PS17 – Plan de pruebas: Historia de usuario "Añadir parada a grupo"

Resumen

El presente plan especifica los casos de prueba diseñados para las pruebas unitarias, integración y aceptación con el objetivo de comprobar la correcta implementación de la historia de usuario *"Añadir parada a grupo",* de manera que pueda ser aplicado con el fin de detectar la presencia de posibles errores en el código desarrollado.

Document ID:	PS17/00/2017-PP007-US242777-AñadirParadaAGrupo
Departamento:	Procesos de Ingeniería de Software, dentro del proyecto integrado
Tipo:	PLANIFICACIÓN
Privacidad:	CONFIDENCIAL
Estado:	ENTREGABLE
Versión:	1.0.5
Fecha:	27/11/2017
Autores:	Cerezo Fernández, Elsa
Revisores:	Sainz-Maza Ruiz, Javier; Martínez Vila, Javier; Oslé García, Luis; Sainz-Maza Ruiz, Javier; Solar Iglesias, Fernando

HISTORIAL DE CAMBIOS

Versión	Fecha	Cambio	Responsable
1.0.0	22/11/2017	Creación del documento y redacción de pruebas de aceptación.	Cerezo Fernández, Elsa
1.0.1	23/11/2017	Redacción de las pruebas unitarias.	Cerezo Fernández, Elsa
1.0.2	23/11/2017	Continuación de redacción de las pruebas unitarias.	Cerezo Fernández, Elsa
1.0.3	24/11/2017	Redacción de las pruebas de integración.	Cerezo Fernández, Elsa
1.0.4	25/11/2017	Cambios en la redacción del documento.	Cerezo Fernández, Elsa
1.0.5	27/11/2017	Cambios en la redacción de las pruebas unitarias.	Cerezo Fernández, Elsa

1. Introducción

A continuación, se recogen las pruebas que posteriormente serán implementadas para examinar el correcto funcionamiento de las funcionalidades realizadas, en especial de la historia de usuario "Añadir parada a grupo".

Sin embargo, antes de detallar los casos de prueba, se especificará el caso de uso que corresponde a la actual historia de usuario.

2. Caso de uso: Añadir parada a grupo

Identificador:	#242777
Título:	Añadir parada a grupo
Descripción:	El usuario desde el listado de paradas, manteniendo seleccionada una parada determinada, puede asignarla a distintos grupos para mayor comodidad.
Actores:	Usuario habitual del TUS.
Secuencia:	1. El usuario visualiza el listado de paradas. 2. El usuario mantiene pulsada alguna de las paradas mostradas. 3. El sistema muestra un listado de los grupos posibles. 4. El usuario selecciona el grupo al que quiere asignar la parada. 5. El sistema introduce la asignación en la base de datos. 6. El sistema muestra la parada seleccionada dentro del grupo asignado en el apartado *"Grupos"*.
Extensiones:	5.a. Si la asignación de la parada a un determinado grupo no puede ser recogida correctamente en la base de datos se notificará el error.

3. Pruebas aceptación

En este apartado, se muestran las pruebas que se realizarán sobre la interfaz de usuario, haciendo uso de las herramientas correspondientes, para comprobar si la funcionalidad implementada responde de forma correcta.

Antes de ejecutar las pruebas se deberá realizar un borrado de las asignaciones que hubiera registradas en la base de datos haciendo uso del método *eliminaParadasDeGrupos()* para que las pruebas se efectúen desde un estado inicial conocido.

PA1-US242777: Añadir parada a grupo, asignación y recogida en la base de datos correctas

1. El usuario selecciona la opción "*Ordenar alfabéticamente"* en el despliegue de acciones.
2. El usuario mantiene pulsada la parada *"296 Abilio Garcia baron 1 (hote4l expres)"* desde la lista ordenada de paradas.
3. El sistema muestra el listado de grupos al que se puede asignar la parada.
4. El usuario selecciona el grupo *"Aguamarina"*.

5. El usuario selecciona el apartado de *"Grupos"*.
6. El sistema muestra la parada seleccionada en el paso 1 dentro del grupo *"Aguamarina"* en el apartado *"Grupos"*.

   El resultado esperado es poder visualizar la parada *"296 Abilio Gracia baron 1 (hote4l expres)"* dentro del grupo *"Aguamarina"* en el apartado *"Grupos"*.

PA2-US242777: Añadir parada a grupo, asignación y recogida en la base de datos incorrecta

1. El usuario selecciona la opción "*Ordenar alfabéticamente"* en el despliegue de acciones.
2. El usuario mantiene pulsada la parada *"296 Abilio Garcia baron 1 (hote4l expres)"* desde la lista ordenada de paradas.
3. El sistema muestra el listado de grupos al que se puede asignar la parada.
4. El usuario selecciona el grupo *"Aguamarina"*.
5. El sistema notifica que no se han podido recoger los cambios realizados.

   El resultado esperado es que el sistema notifique el error de inserción en la base de datos.

PA3-US242777: No hacer ninguna asignación de paradas a grupos, estado de apartado *Grupos* correcto

1. El usuario selecciona el apartado *Grupos* sin haber asignado ninguna parada a ningún grupo.
2. El sistema notifica que no hay paradas para mostrar.

   El resultado esperado es que el sistema notifique no hay ninguna asignación de paradas a grupos.

4. Pruebas integración

Las pruebas de integración se deberían realizar mediante una estrategia incremental guiada por la funcionalidad implementada. Sin embargo, dado que las funcionalidades a integrar no son ni muy numerosas, ni muy dependientes sobre el resto de funcionalidades ya probadas, las pruebas de integración se realizarán mediante la estrategia *Big Bang*.

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 componentes 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*.

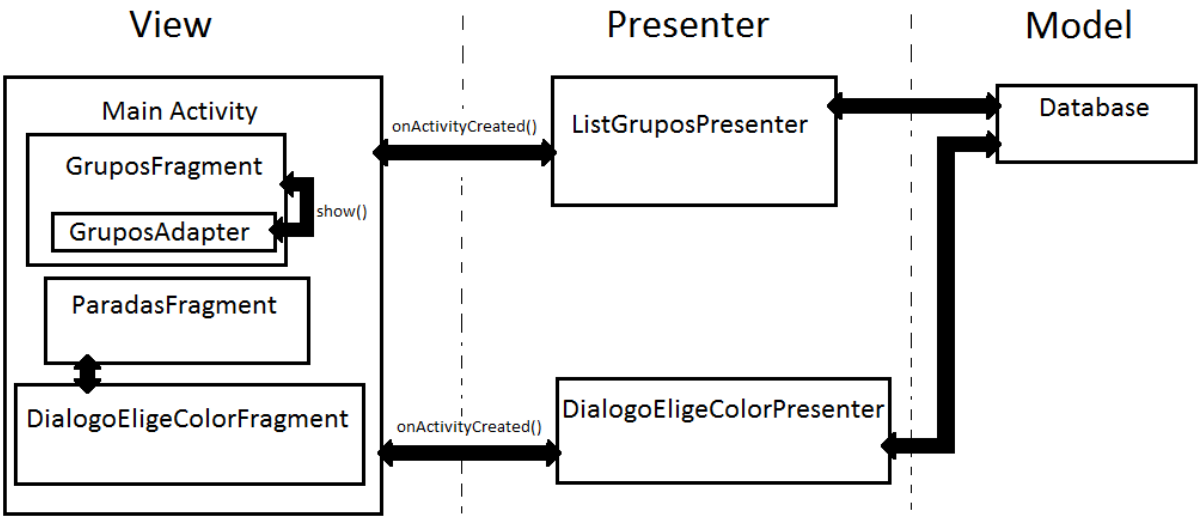

Figura 2. Diseño de arquitectura de la historia de usuario "Añadir parada a grupo"

En este apartado se diseñan las pruebas de integración, encargadas de comprobar si las funcionalidades implementadas se realizan correctamente al integrarse. Por ello, al diagrama anterior (Figura 2), en el que se pueden observar los distintos fragmentos de la aplicación implementada a nivel arquitectónico, se le añaden la funcionalidad de la historia de usuario *"Añadir parada a grupo"*. En el siguiente diagrama, se muestra la interacción que se produce entre los elementos de nuestro producto.

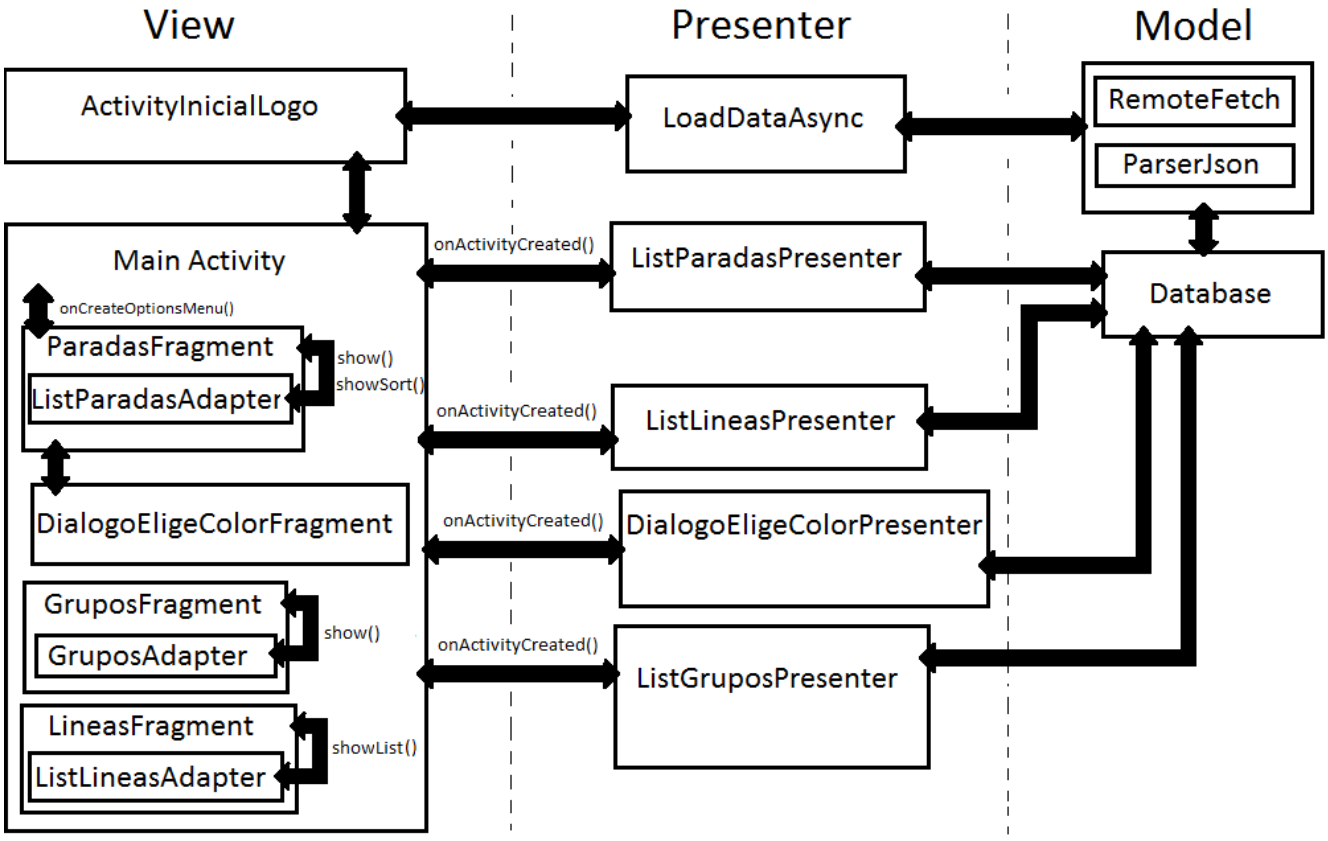

Figura 3. Diseño de arquitectura con todas las funcionalidades integradas

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:

PI-US242777. Probar que se devuelve un listado ordenado de paradas en el apartado de *Paradas* y que, posteriormente, de las paradas que se muestran, se pueden realizar distintas asignaciones y se devuelve un listado de asignaciones correctas, además de la lista que después se mostrará en el apartado *Grupos*.

Se asignarán, de la lista ordenada de paradas, las tres primeras paradas resultantes (***296*** *Abilio Garcia baron 1 (hote4l expres),* ***455*** *Adarzo,* ***235*** *Adarzo*) al grupo *Aguamarina*, la cuarta parada resultante (***489*** *Albert Einstein. 14*) al grupo *Granate,* y la quinta y sexta parada resultantes (***171*** *Alcalde vega lamera 1,* ***83*** *Alto de la Peña*) al grupo *Amarillo.*

Los listados que se esperan obtener son los siguientes:

- Lista de objetos *GrupoParada* que recoge la asignación de paradas a grupos

Id	Grupo-parada
1	- | Aguamarina | #00C2C2 - | Abilio Garcia baron 1 (hote4l expres) | 296 | 90 | null | Null
2	- | Aguamarina | #00C2C2 - | Adarzo | 455 | 54 | null | null
3	- | Aguamarina | #00C2C2 - | Adarzo | 235 | 26416 | null | null
4	- | Granate | #7C0899 - | Albert Einstein. 14 | 489 | 41656 | null | null
5	- | Amarillo | #FFC501 - | Alcalde vega lamera 1 | 171 | 301 | null | null
6	- | Amarillo | #FFC501 - | Alto de la Peña | 83 | 232 | null | null

- Listado que recoge los objetos que se mostrarán en el apartado *Grupos*

Posición	Objeto
0	- | Aguamarina | #00C2C2
1	- | Abilio Garcia baron 1 (hote4l expres) | 296 | 90 | null | null
2	- | Adarzo | 455 | 54 | null | null
3	- | Adarzo | 235 | 26416 | null | null
4	- | Amarillo | #FFC501
5	- | Alcalde vega lamera 1 | 171 | 301 | null | null
6	- | Alto de la Peña | 83 | 232 | null | null
7	- | Granate | #7C0899
8	- | Albert Einstein. 14 | 489 | 41656 | null | null

Nota: *Las pruebas serán realizadas con un JSON local como base para evitar modificaciones en la fuente remota que puedan invalidar las pruebas. Además, el campo Id, tanto de Paradas como de Grupos, dado que dependiendo de la inserción en la base de datos tomará un valor u otro y como se pueden probar los resultados sin conocer ese valor, en el plan se indicará como '-' los campos respectivos a ese atributo.*

5. Pruebas unitarias

En el presente apartado se especificarán las pruebas unitarias que se realizarán para comprobar el correcto funcionamiento, a nivel de código, de la historia de usuario implementada.

En los casos de prueba que se especifican a continuación, se probará la correcta asignación de las paradas a grupos. Para ello se utilizarán como base de prueba los siguientes listados de paradas y grupos:

- Listado de paradas

DbId	Nombre	Número	Identificador	Alias	Notas
1	Avenida de Cantabria 10	213	322	null	null
2	Avenida Cantabria 12	221	323	null	null
3	Los Castros 63	73	45	null	null

- Listado de grupos

Id	Nombre	Color
1	Aguamarina	#00C2C2
2	Amarillo	#FFC501
3	Cian	#00A4EB

PU1-US242777. Se deberá probar que se realizan bien las asignaciones de paradas a grupos. Utilizando los listados de paradas y grupos dados, al realizar la asignación siguiente, se comprobará que, sobre cada objeto *GrupoParada* resultante de cada asignación, al realizar el método *getGrupoAsignado()* sobre el método *getParada()* aplicado al objeto se obtiene el id esperado (se muestran los resultados esperados en el siguiente apartado de resultados) :

- Asignación

  Las siguientes relaciones se realizarán creando objetos *GrupoParada,* los cuales se forman con un objeto parada y un objeto grupo:

  a. gp1(Avenida de Cantabria 10 ⇨ Aguamarina)
  b. gp2(Avenida Cantabria 12 ⇨ Amarillo)
  c. gp3(Los Castros 63 ⇨ Cian)

- Resultado

  A continuación, se indican los resultados esperados de la prueba unitaria desarrollada:

  a) gp1.getParada().getGrupoAsignado() = 1
  b) gp2.getParada().getGrupoAsignado() = 2
  c) gp3.getParada().getGrupoAsignado() = 3

PU2-US242777. Se deberá probar el método *getGruposConParadas()* de la clase *ListGruposPresenter,* al cual se le pasa como parámetros una lista de objetos *Grupos* y una lista de objetos *GrupoParada.* Para el presente caso de prueba, se facilitará el listado de grupos que se especifica al inicio del apartado de pruebas unitarias y el siguiente listado de objetos *GrupoParada*:

Id	Grupo-parada
1	1 | Aguamarina | #00C2C2 1 | Avenida de Cantabria 10 | 213 | 322 | null | null
2	1 | Aguamarina | #00C2C2 2 | Avenida Cantabria 12 | 221 | 323 | null | null
3	2 | Amarillo | #FFC501 3 | Los Castros 63 | 73 | 45 | null | null

El resultado que se espera obtener en la prueba es una lista como la que se muestra a continuación:

Posición	Objeto
0	1 | Aguamarina | #00C2C2
1	1 | Avenida de Cantabria 10 | 213 | 322 | null | null
2	2 | Avenida Cantabria 12 | 221 | 323 | null | null
3	2 | Amarillo | #FFC501
4	3 | Los Castros 63 | 73 | 45 | null | null

6. Sumario

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