Informe de Calidad: Historia de usuario “US-244924-Ver paradas”

Resumen

A continuación, se realizará el análisis de calidad del código que da funcionalidad a la historia de usuario “*US-244924-Ver paradas*”.

Document ID:	US-244924-VerParadas-QAReport
Departamento:	CALIDAD Y AUDITORÍA
Tipo:	ANÁLISIS
Privacidad:	CONFIDENCIAL
Estado:	ENTREGABLE
Fecha:	22/11/17
Autores:	Becerril Crespo, Adrián
Revisores:	Fernández Herrera, Manuel; Sobaler Cuerno, Elisa; Ibrain Rodríguez, Álvaro; Diego Martínez, Javier.

1. Introducción.

El documento contiene el análisis de calidad realizado a la implementación de la historia de usuario "*US-244924-Ver paradas*". En este informe se recogen los análisis inicial, intermedio y final de la implementación, los fallos encontrados y las medidas correctoras que se han aplicado.

2. Análisis inicial.

La situación del código al inicio del análisis mostraba una puntuación A,A,A en las medidas Reliability, Security, Maintainability, respectivamente.

A pesar de obtener la máxima calificación en los 3 apartados anteriormente mencionados, aún existe margen de mejora de la calidad del código. Esto se debe a que en el apartado de Maintainability se han generado treinta y cinco nuevos code smells que aumentan el total de la deuda técnica en cinco horas. En los apartados restantes se considera que se ha alcanzado la calidad óptima debido a que no se han encontrado nuevos bugs en el apartado de Reliability y tampoco nuevas vulnerabilidades en el apartado de Security, por tanto, el margen de mejora es nulo en estos dos casos.

La forma en la que se han conseguido mejorar dichos resultados será explicado en los siguientes apartados.

3. Errores encontrados y estrategia de mejora.

Como se ha comentado anteriormente, el margen de mejora de la calidad del código del proyecto se basa en tratar de resolver al máximo el número de code smells generados.

Para establecer un cierto orden, se han corregido en un primer instante los errores críticos, después los errores mayores y por último los errores menores. Es decir, el orden establecido para corregir los errores ha sido en base a la importancia de éstos para la calidad del código. Los code smells cuya gravedad se marquen como "info" no serán tratados al considerarse poco relativos. Cabe detallar, que se han intentado corregir los code smells citados anteriormente tanto de la parte nueva del proyecto como de la antigua que por un motivo u otro se consideró que se debían resolver en un futuro.

- Por tanto, los errores de carácter crítico que se han encontrado, así como la estrategia de mejora establecida, son los siguientes:

  - **ParserJson:** se deberán sustituir dos imports que hacen referencia a las dos clases existentes en el paquete dao por un solo import que incluya el paquete dao directamente. Se deberá establecer una constante para "UTF-8" y otra para "resources" de manera que no haya repeticiones en el código.

  - **ListLineasAdapter:** se detecta que se debe eliminar la duplicidad del "if infernal" generado a la hora de colorear las diferentes líneas de autobuses. Se decide dejar la corrección de este error para un futuro al no detectar una forma más óptima de escribir dicho bloque de código.

  Los errores críticos generados con esta historia de usuario ya integrada en el proyecto quedarían solucionados.

- A continuación, realizaremos el mismo proceso con los errores mayores:

  - **ParserJson:** se indica que existe código duplicado debido a los métodos readParada y readParadaGlobal. Se decide, por motivos de claridad del código, así como de modularidad, no corregir dicho error ya que permiten una mejor distinción de cuando se necesita leer una parada o bien una parada de línea (los campos provenientes del Json son diferentes lo que da lugar a poco margen de mejora del código).

  - **ListLineasPresenter:** se indica que el atributo de tipo Context nunca llega a ser utilizado más allá de su asignación. Se decide, por motivos de que en un futuro es bastante probable que llegue a utilizarse dicho atributo, no corregir dicho error. Supondría una pérdida de tiempo en un futuro buscar la forma de conseguir ese atributo cuando ya ha sido realizado previamente dicho trabajo.

  - **MainActivity:** existe código comentado que debe ser eliminado.

  - **ParadasFragment:** existe código que no se utiliza nunca, por tanto, se debe eliminar dicho código al ser ocioso.

  Los errores mayores generados con esta historia de usuario ya integrada en el proyecto quedarían solucionados.

- A continuación, realizaremos el mismo proceso con los errores menores:

  - **ParadasTodasFragment, ListParadasLineaPresenter, ListParadasLineaAdapter:** se deben eliminar imports que nunca llegan a ser utilizados.

  - **LineasFragment, ParadasFragment:** existe código que no se utiliza nunca, por tanto, se debe eliminar dicho código al ser ocioso.

  Los errores menores generados con esta historia de usuario ya integrada en el proyecto quedarían solucionados.

4. Análisis intermedio.

Después de tratar de corregir los errores citados en el anterior apartado. Nos encontramos que la situación del código continúa siendo de A,A,A en las medidas Reliability, Security, Maintainability, respectivamente. Se ha reducido el número de code smells de treinta y cinco a veintidos, disminuyendo de esta manera la deuda técnica en una hora.

A pesar de seguir obteniendo la máxima calificación en los 3 apartados anteriormente mencionados y de haber intentado corregir el máximo de code smells, se ha detectado que aún existen code smells que pueden resolverse. Esto se debe a que se han cometido errores a la hora de corregir los code smells del apartado anterior provocando que algunos de ellos no hayan sido corregidos del todo.

La forma en la que se han conseguido mejorar dichos resultados será explicado en los siguientes apartados.

5. Errores encontrados y estrategia de mejora.

- Los errores de carácter crítico que anteriormente se intentaron corregir, pero no han sido corregidos de la forma correcta son los siguientes:

  - **ParserJson:** se debe modificar el nombre de las constantes declaradas transformando las minúsculas en mayúsculas.

  Los errores críticos generados con esta historia de usuario ya integrada en el proyecto quedarían solucionados.

- A continuación, realizaremos el mismo proceso con los errores menores:

  - **ParadasFragment:** se debe eliminar un import que nunca es utilizado y que no se divisó en la corrección anterior.

  Los errores menores generados con esta historia de usuario ya integrada en el proyecto quedarían solucionados.

6. Análisis final.

Después de tratar de corregir los errores citados en los anteriores apartados. Nos encontramos que la situación del código continúa siendo de A,A,A en las medidas Reliability, Security, Maintainability, respectivamente. Se ha reducido el número de code smells de treinta y cinco a diecinueve, disminuyendo de manera significativa la deuda técnica en una hora.

7. Sumario.

En este entregable, se ha expuesto el análisis de calidad de la historia de usuario "*US-244924-VerParadas*" detallando los cambios que se han ido realizando en la implementación para mejorar la calidad del código.

Se puede concluir que con el análisis realizado se ha conseguido mejorar razonablemente la calidad del código.