INGENIERÍA DEL SOFTWARE II

Testing

Grupo Recuperación Prácticas ISOFT2

24/05/2014

Eimard Sobrino Zurera

Raul García-Hidalgo

David Martín García

María Álvarez Rodríguez

Contenido

Introducción

Desde la concepción hasta su puesta en marcha, el producto software pasa por distintas versiones, que cubren de manera incremental el desarrollo del producto. Por ello, surge la necesidad de comprobar que, a medida que van completándose los distintos casos de uso que lo forman, estos funcionan de forma correcta y de acuerdo a las especificaciones, ya sea el primer elemento incorporado, hasta la última modificación.

Por ello existe lo que se denomina *"Testing",* cuya principal función es comprobar todas y cada una de las características que conforman el producto software, tanto las ultimas realizadas como las ultimas incorporadas, de forma que una modificación nueva sea detectable si interrumpe el procesamiento de otra función anteriormente implementada.

Las principales funcionalidades del plan de pruebas para los productos software se detallan a continuación:

- **Integridad.** Es necesario comprobar la integridad del producto así como las diferentes propiedades y funciones que han sido detalladas, tales como comportamientos específicos, reacción ante determinadas entradas o salidas, además de resultados coherentes a los requisitos del problema para cada función detallada.

- **Trazabilidad.** Se deberá controlar todos los cambios realizados al producto durante su ciclo de vida, asegurando que el software sea consistente, y que su funcionalidad perdura a lo largo de todo el proceso de desarrollo.

- **Integración.** Además de desarrollar cada funcionalidad, es necesario comprobar que se integran de forma correcta y coherente, así como que cumplen los resultados esperados.

- **Auditoría y revisión.** Se comprobará que el producto este completo y que cumple con los requisitos especificados en la planificación. Deberá mantener consistencia entre sus componentes.

Propósitos del plan de pruebas

Los propósitos por los que este plan se llevará a cabo son los siguientes:

- Comprobar la integridad durante toda la evolución del proyecto y controlar las partes sometidas a pruebas.

- Documentar las pruebas que se realizan durante todo el desarrollo, así como su resultado, ya que dependiendo de este puede ser necesario modificar para solventar los problemas encontrados.

- Controlar los cambios que surjan tras las pruebas.

- Informar y registrar si los cambios tras la prueba han sido fructíferos o será necesario volver a realizar más cambios.

Alcance del plan de pruebas

El alcance del plan es definir la función de las pruebas del software a lo largo de todas las fases del ciclo de vida del software. Para ello:

- Tras todas las implementaciones de funciones será estrictamente necesario comprobar y documentar los resultados de las pruebas sufridas por el software.
- Las pruebas están diseñadas y estrictamente ligadas a los procesos de gestión de configuración, ya que el propio sistema *maven* integra los diferentes mecanismos de prueba.

El alcance del plan tiene cabida dentro del proyecto actual de nuestra empresa, aunque podría aplicarse a proyectos futuros.

Relación con la organización y con otros proyectos

El proyecto en marcha es el primero realizado en nuestra empresa, asi que no hay relación con ningún proyecto anterior o paralelo. A pesar de eso, la organización y métodos utilizados podrían ser usados en futuros proyectos.

Términos claves

Test-Case: Un *caso de prueba* (*test case)* es un conjunto de valores de entrada, precondiciones de ejecución, postcondiciones de ejecución y resultados esperados, desarrollados con un objetivo en particular, que es comprobar el correcto funcionamiento, de acuerdo a lo indicado en la prueba, comprobando de esta manera el correcto comportamiento.

Error: Equivocación realizada por una persona (en este caso el programador).

Defecto: Es un error al realizar alguna actividad concerniente al software.

Fallo: Es un desvio del comportamiento requerido por el sistema de una parte de este.

Modulo: Ente que conforma una parte de las funcionalidades necesarias por el sistemas.

Test Suite: Conjunto de uno o mas casos de prueba con un proposito que usualmente se ejecutan en conjunto.

Referencias

Nuestro plan para la gestión de las distintas pruebas ha sido desarrollado siguiendo las directrices del estándar IEEE 12207:2008 y IEEE 1987.Este estándar define la metodología a seguir en el Plan de Pruebas: establece el propósito que buscamos, los pasos que deben ocurrir, las diferentes tareas y actividades involucradas durante el proceso, y los diferentes tipos de pruebas.

La información del proyecto, así como su código fuente, está situada en el siguiente repositorio: https://code.google.com/p/iso2-1314-esolutions/.

Objetivo del plan de pruebas

El objetivo principal de un plan de pruebas no es otro que, comprobar el correcto funcionamiento de acuerdo a un comportamiento definido para el elemento al que se va a someter a estas pruebas.

El plan de pruebas tiene como propósito final encontrar errores en el software, ya sea por un mal funcionamiento entre módulos, por errores en la codificación, o por errores en el modelado del comportamiento. Ademas se obtiene datos acerca de las ejecuciones, por lo que se pude evaluar la calidad del producto, asegurando la calidad final asi como su rendimiento.

En nuestro caso, no existen test de rendimiento, y solo evaluaremos la integración entre modulos y el correcto funcionamiento para cada modulo.

Limitaciones y suposiciones que afectan al plan de pruebas

Dado que es imposible comprobar de forma total el comportamiento del programa, en este plan de pruebas se cubrirá al menos el 40% del código del programa.

Esto se realiza debido a:

Calendario de desarrollo: El proyecto tiene fijada una fecha de entrega, y se tomará muy en cuenta para el desarrollo, dado que se debe entregar el producto final al cliente antes de que esa fecha caduque. Por eso existen algunos márgenes de tiempo fijados en un calendario para solventar posibles retrasos con algún componente del desarrollo.

Presupuesto: Nuestro desarrollo debe aprovechar el presupuesto estipulado para el proyecto, y en ningún caso sobrepasarlo a menos que esto sea acordado por el cliente.

Responsabilidades y autoridades del plan

Este plan se ha llevado a cabo por todos los integrantes de la empresa de forma conjunta, a continuación se detallarán los roles de cada uno:

- **Gestor de Pruebas:** (David Martin Garcia)
  Se ha encargado de la planificación, seguimiento y control de los elementos que forman parte del plan pruebas.
- **Coordinadora de la gestión del plan de pruebas:** (María Álvarez Rodríguez)
  Su función ha sido identificar y diseñar las distintas pruebas necesarias para este plan de pruebas.

- **Coordinador de pruebas:** (Eimard Sobrino Zurera)
  Realiza y evalua los resultados de las diferentes pruebas, asi como gestiona a cada responsable si existen errores y los indicios encontrados.

- **Gestor de la implementación de las pruebas:** (Raúl García – Hidalgo Tajuelo)
  Su trabajo ha sido implementar y coordinar con el gestor de configuración el plan de pruebas.

Responsabilidades del plan de pruebas

Todos los integrantes del proyecto deberán cumplir una serie de responsabilidades para que el plan de pruebas pueda ser cumplido de manera eficiente además de satisfactorio.

- Deben cumplirse todas las tareas dentro del tiempo establecido por el calendario.

- El proceso de pruebas será incremental, efectuando pruebas al final de cada iteración, y al terminar se realizarán pruebas globales del sistema con todos los módulos integrados.

- Se generará documentación de cada paso y de cada proceso que se lleve a cabo durante el desarrollo del sistema.

- Se llevara un seguimiento y control de los posibles cambios realizados a lo largo del desarrollo y diseño del software.

Metodología de pruebas

Existen dos tipos básicos de pruebas: pruebas de caja blanca y pruebas de caja negra que a su vez se subdividen en conjuntos más específicos. Las pruebas de caja blanca se aplican normalmente a pruebas unitarias y a ciertos tipos de pruebas de integración. Las pruebas de

caja negra tienen una referencia más general, y son utilizadas tanto en pruebas unitarias como pruebas funcionales, de integración, de sistema e incluso de aceptación.

Todas las pruebas que planteamos en este plan de pruebas son de caja negra, donde tendremos unos datos para la prueba y un objetivo, este objetivo desarrollara la prueba y nos devolverá los datos tras la ejecución o incluso en algunas ocasiones no devolverá nada, lo que nos indicara su correcto funcionamiento. Estos datos se comprobaran con los datos esperados, se juzgara si el resultado es correcto, y por tanto si ha superado la prueba o por el contrario no son los esperados y no se ha superado la prueba.

Los pasos dados para determinar el funcionamiento de cada prueba son:

1. Seleccionar el caso de uso a analizar
2. Determinar la cobertura del caso de uso, en la mayoría de los casos es casi total.
3. Se realizan y se lanzan sobre el objetivo con el fin de encontrar errores.
4. Si los casos de uso encuentran errores, esto eran comunicados al equipo de desarrollo y se solucionaran. Tras esto se volverá a realizar la prueba.
5. Si no se encuentran errores, se mide la cobertura final en el código que hemos alcanzado y se compara con el umbral establecido.
6. Si no se ha alcanzado el umbral será necesario establecer mas pruebas para ese test case.

Casos de prueba

A continuación se detallaran los diferentes casos de prueba por cada iteración. Además al final observaremos un diagrama que mostrara el porcentaje de código cubierto por la batería de pruebas.

Iteración 1:

Corresponde a la iteración 1 en el plan de configuración, y comprueba de forma inicial el correcto funcionamiento del mecanismo de pruebas, comparando dos cadenas iguales, para comprobar si las pruebas fallan antes incluso de ser ejecutadas sobre el código.

Iteración 2:

En primer lugar se comprobara el correcto funcionamiento de la clase AgenteBBDD, que realiza y gestiona toda la comunicación con la base de datos que utiliza el software.

Esta prueba, obtendrá desde la base de datos un valor introducido anteriormente a propósito, que será comprobado con el valor establecido que debería encontrar.

Además comprobara el correcto funcionamiento del método ValidarUsuario de la interfaz gráfica, a fin de obtener si es capaz de validar usuarios de forma correcta.

Se comprobaran dos casos:

- Uno correcto, con la tupla DNI;Password {05713540,9999]
- Otro incorrecto, con la tupla DNI;Password {05713540,8999]

Este usuario se debe encontrar previamente introducido en la base de datos.

Ambos dos pruebas corresponden las funcionalidades indicadas para esta iteración.

Iteración 3:

En esta iteración, se desarrollan dos casos de usos ligados al ente Producto.

Se realizaran pruebas que comprueben la corrección de la funcionalidad de eliminar productos. Se añadirán manualmente productos a la base de datos y se ejecutara la prueba, con el fin de comprobar si se han eliminado los productos esperados.

Además también existe la funcionalidad comprobar almacén, que nos debería dar todos los productos necesarios que tengamos en la base de datos, y dado que desconocemos la situación en la base de datos, el simple hecho de que se ejecute sin error, ya es muestra de su funcionamiento. Se preferiría otra situación más restrictiva, pero no ha sido posible alcanzar otra solución.

Iteración 4:

En esta iteración los casos las funcionalidades a tratar son añadir productos y comprobar mercancías.

El primer caso a tratar es el de añadir productos, dado que es mas necesario que el otro.

Se añadirá un producto a la base de datos y se comprobara manualmente tras la ejecución si existe, la prueba nos indicara en primera instancia si no ha ocurrido algún error de ejecución, pero nosotros tendremos que comprobar en la base de datos que se ha generado correctamente la información del producto.

Para comprobar mercancía, comprobaremos un artículo conocido y comprobaremos si concuerda el resultado con el esperado.

Iteración 5:

No se ha terminado totalmente la programación de esta iteración dada la ausencia del tiempo necesario, por lo que no se han realizado pruebas.

Conclusión:

Tras generar todas las pruebas y haber comprobado la corrección de todas ellas, se modificaron algunos pequeños fragmentos de código para que el resultado fuera correcto.

La principal ventaja fue la de poder ejecutar de forma ágil y rápida la misma prueba para comprobar el funcionamiento, evitando introducir errores humanos en la ejecución y aislando el resto del sistema. Además la información devuelta es fácil de interpretar y sirve para observar las divergencias entre lo obtenido y lo esperado.

Resultados obtenidos

Tras la ejecución de todos los casos de prueba, el porcentaje de código cubierto es superior al porcentaje requerido para cada parte, ya que en el porcentaje de cogido general, no se ha superado, dado el gran número de porcentaje de código dedicado a la interfaz gráfica.

Al no existir bifurcaciones en las funcionalidades planteadas, realizar pruebas ha sido sencillo, a la par que hemos obtenido un número reducido para poder cubrir todas las opciones.