ESTRATEXIA DE PROBAS

Título proxecto: practica-vvs
Ref. proxecto:
https://github.com/Xokage/practica-vvs-4malladores.git

Validación e Verificación de Software

Data de aprobación	Control de versións	Observacións
08/12/2015	1.0	Primeira revisión

1. Introdución

1.1. Propósito do documento

Identificar a situación actual, os problemas identificados no proxecto software, e dar unha ou máis posibles vías de actuación/solución, xunto cos criterios para seleccionalas e as medidas para avaliar a súa efectividade.

1.2. Alcance e difusión

Este documento estará dispoñible tanto para os desarrolladores do software como para calquera persoa interesada nel, e poderá acceder ao dito a través do repositorio.

1.3. Contexto

Unha vez implementadas as diferentes funcionalidades, e xa decididos as probar todas as posibles situacións, redactamos o presente documento para aclarar cal é a situación actual e como se solventarán os principais problemas atopados.

2. Situación actual

Trátanse de probas Unitarias (xa que probamos cada módulo por separado) Dinámicas (xa que executamos o código mediante JUnit) de Caixa Negra (Xa que non necesitamos saber cómo funciona o software pois únicamente usamos os métodos que nos proporcionaron), Positivas (posto que probamos o uso normal), Funcionais (xa que proban o correcto funcionamento dos módulos). Consideramos o modelo UML proporcionado pola profesora como punto de partida para desenrolar as probas.

Non realizamos Particións Equivalentes nin comprobamos os Valores Fronteira xa que nos parecía complicar innecesariamente as probas. Prescindimos de Mapas de Transicións xa que non usamos estados nin memoria de forma complexa. Non fixemos Árbores causa-efecto nin Táboas de Decisión porque consideramos este tipo de documentación prescindible no noso desenrolo áxil. Así mesmo evitamos a Selección de datos Aleatoria posto que non o consideramos ao deseñar as probas.

Á hora de realizar as probas, tivemos en conta todas as posibles coberturas, é dicir, a cobertura de ramas, de condicións e de instrucións. As probas unitarias realizadas son probas positivas, o obxectivo non é alcanzar a cobertura de todas as instruccións. En canto ao análise do fluxo de datos, realizamos un análise superficial das variables sen utilizar e das referenciadas sen inicializar.

En canto ás probas funcionais de caixa branca de mutación de código e de inserción de fallos, non se tiveron en conta.

A revisión do código fíxose por pares entre os compoñentes do grupo e non se fixo ningunha revisión por parte dun experto nin se empregou ninguna ferramenta automática. Tampouco solicitamos a inspección visual de terceiros.

Non se tiveron en conta a realización das probas non funcionais, tanto de caixa negra como de caixa branca. Tampouco se realizaron métricas de proba.

O software atopábase sen probar polo que a situación da que partimos é dun software sinxelo que aínda non foi probado. O obxectivo é realizar unha suite completa de probas, tanto estáticas como dinámicas, unitarias, funcionais e non funcionais...). É necesario telo realizado antes do 16 de Decembro de 2015.

3. Análise de problemas

Listaxe de problemas individualizados identificados no análise da situación actual. A lista non deberá superar a decena de problemas.

1. O código tiña moitas faltas de estilo, non cumpría correctamente o estándar.
2. As probas eran incompletas.

3.1. Problema - Faltas de estilo

O estándar de codificación non se seguía correctamente. Indentación irregular, nomes de variables confusos, estrutura desigual... Isto pódese deber a non deixar claro o estándar a seguir, así como faltas de coñecemento, por exemplo usar o atributo final para variables e métodos que non cambian. O impacto deste problema é mínimo debido a que o volume de código é moi pequeno, xa que noutro caso o impacto sería moito maior.

3.1.1. Solución

O responsable da realización dos cambios necesarios para solventar o problema 3.1 é Xoán Antelo Castro. Utilizar unha ferramenta como CheckStyle para comprobar se o código novo que se escribe cumpre coas convencións. Esto permite asegurarnos on the fly que o código que estamos escribindo cumpre o estándar e, en caso contrario, solucionalo no momento.

3.2. Problema - Probas incompletas

Ao realizar o deseño inicial das probas realizáronse únicamente sinxelas probas de unidade. Isto debeuse a que non tiñamos como obxectivo unha suite de probas completas.

3.2.1. Solución

Analizando as probas realizadas a través da ferramenta cobertura, obtemos que existen certas funcionalidades que non son comprobadas en ningún dos tests de unidade, como son por exemplo, as funcionalidades alta e baixa do servidor simple. Para solucionalo, incluímos ditas probas no test JUnit correspondente. Para elo, só necesitamos o soporte necesario para poder executar a ferramenta cobertura e varias persoas que sexan as encargadas de levar isto a cabo, que neste caso concreto foron Cristian Canosa Pérez e Francisco Pais Fondo. De non realizar as probas dos métodos que non se tiveron en conta, obteriamos un porcentaxe baixo de cobertura, co cal non sería moi recomendable.

4. Criterios de diagnóstico solución-problema

Se antes de aplicar unha solución escollida, o problema existía e despois de aplicala deixa de existir, a solución resolveu o problema.

5. Actuacións recomendadas

- (3.1 , 3.1.1)
- (3.2 , 3.2.1)

6. Política de revisión

Revisarase a estratexia de proba cando nolo indique o profesorado.

7. Outros aspectos de interese

Non hai outros aspectos de interese.