Escribir Casos de Prueba

English

Entradas

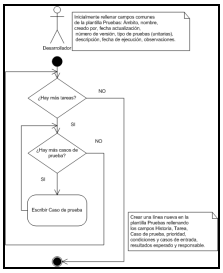
- Tareas Historias versión X Programador Y

Salidas

- Pruebas.xls

Solución

Proceso



Tiempo de Desarrollo

- Para adquirir el conocimiento necesario para desarrollar el producto software:
- Para crear el Patrón de Producto: 45 minutos.
- Para aplicar el Patrón de Producto:

Video Explicación

- No aplica

Patrones Relacionados

- Patrón Establecer Balanceo
- Patrón Pruebas Unitarias

Controladores de Calidad

- Ninguno

Plantillas

Pruebas.xls

Ejemplos

Caso de prueba

Herramientas de Soporte

- Será necesario un editor de texto como OpenOffice Writer o Microsoft Word.
- Será necesario un editor de hojas de cálculo como OpenOffice Calc o Microsoft Excel
- Además de una herramienta como Visual Paradigm for UMLpara la realización de los diagramas expuestos.

Contexto Inicial

Este producto puede usarse en cualquier proyecto en el se deban identificar casos de prueba para una funcionalidad concreta antes de su implementación. Las características del software que no pueden ser demostradas mediante pruebas, simplemente, no existen. Las pruebas dan la oportunidad de saber si lo implementado es lo que en realidad se tenía en mente. Las pruebas nos indican que nuestro trabajo funciona, cuando no podemos pensar en ninguna prueba que pudiese originar un fallo en nuestro sistema, entonces habremos acabado por completo.

Contexto Resultante

Los desarrolladores obtienen un conjunto de casos de prueba relacionados con una funcionalidad en concreto a desarrollar.

Problema

Los desarrolladores deben escribir pruebas que ayuden a obtener un programa que funcione y que se mantenga funcionando a lo largo del tiempo. Los programadores escriben pruebas método a método bajo las siguientes circunstancias:

1. Si la interfaz para un método no esta clara, escribe una prueba antes de escribir el método.
2. Si la interfaz esta clara, pero imagina que la implementación podría ser un poco menos complicada, escribe una prueba antes de escribir el método.
3. Si piensa en una circunstancia no usual en la cual el código debiera funcionar conforme esta escrito, escribe una prueba para comunicar la circunstancia.

Restricciones (*Forces*)

- **Características de las organizaciones:** Este patrón puede utilizarse en los proyectos existentes en cualquier tipo de compañía.
- **Tipo de sistema a desarrollar:** Este producto puede utilizarse en proyectos en los que los requerimientos de usuario sean cambiantes.
- **Tipo de Cliente:** Debe existir, o debe conseguirse, que el área de negocio destinataria del desarrollo se implique en la consecución del mismo.
- **Heurísticas de uso:** :Si se necesita disponer urgentemente del aplicativo o de algunas de sus funcionalidades.

Roles

- Desarrolladores (2 a 12)

Lecciones Aprendidas

- No se debe caer en la tentación de subestimar las pruebas. Trabajar en parejas de programadores puede reducir la posibilidad anterior. Las pruebas deben ser aisladas y automatizadas. El resultado de las pruebas debe ser positivo o negativo sin puntos intermedios. Como es imposible probar todo hay que escribir los casos de pruebas de lo que se piensa que puede fallar. Hay que tener especial cuidado con las pruebas que funcionan cuando no lo esperábamos (debemos revisar ese caso de prueba). Por tanto para escribir los casos de prueba debemos reflexionar sobre las pruebas que merece la pena hacer y cuales no.
- El archivo de salida Pruebas.xls es un documento genérico, para todos los tipos de prueba con la casilla pruebas unitarias marcada, que aglutina todos los casos de prueba para una tarea en cuestión.
- Un desarrollador elige una tarea asignada y junto con un compañero analizan la funcionalidad.

Nivel de Madurez

- Este Patrón de Producto no se relaciona con ningún nivel de madurez(N/A).

Conocimientos y Habilidades Básicos

Conocimientos

- Conocimiento del estándar de codificación que 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.
- Conocimiento de la visión común de cómo funciona el programa en el que se desarrollan las actividades.

Habilidades

- Capacidad de trabajo en grupo.Todos en un equipo XP contribuyen de la manera que pueden.
- Predicción de qué se habrá terminado para la fecha de entrega, y determinación de qué hacer después.
- Capacidad de programación de a pares.Además de generar mejor código y pruebas, sirve para comunicar el conocimiento a través de los equipos.

Recursos de Información

- Álvarez, José R. y Arias Manuel. Método Extreme programming.Recuperado el 2010-03-05 de http://www.ia.uned.es/ia/asignaturas/adms/GuiaDidADMS/node61.html
- Anaya Villegas, Adrian. A proposito de programación extrema XP(extreme Programming).Recuperado el 2010-02-10 de http://www.monografias.com
- Beck, K.(2000), Una explicación de la programación extrema. Aceptar el cambio. Ed. Addisson Wesley.
- De Seta, Leonardo. Una introducción a Extreme Programming.Recuperado el 2010-03-02 de http://www.dosideas.com/noticias/metodologias/822-una-introduccion-a-extreme-programming.html
- Extreme Programming: A gentle introduction. Recuperado el 2010-03-15 de http://www.extremeprogramming.org/

- Joskowicz, José. Reglas y prácticas en Xtreme Programming. Recuperado el 2010-03-15 de http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf
- Letelier, Patricio y Panadés Mª Carmen. Metodologías Ágiles en el desarrollo de software: extreme programming. Recuperado el 2010-03-15 de http://www.willydev.net/descargas/masyxp.pdf
- Newkirk, James y Martin, Robert C.(2001), La programación Extrema en la Práctica.Ed Addisson Wesley.