MONITORIZACIÓN DE PROBAS

Título proxecto: Práctica VVS
Ref. proxecto: andy135

Validación e Verificación de Software

Data de aprobación	Control de versións	Observacións
26/11/2015	1.0	Creación inicial do documento

Iago Santos Domínguez
José Andy Quintero Melo
Elias Ferreiros Borreiros

1. Contexto

O presente documento de monitorización de probas fai referencia a versión basica do proxecto presente en github.

2. Estado actual

- Funcionalidad 01: Manejo de contenidos: Se han definido una serie de contenidos, entre los que se encuentran:
  - Anuncio, contenido simple del que podremos obtener el título, que siempre será PUBLICIDAD, y la duración, siempre 5.
  - Canción, contenido simple del que podremos obtener el título y la duración.
  - Emisora es un caso de contenido del que además podremos obtener la lista de reproducción.

  Persona responsable de su desarrollo: Andy Quintero Melo.
  Persona responsable de las pruebas: Elías Ferreiro Borreiros.

- Funcionalidad 02: Manejo de servidores: Se han definido dos tipos de servidores:
  - Servidor Simple: En él se pueden almacenar una serie de contenidos asi como eliminarlos y buscarlos.
  - Servidor Respaldado: Realiza la misma función que un servidor simple salvo en el caso de buscar contenidos en el cual se diferencia al poseer otro servidor que le servirá de respaldo.

  Persona responsable de su desarrollo: Iago Santos Domínguez. Persona responsable de las pruebas: Elías Ferreiro Borreiros.

Funcionalidad	Pruebas objetivo	Pruebas preparadas	% ejecutada	% superada
01	25	25	100	100
02	19	19	100	100

3. Registro de pruebas

Tipos de pruebas realizadas hasta el momento:

- Happy testing: Durante la primera semana de testing usamos nuestro sentido común a la hora de hacer pruebas.
- Pruebas de caja negra: Al comenzar a adquirir conocimientos sobre la forma de hacer pruebas bien, usamos las siguientes técnicas:
  - Particiones equivalentes
  - Valores-frontera
- Pruebas de caja blanca: Finalmente para ver de una manera más visual el abarque de código de las pruebas empleamos:
  - Cobertura de instrucciones
  - Cobertura de ramas
- Se ha descartado el uso de pruebas automatizadas debido a la simplicidad de los valores frontera y del código en general.

4. Registro de errores

- Errores en la nomenclatura de las interfaces. #2 #3
- Los métodos buscar de Canción y Anuncio tenían una respuesta incorrecta. #5
- Error en la propagación de los tokens en los servidores respaldados. #6
- Al añadir los anuncios recuperando canciones de un servidor la publicidad se añadía siempre al principio en lugar de cada tres canciones. #7
- Al realizar las pruebas de valores frontera en los métodos de los servidores nos dimos cuenta que no contemplábamos parámetros null en la mayoría de ellos. #9
- Al realizar las pruebas con respecto al contenido Emisora, nos dimos cuenta de que no disminuíamos la duración de la misma al eliminar algún contenido de ella. #10
- Al realizar las pruebas de valores frontera con respecto a la duración de la canción, nos dimos cuenta de que no contemplábamos los casos en los que se especificara una duración menor o igual a 0. #11
- La clase SessionIdentifierGenerator debería ser estática y no lo es.#13

5. Estadísticas

5.1. Errores encontrados diariamente y semanalmente

Tomando como referencia tres semanas de duración del proyecto, podríamos decir que hemos encontrado menos de un error diariamente y un poco más de dos errores semanalmente. Este valor no es muy estable ya que los días en los que nos dedicamos a la realización de las pruebas, se encontraron muchos más errores que en el resto.

5.2. Nivel de progreso en la ejecución de las pruebas.

Toda la suite de pruebas se ejecuta sin problemas.

5.3. Informe de errores abiertos y cerrados por nivel de criticidad

5.3.1. Errores cerrados

- Canciones con duración menor o igual a 0.
- Disminución de la duración de la Emisora al eliminar contenidos de ella.
- Parámetros null en métodos de servidores.
- Publicidad añadida siempre al principio.
- Clase interna no estática.

5.3.2. Errores abiertos

No tenemos ningún error abierto actualmente.

5.4. Evaluación global del estado de calidad y estabilidad actuales

Podríamos afirmar con confianza que el nivel actual de calidad y de estabilidad es bueno ya que en la suite de pruebas cubrimos casi todas las posibilidades para nuestra aplicación como se puede comprobar en la Cobertura.

6. Otros aspectos de interés

Debido a las características del proyecto las herramientas usadas son JUnit (automatización de pruebas), Cobertura (Validar la calidad de nuestras pruebas), FindBugs (pruebas estructurales), Checkstyle (pruebas estructurales) y PMD (pruebas estructurales). Hemos decidido no usar herramientas de generación de datos debido a la simplicidad de los valores frontera de nuestros métodos.