PS17 – Plan de pruebas: Historia de usuario "Buscar parada por texto"

Resumen

El presente plan detalla el diseño de pruebas unitarias, integración y aceptación ideado para la historia de usuario *"Buscar parada por texto",* de manera que pueda ser aplicado con el objetivo de detectar la presencia de posibles errores en el código desarrollado.

Document ID:	PS17/00/2017-PP003-US241948-BuscarParadaPorTexto
Departamento:	Procesos de Ingeniería de Software, dentro del proyecto integrado
Tipo:	PLANIFICACIÓN
Privacidad:	CONFIDENCIAL
Estado:	ENTREGABLE
Versión:	1.0.4
Fecha:	15/11/2017
Autores:	Sainz-Maza Ruiz, Javier
Revisores:	Cerezo Fernández, Elsa; Martínez Vila, Javier; Oslé García, Luis; Sainz-Maza Ruiz, Javier; Solar Iglesias, Fernando;

HISTORIAL DE CAMBIOS

Versión	Fecha	Cambio	Responsable
1.0.0	07/11/2017	Creación del documento.	Sainz-Maza Ruiz, Javier
1.0.1	08/11/2017	Descrito el caso de uso a probar.	Sainz-Maza Ruiz, Javier
1.0.2	09/11/2017	Incorporadas las pruebas de aceptación acordadas con el *Product Owner*.	Sainz-Maza Ruiz, Javier
1.0.3	12/11/2017	Añadidas las pruebas unitarias para "Buscar paradas por texto".	Sainz-Maza Ruiz, Javier
1.0.4	14/11/2017	Descritas las pruebas de integración a implementar.	Sainz-Maza Ruiz, Javier
1.0.5	14/11/2017	Revisar y corregir fallos en el documento.	Cerezo Fernández, Elsa

1. Introducción

A lo largo de este documento se recoge el conjunto de pruebas unitarias, de integración y de aceptación con el objetivo de poder implementarse posteriormente y descubrir así la presencia de posibles errores en el código desarrollado para la historia de usuario.

Primero, se detalla el caso de uso que corresponde a la historia de usuario #241948-BuscarParadaPorTexto para definir, a continuación, cada una de las pruebas indicadas.

2. Caso de uso: Buscar parada por texto

Identificador:	#241948
Título:	Buscar parada por texto
Descripción:	El usuario visualiza el listado de todas las paradas del transporte urbano de Santander (TUS) que coinciden con el texto de búsqueda introducido por el usuario: nombre, alias, número y notas.
Actores:	Usuario habitual del TUS.
Secuencia:	1. El usuario visualiza el listado de paradas pudiendo desplazarse por éste. 2. El usuario pincha sobre el icono de búsqueda situado en la parte superior de la interfaz. 3. El usuario introduce el texto de búsqueda en el campo que se visualiza. 4. El sistema procesa el texto introducido, mostrando en pantalla las paradas que coinciden en alguno de los campos: nombre, alias, número y notas.
Extensiones:	4.a. Si no se puede realizar la búsqueda se notifica al usuario de la situación.

3. Pruebas aceptación

A continuación, se muestran las pruebas a realizar automáticamente sobre la interfaz de usuario, haciendo uso de las herramientas correspondientes del entorno de desarrollo, de manera que se compruebe que las siguientes interacciones muestran los resultados previstos.

PA1: Comprobar filtrado del listado de paradas: nombre de parada

1. El usuario pulsa sobre el icono de búsqueda de la app en la parte superior de la interfaz.
2. El usuario introduce el texto con el que desea filtrar en el campo de búsqueda.
3. El sistema lista las paradas que contengan como nombre el texto "ojaiz".

   El resultado esperado es poder visualizar el listado de paradas: *447, Ojaiz 166; 446, Ojaiz 89; 355, Ojaiz; 54, Barrio de Ojaiz 7*.

PA2: Comprobar filtrado del listado de paradas: número de parada

1. El usuario pulsa sobre el icono de búsqueda de la app en la parte superior de la interfaz.
2. El usuario introduce el texto con el que desea filtrar en el campo de búsqueda.
3. El sistema lista las paradas que contengan como número el texto "477".

   El resultado esperado es poder visualizar el listado de paradas: *477, Jose Ortega y Gasset - 2.*

PA3: Comprobar filtrado del listado de paradas: alias de parada

1. El usuario pulsa sobre el icono de búsqueda de la app en la parte superior de la interfaz.
2. El usuario introduce el texto con el que desea filtrar en el campo de búsqueda.
3. El sistema lista las paradas que contengan como alias el texto "mi casita".

   El resultado esperado es poder visualizar el listado de paradas: *499, Camarreal Peñacastillo.*

PA4: Comprobar filtrado del listado de paradas: anotación de parada

1. El usuario pulsa sobre el icono de búsqueda de la app en la parte superior de la interfaz.
2. El usuario introduce el texto con el que desea filtrar en el campo de búsqueda.
3. El sistema lista las paradas que contengan como anotación el texto "domino".

   El resultado esperado es poder visualizar el listado de paradas: *481, Jeronimo Sainz de la Maza (Mercado Mexico).*

4. Pruebas integración

Las pruebas de integración se realizarán mediante una estrategia incremental guiada por la funcionalidad implementada.

A continuación, se muestra el diseño en el que se basa la arquitectura de nuestro producto, el modelo MVP (*Model View Presenter*):

Figura 1. Diseño de arquitectura global

En la capa *Model*, se recogen aquellos fragmentos de la aplicación dedicados a la captura de datos. La capa *Presenter* se dedica al tratamiento de los datos obtenidos, preparándolos para ser mostrados por la capa *View*.

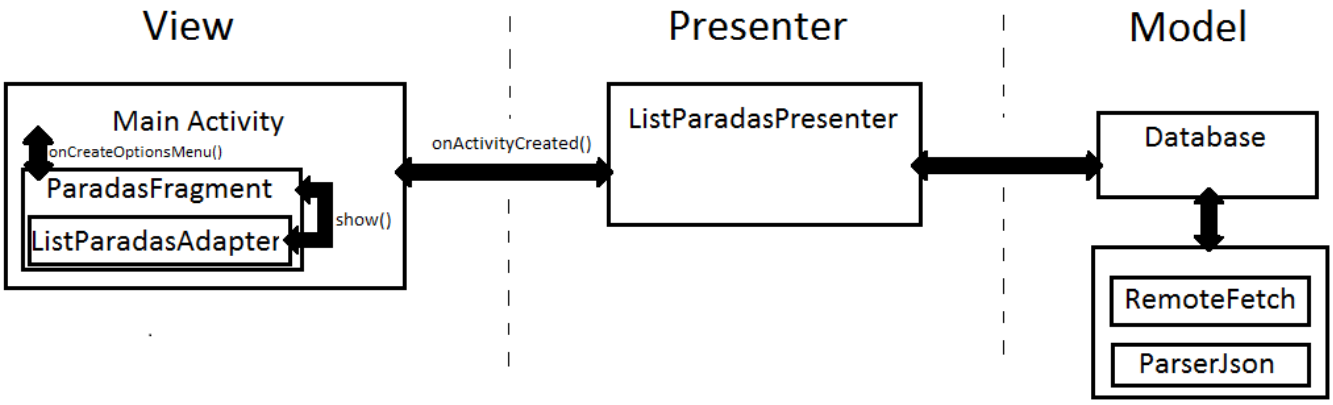

Figura 2. Diseño de arquitectura de la historia de usuario "Buscar parada por texto"

En el anterior diagrama (Figura 2) se puede observar los distintos fragmentos de la aplicación implementada a nivel arquitectónico. El método que da funcionalidad a la historia de usuario *"Buscar parada por texto"* es *onCreateOptionsMenu()* el cuál se utiliza en la capa *View*, ya que sirve para determinar la forma en la que se muestran los datos.

En este apartado se diseñan las pruebas de integración, encargadas de comprobar si las funcionalidades implementadas funcionan correctamente al integrarse. Por ello, al diagrama anterior (Figura 2) se le añade las funcionalidades anteriormente implementadas. En el siguiente diagrama, se muestra la interacción que se produce entre los elementos de nuestro producto y los métodos que les da la funcionalidad a las historias de usuario *"Listar paradas"*, *"Ordenar paradas alfabéticamente"*, *"Visualizar logo en arranque"*, *"Buscar parada por texto"* y *"Consultar listado de líneas"*.

Tras haber comprobado mediante pruebas de integración el buen funcionamiento del sistema integrado por las funcionalidades *"Listar paradas"*, *"Ordenar paradas alfabéticamente"* y *"Visualizar logo en arranque"* (en el anterior sprint se comprobó la

correcta respuesta del sistema mostrando la actividad en la que se muestra el logo), por lo tanto, se comprobará el buen funcionamiento del producto al integrar las historias *"Consultar listado de líneas"* y *"Buscar parada por texto"*.

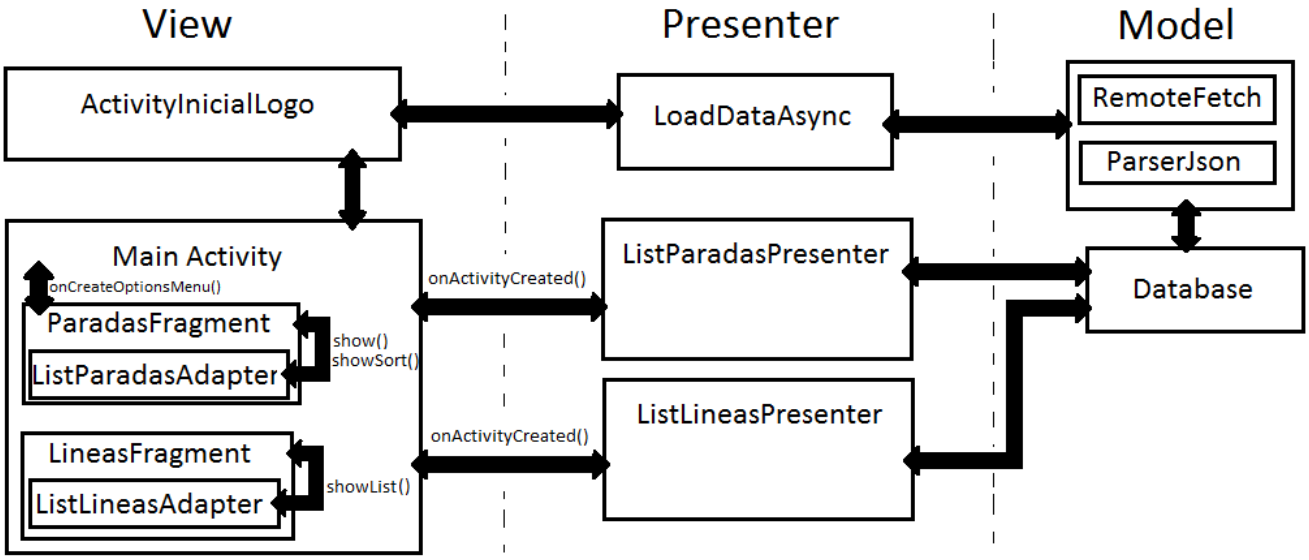

Figura 3. Diseño de arquitectura con todas las funcionalidades integradas

A partir del diseño planteado, y teniendo en cuenta las operaciones que se implican, se ha diseñado el caso de prueba que se indica a continuación:

PI1. Probar que se devuelve un listado de paradas que coincida con la búsqueda introducida desde la pantalla del listado de paradas. Con ello se comprobará el correcto funcionamiento de la historia de usuario *"Buscar parada por texto"* integrada con el resto de funcionalidades implementadas. Este caso se realizará obteniendo los datos de un Json local, dado que los ficheros del repositorio remoto son actualizados, por lo tanto, los datos de estos archivos pueden variar invalidando las pruebas realizadas.

5. Pruebas unitarias

Se listan a continuación los casos correspondientes a las pruebas unitarias. Como base de prueba se utiliza el siguiente listado de paradas:

Nombre	Número	Identificador	Alias	Nota
Avenida de Cantabria 10	213	322	null	null
Avenida Cantabria 12	221	323	Mi casita	null
Los Castros 63	73	45	null	Domino

PU1. Se deberá probar el método *searchFilterList()* de la clase *ParadasFragment* que es utilizado para realizar el filtrado.

Al introducir como entrada el listado de paradas proporcionado, se deberá comprobar que con las siguientes cadenas de caracteres de filtrado se obtiene como resultado el listado de paradas filtradas esperadas.

Cadena de caracteres	Paradas
Avenida	Avenida de Cantabria 10 Avenida Cantabria 12
Mi casita	Avenida Cantabria 12
Domino	Los Castros 63

PU2. Se deberá probar el método *searchFilterList()* de la clase *ParadasFragment,* utilizado para realizar el filtrado, de manera que al introducir el listado de paradas base y una cadena de caracteres vacía se devuelva un listado de paradas vacío.

6. Sumario

El presente documento ha detallado el diseño completo del conjunto de pruebas para el caso de uso *"Buscar parada por texto"*, que debe ser usado a continuación para implementar cada una de ellas con la finalidad de comprobar la presencia de errores en el código implementado.