La programación extrema o EXtreme Programming (PX) es un enfoque de la Ingeniería de Software formulado por Kent Beck el 6 de marzo de 1996. Es el más destacado de los procesos ágiles de desarrollo de software.

Principios básicos

La Programación Extrema se basa en 12 principios básicos agrupados en cuatro categorías:

1. Retroalimentación a escala fina.
    1) El principio de pruebas: se tiene que establecer un período de pruebas de aceptación del programa (llamado también período de caja negra) donde se definirán las entradas al sistema y los resultados esperados de estas entradas.
    2) Proceso de planificación: en esta fase, el usuario tendrá que escribir sus necesidades, definiendo las actividades que realizará el sistema. Se creará un documento llamado Historias del usuario (User Stories). Entre 20 y 80 historias (todo dependiendo de la complejidad del problema) se consideran suficientes para formar el llamado Plan de Liberación, el cual define de forma específica los tiempos de entrega de la aplicación para recibir retroalimentación por parte del usuario.
    3) El cliente en el sitio: se le dará poder para determinar los requerimientos, definir la funcionalidad, señalar las prioridades y responder las preguntas de los programadores. Esta fuerte interacción cara a cara con el programador disminuye el tiempo de comunicación y la cantidad de documentación, junto con los altos costes de su creación y mantenimiento. Este representante del cliente estará con el equipo de trabajo durante toda la realización del proyecto.
    4) Programación en parejas (pair-programing): uno de los principios más radicales y en el que la mayoría de gerentes de desarrollo pone sus dudas. Requiere que todos los programadores XP escriban su código en parejas, compartiendo una sola máquina. De acuerdo con los experimentos, este principio puede producir aplicaciones más buenas, de manera consistente, a iguales o menores costes.
2. Proceso continuo en lugar de por lotes
    1) Integración continua: permite al equipo hacer un rápido progreso implementando las nuevas características del software. En lugar de crear builds (o versiones) estables de acuerdo a un cronograma establecido, los equipos de programadores XP pueden reunir su código y reconstruir el sistema varias veces al día. Esto reduce los problemas de integración comunes en proyectos largos y estilo cascada.
    2) Refactorización: permite a los equipos de programadores XP mejorar el diseño del sistema a través de todo el proceso de desarrollo. Los programadores evalúan continuamente el diseño y re codifican lo necesario. La finalidad es mantener un sistema enfocado a proveer el valor de negocio mediante la minimización del código duplicado y/o ineficiente.
    3) Entregas pequeñas: colocan un sistema sencillo en producción rápidamente que se actualiza de forma rápida y constante permitiendo que el verdadero

valor de negocio del producto sea evaluado en un ambiente real. Estas entregas no pueden pasar las 2 o 3 semanas como máximo.

3. Entendimiento compartido
    1) Diseño simple: se basa en la filosofía de que el mayor valor de negocio es entregado por el programa más sencillo que cumpla los requerimientos. Simple Design se enfoca en proporcionar un sistema que cubra las necesidades inmediatas del cliente, ni más ni menos. Este proceso permite eliminar redundancias y rejuvenecer los diseños obsoletos de forma sencilla.
    2) Metáfora: desarrollada por los programadores al inicio del proyecto, define una historia de cómo funciona el sistema completo. XP estimula historias, que son breves descripciones de un trabajo de un sistema en lugar de los tradicionales diagramas y modelos UML (Unified Modeling Language).
    3) Propiedad colectiva del código: un código con propiedad compartida. Nadie es el propietario de nada, todos son el propietario de todo. Este método difiere en mucho a los métodos tradicionales en los que un simple programador posee un conjunto de código. Los defensores de XP argumentan que mientras haya más gente trabajando en una pieza, menos errores aparecerán.
    4) Estándar de codificación: define la propiedad del código compartido así como las reglas para escribir y documentar el código y la comunicación entre diferentes piezas de código desarrolladas por diferentes equipos. Los programadores las han de seguir de tal manera que el código en el sistema se vea como si hubiera estado escrito por una sola persona.
4. Bienestar del programador
   La semana de 40 horas: la programación extrema sostiene que los programadores cansados escriben código de menor cualidad. Minimizar las horas extras y mantener los programadores frescos, generará código de mayor calidad.

Valores

Los Valores originales de la programación extrema son: simplicidad, comunicación, retroalimentación (feedback) y coraje. Un quinto valor, respeto, fue añadido en la segunda edición de Extreme Programming Explained. Los cinco valores se detallan a continuación:

1. Simplicidad: es la base de la programación extrema. Se simplifica el diseño para agilizar el desarrollo y facilitar el mantenimiento.
2. Comunicación: La comunicación se realiza de diferentes formas. Para los programadores el código comunica mejor cuanto más simple sea. Si el código es complejo hay que esforzarse para hacerlo inteligible. Las pruebas unitarias son otra forma de comunicación ya que describen el diseño de las clases y los métodos al mostrar ejemplos concretos de cómo utilizar su funcionalidad.
3. Retroalimentación (feedback): Al estar el cliente integrado en el proyecto, su opinión sobre el estado del proyecto se conoce en tiempo real. Al realizarse ciclos muy cortos tras los cuales se muestran resultados, se minimiza el tener que rehacer partes que no cumplen con los requisitos y ayuda a los programadores a centrarse en lo que es más importante.
4. Coraje o valentía: Hay que ser valiente para confiar en que la programación por parejas beneficia la calidad del código sin repercutir negativamente en la productividad. La simplicidad es uno de los principios más difíciles de adoptar. Se requiere coraje para implementar las características que el cliente quiere ahora sin

caer en la tentación de optar por un enfoque más flexible que permita futuras modificaciones.

5. Respeto: Los miembros del equipo se respetan los unos a otros, porque los programadores no pueden realizar cambios que hacen que las pruebas existentes fallen o que demore el trabajo de sus compañeros.

Características fundamentales

Las características fundamentales del método son:

- Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- Pruebas unitarias y continuas: las herramientas de prueba JUnit orientada a Java, DUnit orientada a Delphi y NUnit para la plataforma.NET. Estas dos últimas inspiradas en JUnit.
- Programación en 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 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. 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: es la mejor manera de que las cosas funcionen. Cuando todo funcione se podrá añadir funcionalidad si es necesario. 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.

La simplicidad y la comunicación son extraordinariamente complementarias. Con más comunicación resulta más fácil identificar qué se debe y qué no se debe hacer.

Roles

1. Programador: Escribe las pruebas unitarias y produce el código del sistema. Es la esencia del equipo.

2. Cliente: Escribe las historias de usuario y las pruebas funcionales para validar su implementación. Asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar el mayor valor de negocio.

3. Tester: Interpreta el pedido del cliente y ayuda al equipo de desarrollo a escribir las pruebas funcionales. Ejecuta pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.

4. Tracker: Es el encargado de seguimiento. Proporciona realimentación al equipo. Debe verificar el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, comunicando los resultados para mejorar futuras estimaciones.

5. Entrenador (coach): Responsable del proceso global. Guía a los miembros del equipo para seguir el proceso correctamente.

6. Consultor: Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto. Ayuda al equipo a resolver un problema específico. Además este tiene que investigar según los requerimientos.

7. Gestor (Manager): Es el dueño de la tienda y el vínculo entre clientes y programadores. Su labor esencial es la coordinación.

   Los dos equipos presentes en el proceso de desarrollo son el equipo de gestión y el equipo de desarrollo.

Preguntas
Escoger la mejor respuesta
Nivel Principiante

1. ¿Qué es la Programación Extrema?
    a) Es una metodología de desarrollo ágil que tiene como principal objetivo aumentar la productividad a la hora de desarrollar un proyecto software.
    b) Es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto.
    c) Modelo para la mejora o evaluación de los procesos de desarrollo y mantenimiento de sistemas y productos de software.
2. ¿Quién Formulo la Programación Extrema?
    a) Steve Jobs.
    b) Kent Beck.
    c) Dennis Ritchie.
3. Objetivo principal de la programación extrema:
    a) Asegurar la producción de software de alta calidad que cumpla con las necesidades de los usuarios, con una planeación y presupuesto predecible.
    b) Minimizar los riesgos durante la realización de un proyecto, pero de manera colaborativa.
    c) Dar al cliente lo que quiere y cuando quiere. Por tanto, se debe responder rápidamente a las necesidades del cliente.

4. ¿En que se basa la programación extrema?
    a) La programación extrema se basa en la simplicidad, la comunicación y el reciclado continuo de código.
    b) La programación extrema se basa en realizar un seguimiento diario de los avances del proyecto, logrando que los integrantes estén unidos, comunicados y que el cliente vaya viendo los avances.
    c) Ninguna de las anteriores.

5. ¿Quiénes participan del modelo de Programación Extrema?
    a) Programador, Cliente, Tester, Tracker, Coach, Consultor, Gestor
    b) Clientes, Propuesta de valor, Canales de distribución, Fuentes de ingresos, Recursos claves.
    c) Dueño de proceso, gestor de proceso, Analista de negocio, ejecutivo de negocio, Analista de negocio, ingeniero de procesos, Arquitecto SOA, ingeniero de desarrollo.

6. ¿Principales problemas que causan el fracaso de muchos proyectos software?
    a) Procesos altamente burocráticos, insuficiente coordinación y colaboración, muy poco tiempo.
    b) Retrasos, desviaciones, cancelación de un proyecto software y sistemas deteriorados, defectos.
    c) Entendimiento entre los participantes, coordinación entre el Front End y el Back End.

7. ¿En qué consiste la semana de 40 horas?
    a) Esperar que pasen 40 horas para comenzar a programar lo que el cliente pide en su producto.
    b) Cada 40 horas los líderes asignan el trabajo a cada miembro del equipo.
    c) Consiste en minimizar las horas extras de los programadores, de manera que se mantengan frescos, pues los programadores cansados escriben código de menor calidad.

8. ¿Con que nombre se le conoce a la programación en parejas?
    a) Pair-programming.
    b) Extra-programming.
    c) Three-programming.

9. ¿Cuáles son las 2 fases de la historia de usuario?
    a) Primera fase.
    b) Segunda fase.
    c) Todas las anteriores.

10. ¿La sección más importante en la programación extrema?
    a) Desarrollo.
    b) Diagramas.
    c) Ninguna de las anteriores.

Nivel Intermedio

1. ¿En la Programación Extrema se distinguen 2 tipos de roles, cuáles son?
    a) Programadores, desarrolladores, jefe de planta.
    b) Equipos y clientes.
    c) Ninguna de las anteriores.

2. ¿Cuáles son los dos tipos de equipo en el proceso de desarrollo?
    a) Equipo a, equipo b.
    b) Equipo primario, equipo segundario.
    c) Equipo de gestión y equipo de desarrollo.
3. ¿Qué sucede en el proceso de retroalimentación con relación al cliente?
    a) Su opinión se conoce en tiempo real.
    b) Su opinión no puede ser tomada en cuenta en este momento.
    c) Su opinión será tomada cuando el equipo de desarrollo encuentre el tiempo o esté libre de trabajo.
4. ¿Diga 3 actividades realizadas dentro de la XP?
    a) Insertar, eliminar, actualizar.
    b) Construir, insertar, realizar pruebas.
    c) Codificar, hacer pruebas, escuchar.
5. A la hora de hacer las pruebas, deben realizarse en pequeñas secciones de código y con pruebas sencillas, diga si es:

a) Falso
b) Cierto

6. ¿En qué tipo de lenguaje se puede realizar el ambiente de prueba?
    a) C.
    b) C++.
    c) Java.
    d) Todas las anteriores.
7. Escoja un ejemplo de un entorno de desarrollo para java.
    a) JUnit.
    b) Dev C++.
    c) Ms.net.
8. ¿Qué se hace en el proceso de codificación?
    a) Se plasman bien las ideas en código.
    b) Se escucha a los clientes.
    c) Se realizan diseños sencillos.
9. ¿En este valor se facilita el diseño para agilizar el desarrollo y el mantenimiento?
    a) Comunicación.
    b) Simplicidad.
    c) Respeto.
10. ¿Quién es el encargado de las pruebas?
    a) Tester.
    b) Consultor.
    c) Entrenador.

Nivel Experto

1. ¿Quién escribe las historias de usuario y las pruebas funcionales para validar su implementación?
    a) El cliente.
    b) La computadora.
    c) El código.
2. ¿Qué es el desarrollo iterativo e incremental?
    a) Reescribir ciertas partes del código para aumentar su legibilidad.
    b) Dividir la responsabilidad en el desarrollo.
    c) Pequeñas mejoras, unas tras otras.
3. ¿JUnit, DUnit y NUnit son plataformas para herramientas de?
    a) Pruebas unitarias.
    b) Pruebas continuas.
    c) Todas las anteriores.
4. ¿Permite a los equipos de programadores XP mejorar el diseño del sistema a través de todo el proceso de desarrollo?
    a) Refactorizar.
    b) Programar.
    c) Escribir.
5. ¿Cuántos principios tiene la XP y en cuántas categorías es agrupada?
    a) 5 y 4.
    b) 12 y 3.

c) 12 y 4.

6. ¿La simplicidad y la comunicación son extraordinariamente complementarias?
    a) Cierto.
    b) Falso.

7. ¿Cuántas fases tiene el ciclo de vida de un proyecto XP?
    a) 3.
    b) 6.
    c) 20.
8. ¿Cómo se le conoce a la fase de planeamiento o planificación de la entrega?
    a) Release.
    b) Salida.
    c) Entrada.
9. ¿Etapa en la que se le da soporte al cliente?
    a) Programación.
    b) Producción.
    c) Mantenimiento.

10. ¿Quién propuso los roles en la XP?
    a) Mark Zuckerberg.
    b) Steve Jobs.
    c) Ninguna de las anteriores.

Nivel Principiante

Cierto

1. ¿Es la Programación Extrema conocida por su nombre en inglés Extreme Programming (PX)?
   R: Cierto.
2. ¿Es conocida por ser una metodología Ágil?
   R: Cierto.
3. ¿Fue la programación extrema formulada por Kent Beck?
   R: Cierto.
4. ¿La simplicidad y la comunicación son extraordinariamente complementarias?
   R: Cierto.

Falso

1. ¿Es la metodología más difícil de llevar a cabo?
   R: Falso, fácil.
2. ¿No es útil para proyectos largos y extensos?
   R: Falso, actualmente su función principal es desarrollada en proyectos largos.
3. ¿El respeto no es un valor?
   R: Falso, es.
4. ¿Su programación no es organizada?
   R: Falso, es.

Nivel Intermedio

Cierto

1. ¿Los valores originales de la programación extrema son cuatro?

R: Cierto.

2. ¿El programador escribe las pruebas unitarias y produce el código del sistema?

R: Cierto.

3. ¿La simplicidad es la base de la programación extrema?

R: Cierto.

4. ¿Tiene un Desarrollo iterativo e incremental?

R: Cierto.

Falso

1. ¿Diseñar es plasmar las ideas en código?
   R: falso, codificar.
2. ¿El entrenador es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto?
   R: Falso, el consultor.
3. ¿Los ambientes de prueba solo se pueden realizar en lenguaje c++?
   R: Falso, se pueden realizar en cualquier lenguaje de programación
4. ¿El feedback es simplemente recibir quejas del cliente?
   R: Falso, es recibir a menudo correcciones o modificaciones por parte de los clientes.

Nivel Experto

Cierto

1. ¿La metáfora es desarrollada por los programadores al comienzo del proyecto, y define una historia de cómo funciona el sistema completo?
   R: Cierto.
2. ¿La historia de usuario consta de una lista de características necesarias para el cliente y que deben estar presentes en el producto final?
   R: Cierto.
3. ¿La sección más importante en la programación extrema es el desarrollo?
   R: Cierto.
4. ¿Una desventaja seria que requiere de un rígido ajuste a los principios de XP?
   R: Cierto.

Falso

1. El consultor es el dueño de la tienda y el vínculo entre clientes y programadores. Su labor esencial es la coordinación.
   R: Falso, el gestor o manager.
2. ¿Los programadores podrán realizar cambios que hacen que las pruebas existentes fallen o que demoren el trabajo de sus compañeros?
   R: Falso, los programadores deben respetar el trabajo de sus compañeros.

3. El cliente escribe las pruebas unitarias y produce el código del sistema. Es la esencia del equipo.
   R: Falso, el programador.
4. ¿El primer proyecto de XP se llevó a cabo el 6 de marzo de 2015 por Steve Jobs?
   R: Falso, el 6 de marzo de 1996, por Kent Beck.

Clave

Escoger mejor respuesta

I. Nivel principiante
1. A
2. B
3. C
4. A
5. A
6. B
7. C
8. A
9. C
10. A

II. Nivel Intermedio
1. B
2. C
3. A
4. C
5. B
6. D
7. A
8. A
9. B
10. A

III. Nivel Experto
1. A
2. C
3. C
4. A
5. C
6. A
7. B
8. A
9. C
10. C