Diseño Ágil y Programación Extrema

Guillermo Choque Aspiazu
gchoque@inf.umsanet.edu.bo

Resumen
Actualmente la programación extrema es una de las materias de mayor discusión en la comunidad para el desarrollo del producto software. Pero ¿cuáles son las características centrales de la programación extrema? y ¿cómo se involucra la programación extrema en el nuevo mundo de las metodologías ágiles? Este artículo intenta establecer el apuntalamiento de las metodologías ágiles y explica las razones por las cuales uno debe involucrarse en esta temática. Posteriormente se observa como la programación extrema utiliza un conjunto de prácticas para construir un equipo eficiente para el desarrollo del software que produzca software de calidad de una manera predecible y repetida.
***Palabras clave**: Producto software, diseño, método ágil, programación extrema.*

1. INTRODUCCION

Los métodos ágiles actualmente constituyen la tendencia entre muchos desarrolladores e ingenieros del software. Pero, ¿Qué en un método ágil?, la respuesta indica que es uno de los métodos más populares y que uno de los métodos prescriptivos asociados al mismo es la programación extrema. La programación extrema[1] fue desarrollada para direccionar las necesidades de grupos de desarrollo pequeño que están en constante confrontación con requerimientos inciertos y cambiantes [Newkirk, 2003].

La programación extrema es una metodología de desarrollo ligera basada en una serie de valores y una docena de prácticas de, llamémoslas así, buenas maneras que propician un aumento en la productividad a momento de generar un producto software. Actualmente la programación extrema es una de las materias de mayor discusión en la comunidad para el desarrollo del software. Pero ¿cuáles son las características centrales de la programación extrema? y ¿cómo se involucra la programación extrema en el nuevo mundo de las metodologías ágiles? Este pequeño estudio intenta establecer el apuntalamiento de las metodologías ágiles y explica las razones por las cuales uno debe involucrarse en esta temática. Posteriormente se observa como la programación extrema utiliza un conjunto de prácticas para construir un equipo eficiente para el desarrollo del software que produzca software de calidad de una manera predecible y repetida.

2. VALORES

El proceso de desarrollo de la programación extrema está fundamentado en una serie de valores y principios que lo guían. Los valores representan aquellos aspectos que los autores de XP han considerado como fundamentales para garantizar el éxito de un proyecto de desarrollo de software. Cuando se evalúan ciertas actividades en el proceso de desarrollo del software se tiende a contar con una referencia para entender si se están logrando los progresos respectivos en dicho proceso. La programación extrema tiene cuatro valores centrales que son utilizados para guiar las prácticas a ser empleadas [Beck, 2000]. Esos valores son:

1. **Comunicación**. Muchas fallas de los proyectos pueden ser trazadas a los problemas con comunicación. La comunicación evita casi todos los otros valores. Esto incluye comunicación entre todos los miembros del equipo, clientes, programadores y administradores.
2. **Simplicidad.** ¿Cuál es la cosa más simple que posiblemente podría trabajar? El mensaje es bastante claro. Se debe dar los requerimientos diarios para el diseño y escritura del software. No se debe tratar de anticipar el futuro, deje que el futuro se revele. Esto frecuentemente se realizara de maneras impredecibles haciendo que la anticipación constituya un costo que es demasiado alto para ser proporcionado.

[1] De la palabra inglesa: Extreme Programming, en el resto del artículo se referencia este término a través de la abreviación XP.

3. **Retroalimentación**. Las prácticas de la programación extrema están diseñadas para elicitar la retroalimentación de manera anticipada y frecuente. Las practicas tales como: entregas cortas, integración continua y pruebas, proporcionan una retroalimentación bastante clara. Esta retroalimentación permite al proyecto moverse hacia adelante en un fundamento bastante sólido.
4. **Coraje**. Este es uno de los valores más importantes. Frecuentemente muchas metodologías están basadas en ser netamente defensivas. La programación extrema cambia esta noción recomendando a los participantes que deben jugar para ganar. Esto equivale al coraje para decir "se hizo bastante diseño por ahora y dejare que ocurra el futuro".

Muchos partidarios de la programación extrema mencionan que los valores descritos son los necesarios para conseguir diseño y código simple, métodos eficientes para el desarrollo del software y clientes contentos. Estos valores deben ser intrínsecos al equipo de desarrollo.

De estos cuatro valores, quizá el que llame más la atención sea el referido al coraje. Detrás de este valor se encuentra el lema "*si funciona, mejóralo*", que definitivamente está en contraposición con la práctica habitual de no tocar algo que funciona. Aunque también es cierto que se cuenta con las pruebas unitarias, de modo que no se pide a los desarrolladores que asuman posturas heroicas al desarrollar diseño y código, sino sólo coraje.

Algunos autores además, añaden un quinto valor: la humildad. Y es que con la denominada solidaridad[2] en el diseño y la construcción de código, la retroalimentación y el trabajo estrecho en equipo, una buena dosis de humildad siempre contribuirá de manera positiva en la calidad del producto final.

3. PRACTICAS

La programación extrema es implantada con 12 prácticas. Estas prácticas son denominadas el "circulo de la vida" [Jeffries, Hendrickson y Anderson, 2001]. Dichas prácticas son la sangre que proporciona una especie de vida a la programación extrema. Cada práctica por si misma puede ser descrita en términos de su adherencia a los cuatro valores de la programación extrema descritos en el punto 2. Estas practicas trabajan de manera conjunta para formar un todo que es mucho mayor que la suma de las partes[3].

1. **Planificación**. Determina el alcance de la siguiente iteración mediante el trabajo con los clientes que proporcionan las prioridades de los negocios y con los programadores que proporcionan estimaciones técnicas.
2. **Entregas pequeñas**. Permite que el sistema entre en producción rápidamente. Este es un factor clave para tomar la retroalimentación sobre el software actual.
3. **Metáfora**. Entender como trabaja el sistema de manera completa. Es tan importante para el cliente comprender la metáfora con los programadores.
4. **Diseño simple**. Uno de los valores clave es la simplicidad. El sistema debe ser diseñado para las características que sean implementadas en el día. Debe dejarse que el futuro establezca como el sistema evoluciona a dicho punto, no se debe tratar de anticiparse al futuro. Probablemente, como no se hace una planificación detallada, las previsiones pueden estar equivocadas.
5. **Pruebas**. La retroalimentación es también uno de los valores clave. Las pruebas incluyen pruebas unitarias, con las cuales los programadores responden y aceptan los cuestionarios escritos por los clientes. Las pruebas son los indicadores de la completitud.
6. **Refactorización**. Los programadores son los responsables de mejorar el diseño del software existente sin alterar el comportamiento del mismo. La refactorización es parte de las actividades diarias de los programadores [Fowler, 1997]
7. **Programación por pares**. Trabajar con un compañero es un requerimiento cuando se empieza a escribir el código de producción.
8. **Propiedad colectiva**. Cualquier integrante del equipo puede modificar cualquier parte del sistema.

[2] Entiéndase como el acto de compartir el diseño y el código del problema.

[3] El holismo establecido por Aristóteles sostiene que: "*el todo es mas que la suma de las partes*".

9. **Integración continua**. Los programadores integran y construyen el producto software muchas veces en un día.
10. **40 horas semanales**. Una mejor denominación podría ser trabajar hasta estar cansado. Sin embargo, debe tenerse cuidado de las bajas en la producción debido al trabajo de excesivas horas, días y semanas.
11. **Cliente en casa**. El cliente se encuentra en el equipo, disponible para responder las consultas a tiempo completo. El cliente es también el encargado de escribir las pruebas de aceptación del producto software.
12. **Estándares de codificación**. La comunicación es el valor clave. Adoptar los estándares de codificación mejora la comunicación debido a que el código es consistente de clase a clase.

No es fácil aplicar una nueva metodología en un equipo de desarrollo ya que obliga al aprendizaje de una nueva forma de trabajar. También obliga al abandono de las cosas tradicionales, a cómo se hacían las cosas antes. XP ha sido adoptada por un gran número de equipos en los últimos años y de sus experiencias se ha extraído una conclusión sencilla: es mejor empezar a aplicar XP gradualmente.

El proceso que recomienda el autor de XP [Beck, 2000] es el siguiente:

1. Identificar el problema principal del proceso de desarrollo actual.
2. Escoger la mejor práctica que ayuda a resolver ese problema y aplicarla.
3. Cuando ese haya dejado de ser un problema, escoger el siguiente.

En realidad se recomienda que se apliquen las prácticas de dos en dos. El objetivo es que las prácticas de XP se apoyan unas a otras y por tanto dos prácticas aportan más que la suma de ambas y por tanto es más fácil comprobar los resultados. El objetivo final debe ser aplicar todas las prácticas, ya que representan un conjunto completo, "*si no se aplica todas las prácticas no se está haciendo programación extrema*" [Newkirk, 2003].

4. CONCLUSIONES

Según Kent Beck (2000), "*la programación extrema es una forma ligera, eficiente, flexible, predecible, científica y divertida de generar software*". Esta metodología ha surgido desde la experiencia, como una forma de resolver los problemas encontrados en los procesos de desarrollo del producto software en los que se han visto involucrados sus autores. Este tipo de desarrollos eran en general de creación de software a la medida del cliente y hay numerosas opiniones que relatan el éxito de esta metodología en este ámbito. Este artículo intentó establecer las metodologías ágiles y explicar al mismo tiempo las razones por las cuales el lector debe involucrarse en esta temática. Se observó como la programación extrema utiliza un conjunto de prácticas para construir un equipo eficiente para el desarrollo del software que produzca software de calidad de una manera predecible y repetida.

5. BIBLIOGRAFIA

- Beck, K. (2000) *Extreme Programming Explained*, Addison- Wesley, Reading MA.
- Jeffries, R, Hendrickson, C. and Anderson, A. (2001) *Extreme Programming Installed*, Addison-Wesley.
- Cockburn, A. (2002) *Software Development*, Addison- Wesley, 2002
- Fowler, M. *Refactoring: Improving the Design of Existing Code*, Addison-Wesley, Reading MA, 1997
- Newkirk J. *Introduction to Agile Processes and Extreme Programming*. ThoughtWorks Chicago, IL, USA. 2003. http://www.thoughtworks.com

Profesor de la Universidad Mayor de San Andrés.
La Paz – Bolivia.

Artículo de divulgación.
Elaborado para la materia: Ingeniería del Software
Agosto 2005.