que no permitía tener al cliente en sitio, o que algunos desarrolladores se oponía al trabajo en parejas.
- Los elementos más útiles de XP fueron la propiedad colectiva del código, las pruebas y la integración continua. Las más críticas fueron la metáfora y el cliente en sitio.

Una encuesta más reciente, realizada en 2005, presenta como resultados destacables que el 54.9% de los encuestados dicen tener amplia experiencia en XP, pero solo el 23.2% lo utiliza en la mayoría de los proyectos. Por otro lado, las prácticas marcadas como más útiles son el diseño simple, la programación dirigida por los test y el uso de estándares, en ese orden.

Cualquiera de estas prácticas podría aplicar perfectamente a otras metodologías.
XP es una de las nuevas metodologías ágiles. Siendo de reciente divulgación, está comenzando a ser utilizada por algunos, y criticada por otros. Es claro que no existe una metodología única para garantizar el éxito de cualquier proyecto de desarrollo de software, y esto aplica también a XP. Toda metodología requiere de cierta adaptación al proyecto, al cliente y a la idiosincrasia de la empresa desarrolladora.
Las metodologías tradicionales han intentado abarcar la mayor cantidad de situaciones, pero exigen un esfuerzo considerable en documentación y gerenciamiento de proyecto, que muchas veces genera una sobrecarga inaceptable en proyectos pequeños y medianos. XP propone una metodología ágil y concreta, aunque requiere de una nueva manera de pensar, ver y hacer las cosas, tanto por parte de los gerentes (responsables de la rentabilidad general del proyecto), como de los desarrolladores y también del cliente. La aplicabilidad de ésta metodología a cada proyecto debería ser analizada en cada caso, teniendo en cuenta el tamaño y tipo de proyecto, así como sus ventajas y desventajas.