Relación de problemas 5

Santiago Muñoz Castro

1. **El problema básico del desarrollo del software es el riesgo, algunos ejemplos de situaciones de riesgo son:**

   …… ¿Cómo trata XP el riesgo que hemos comentado anteriormente?

   Los retrasos en la planificación, XP lo soluciona realizando una planificación ágil en la cual se plantea como un diálogo continuo entre todos los involucrados del proyecto. Se planifica tanto el inicio como todos los días de desarrollo del proyecto. Esta planificación sirve como guía, pudiendo actualizarla según se vaya obteniendo información del producto emergente.

   Los proyectos cancelados, XP lo evita haciendo como ya hemos dicho una planificación ágil, la cual va a evitar en la medida de lo posible retrasos en el producto, a esto le aportamos que en cada, además en cada iteración podemos obtener un producto entregable y funcional (weekly cycle y quarterly cycle), gracias a las constantes pruebas de integración, por lo que, si se retrasa la entrega, se puede obtener un producto que aporte la mayor funcionalidad posible.

   Para evitar que el sistema se deteriore, XP aplica slack reduciendo los compromisos imposibles, que después hagan que surjan estos problemas, aunque lo determinante es el diseño incremental, donde diseñamos el producto de la forma más simple y con coste de cambio bajo, gracias al refactoring (mejorar estructura interna del código sin realizar alterar el comportamiento externo).

   Para evitar tasa de defectos, XP realiza pruebas durante todo el proyecto, escritas por los propios programadores, se integran de forma continua y automática, generando una plataforma estable y detectando en cualquier momento fallos o defectos en el sistema. Defectos estéticos o de añadir funcionalidad se resuelven con diálogos constantes con todos los participantes del proyecto.

   Para evitar requisitos mal comprendidos, XP directamente renuncia a ellos, sustituyéndolos por historias de usuarios que contienen trozos de funcionalidad que aportan valor al negocio, estan establecidas por el cliente a partir de conversaciones. De esta forma, también permitimos una mayor adaptabilidad.

   Para evitar problemas en el cambio de negocio, como ya he mencionado antes, XP utiliza historias de usuarios y una planificación ágil por iteraciones, lo que permite una mayor adaptabilidad ante los cambios durante el desarrollo del proyecto.

   Ante la falsa riqueza de características, XP se centra en la simplicidad acorde a lo que debe hacer el producto, es decir, hacer que funcione de la forma más sencilla, obteniendo el producto a entregar en el menor periodo de tiempo, y si hace falta añadir más funcionalidad que sea porque el cliente la ha solicitado.

Por último, para que no surjan cambios en el personal, XP fomenta y utiliza la programación en parejas que permite desarrollar un mejor código en base a unos estándares, cada programador tiene asignado las tareas que mejor van a desarrollar, evitando así la frustración de estos, además programar en parejas suele dar más satisfacción, ya que no se trabaja de forma individual, las parejas pueden cambiar para no abusar de la monotonía.

2. Comenta brevemente las diferencias principales entre XP y Scrum.

En Scrum la programación se realiza de forma individual, mientras que en XP se realiza en parejas. XP está orientado en mayor medida que Scrum a las pruebas de integración y aceptación.

La prioridad en las HU está marcada en XP por el cliente, mientras que en Scrum esta establecida entre el Product Owner y los stakeholders. XP esta pensada para equipos pequeños de 10 trabajadores y en Scrum es lo mismo pero se aconseja un máximo de 8.

En Scrum una iteración puede llegar a durar hasta un mes, mientras que en XP la máxima duración es de 3 semanas.

3. Comenta los roles de XP con los roles de Scrum.

Los roles que se comparten entre las dos metodologías son los programadores y lo clientes. En XP en vez de haber un Scrum Master hay un entrenador que es líder del grupo que orienta al principio del desarrollo al equipo y es responsable del proceso, este va perdiendo protagonismo mientras más avanza el proyecto.

Los roles de XP que no posee Scrum son:

- El consultor, que cumple el papel de ayudar en conocimiento al equipo de desarrollo, funcionalidad que tiene en Scrum el Scrum Master.
- El rastreador (Tracker), seguimiento de proceso en cada iteración, realiza las estimaciones y observa el proyecto sin interferir. De este papel se encarga el Product Owner el Scrum.
- El gestor (big boss), es el coordinador entre clientes y programadores, en este caso Scrum no tiene un gestor que coordinase a ambos miembros, pero si se podían comunicar en reuniones con todo el equipo de desarrollo.
- Por último el tester, que se encarga de realizar las pruebas funcionales, es un rol que posee únicamente XP, mientras que Scrum el que realizaba las pruebas si tocaban era un programador del equipo de desarrollo u otro cualquier integrante.

4. Relacionar los 14 principios de XP2 con las prácticas propuestas en XP.

a. Sentarse Juntos. Responsabilidad
b. Sensación de equipo. Humanidad
c. Espacio de trabajo informativo. Flujo, Reflexión
d. Trabajo a pleno rendimiento. Responsabilidad.
e. Programación en parejas. Calidad, Diversidad
f. Historias de usuario. Beneficio Mutuo, Humanidad.

g. Ciclo semanal. Pasos de bebé
h. Ciclo trimestral. Mejora
i. Aflojar. Beneficio Mutuo, Redundancia, Economía
j. Ten minute build. Flujo
k. Integración continua. Fallo, Mejora
l. Pruebas antes de implementar. Reflexión, Calidad
m. Diseño incremental. Calidad
n. Metáfora. Diversidad
o. Propiedad colectiva. Beneficio mutuo, mejora, calidad
p. Cliente in situ. Reflexión, flujo, humanidad, mejora
q. Estándares de programación. Auto-similitud.

5. Clasifica las prácticas de XP que hemos comentado en el tema 4 en las siguientes categorías.

Planificación y análisis de requisitos:

- Historias de usuario.
- Ciclo trimestral.

Factores humanos y equipo.

- Sentarse juntos.
- Sensación de equipo.
- Cliente in situ.
- Trabajo a pleno rendimiento.
- Metáfora.

Diseño

- Programación en parejas.
- Aflojar.
- Estándares de programación.
- Propiedad colectiva.
- Diseño incremental
- Pruebas antes de implementar.

Codificación del software y entrega.

- Ciclo semanal
- Espacio de trabajo informativo
- Ten minute build.
- Integración continua.

6. ¿Para qué se utiliza WIP(Work in progress) en Kanban?

Para exponer cuellos de botella, colas, variabilidad y desperdicios. Es decir, si hay algo que bloquea el desarrollo, el proceso se para en seco, para que todo el equipo se centre en solucionar el problema.

7. Comenta brevemente las diferencias entre Kanban y Scrum.

La principal diferencia es que Kanban no es una metodología si una metodología funcional a utilizar en una metodología. En Scrum el cuello de botella esta limitado por la carga de trabajo a realizar según los días de la iteración, mientras que en Kanban cada columna puede tener limitado su propio cuello de botella.