

TRABAJO INGENIERÍA DE SOFTWARE: AMPLIACION APLICACIÓN DE NOTAS PARA ANDROID

Curso 2019-2020
Álvaro Echávarri, 737400
Abel Naya, 544125

1.-Resumen, introducción y objetivos.

Para el análisis, diseño y prueba de nuestra aplicación de notas para móvil, partimos de un simple esquema con 4 funcionalidades: crear nota, modificar nota, borrar nota y mostrar notas. Tal y como nos mostraron los profesores, la idea base era establecer una base de datos y una aplicación que comunique al usuario con esa base de datos. La base de datos almacena las notas y sus elementos y la aplicación muestra esas notas y las diferentes funcionalidades.

Siguiendo el esquema de prácticas propuesto por el profesorado comenzamos completando un código de prueba de nuestra aplicación, que nos sirvió más tarde para realizar los distintos diagramas de análisis que aparecen en este informe, más adelante completamos una versión de la aplicación algo más compleja, que permite dar un título y un cuerpo a cada nota.

Como último añadido, realizamos diagramas mucho más detallados y extensos que se encuentran incluidos también en este pdf como diagramas o fase de diseño, en estos también se añadió una nueva funcionalidad, las categorías, una nota podía tener una categoría y se podían filtrar por su nombre, la base de datos también se encargaría de almacenar estas categorías. Además añadimos el ordenar la lista de notas por título y por categoría tal y como pedía la entrega de este trabajo.

2.-ÍNDICE

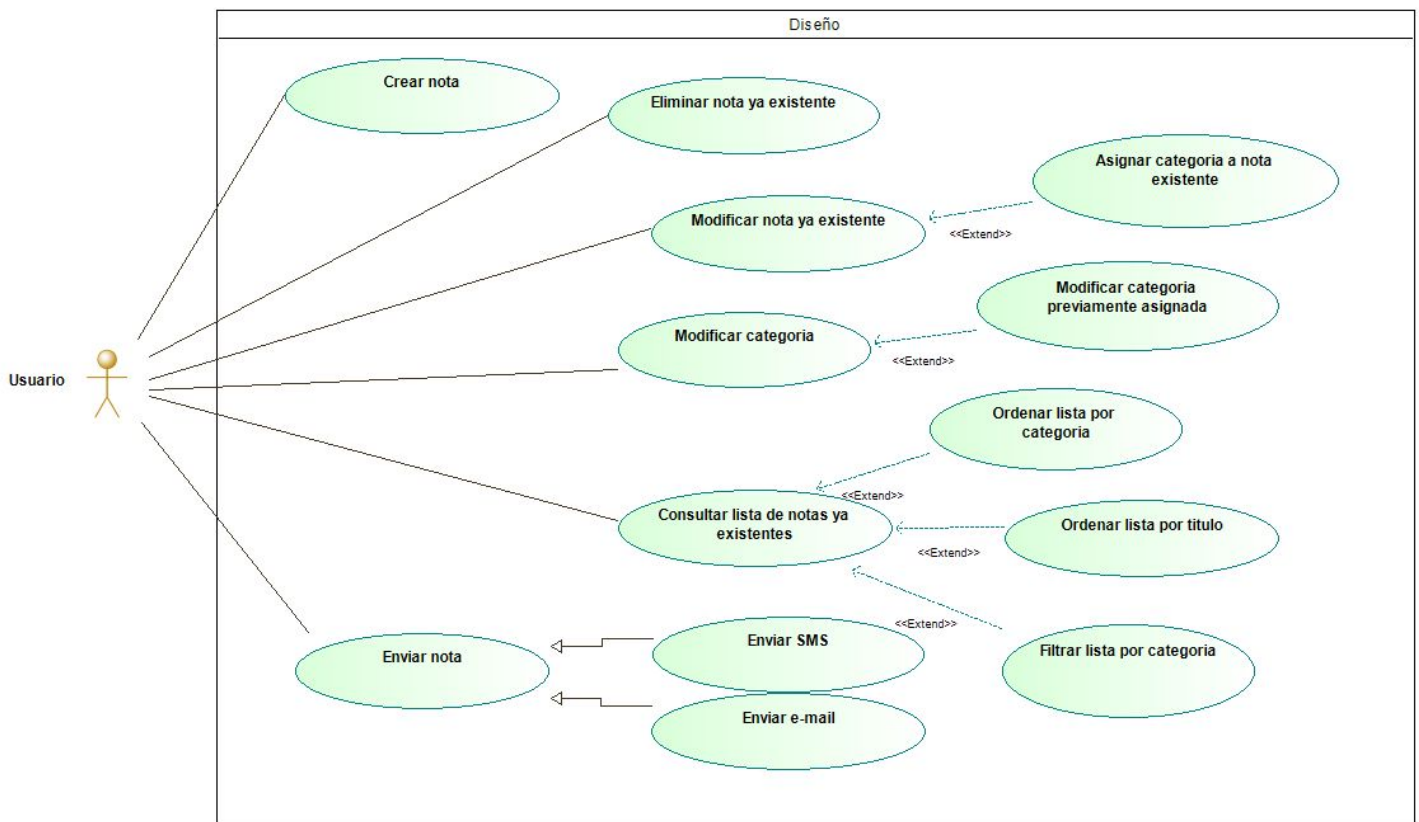
1.-Resumen, introducción y objetivos.	1
2.-ÍNDICE	2
3.-Requisitos.	3
3.1-Catálogo de requisitos.	3
3.2-Diagrama de casos de uso	4
3.2.1-Flujos requisitos funcionales	4
4.-Diagramas fase de análisis	6
4.1-Diagrama de clases	6
4.2-Diagramas de secuencia	7
4.3-Mapa de navegación	11
5.-Diagramas de fase de diseño del sistema	12
5.1-Diagrama de paquetes	12
5.2-Diagrama de componentes	13
5.3-Diagrama de despliegue	13
6.-Diagramas fase de diseño de objetos	14
6.1-Selección patrón de diseño	14
6.2-Diagrama de clases	15
6.3-Diagramas de secuencia	16
6.4-Modelo entidad-relación	23
7.-Pruebas	24
7.1-Pruebas de caja negra	24
7.2-Prueba de volumen	31
7.3- Análisis de los resultados	31
7.4-Pruebas de sobrecarga (monkey)	32
7.5- Descripción pruebas	33
8.-Resultados y conclusiones.	33
9.-Bibliografía	33

3.-Requisitos.

3.1-Catálogo de requisitos.

RF1	El sistema debe tener notas. Cada nota se identificara con un titulo (string), y un cuerpo (string).
RF2	El sistema debe permitir crear notas
RF3	El sistema debe permitir modificar el cuerpo y título de una nota.
RF4	El sistema debe permitir borrar una nota.
RF5	El sistema debe permitir mostrar una lista con todas las notas. No ordenadas de forma predeterminada.
RF6	El sistema debe tener categorías de notas. Cada categoría se identifica por un string.
RF7	El sistema debe permitir crear categorías.
RF8	El sistema debe permitir modificar el identificador de las categorías.
RF9	El sistema debe permitir eliminar categorías.
RF10	El sistema debe permitir asignar una categoría previamente creada a una nota.
RF11	El sistema debe permitir modificar la categoría asignada a una nota.
RF12	El sistema debe permitir eliminar la categoría asignada a una nota.
RF13	El sistema debe permitir ordenar la lista de notas por categoría.
RF14	El sistema debe permitir ordenar la lista de notas por título.
RF15	El sistema debe permitir mostrar sólo las notas de una categoría.
RF16	El sistema debe permitir enviar notas.
RNF 17	El sistema no va a almacenar más de 1000 notas.

3.2-Diagrama de casos de uso



3.2.1-Flujos requisitos funcionales

Crear nota:

-Flujo principal

- Inicio en el menú principal
- Desplegar menú.
- Darle a crear.
- Introducir un título y un cuerpo.
- Aceptar

Modificar nota ya existente:

-Flujo principal

- Inicio en el menú principal
- Elegir una nota
- Darle a editar
- Cambiar el título y cuerpo.
- Aceptar

Borrar nota ya existente:

-Flujo principal

- Inicio en el menú principal
- Elegir una nota
- Darle a eliminar.

Consultar lista de notas ya existente:

-Flujo principal

- El caso de uso se inicia cuando se arranca la app.
- El sistema recupera las notas y muestra sus títulos.

-Flujo alternativo:

- Cuando termina el caso de uso Crear Nota se inicia el caso de uso.
- Caso: Cuando termina el caso de uso Modificar Nota se inicia el caso de uso.
- Cuando termina el caso de uso Eliminar Nota se inicia el caso de uso.
- Cuando termina el caso de uso Eliminar categoría se inicia el caso de uso.
- Cuando termina el caso de uso Crear categoría se inicia el caso de uso.
- Cuando termina el caso de uso Modificar categoría se inicia el caso de uso.
- Cuando termina el caso de uso Asignar categoría se inicia el caso de uso.

Crear categoría:

-Flujo principal

- Inicio en el menú principal
- Desplegar menú
- Elegir categorías
- Desplegar menú
- Darle a añadir
- Introducir un nombre
- Aceptar

Modificar categoría:

-Flujo principal

- Inicio en el menú principal
- Desplegar menú
- Elegir categorías
- Elegir una categoría
- Darle a editar
- Cambiar el nombre
- Aceptar

Asignar categoría a nota ya existente:

-Flujo principal

- Inicio en el menú principal
- Elegir una nota
- Darle a editar
- Asignar una categoría.
- Aceptar

Eliminar categoría:

-Flujo principal

- Inicio en el menú principal
- Desplegar menú
- Elegir categorías
- Elegir una categoría
- Darle a eliminar

Ordenar lista de notas por categoría

-Flujo principal

- Inicio en el menú principal
- Elegir ordenar por categoría

Ordenar lista de notas por título

-Flujo principal

- Inicio en el menú principal
- Elegir ordenar por título

Mostrar sólo las notas de una categoría

-Flujo principal

- Inicio en el menú principal
- Elegir categoría en el desplegable
- Marcar opción filtrar por categoría

Enviar notas por E-MAIL

-Flujo principal

- Inicio en el menú principal
- Elegir una nota
- Darle a enviar por e-mail

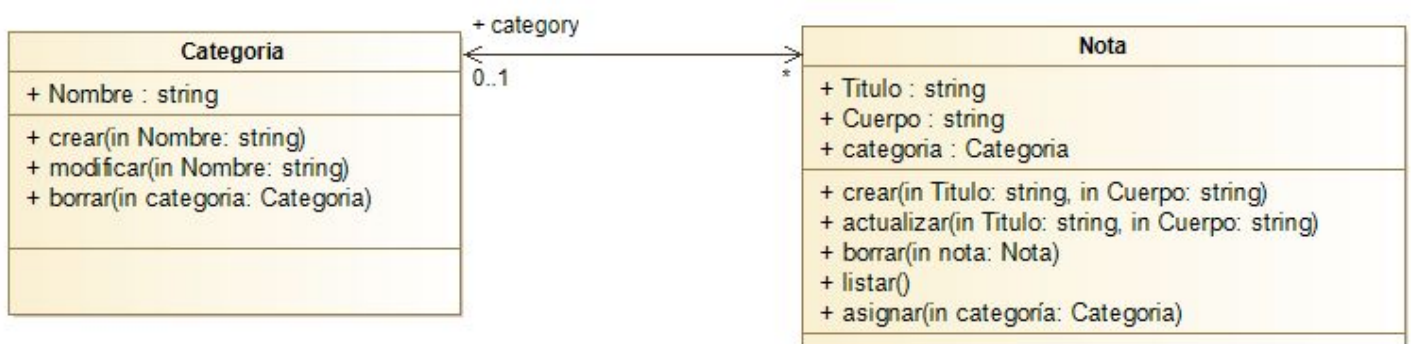
Enviar notas por SMS

-Flujo principal

- Inicio en el menú principal
- Elegir una nota
- Darle a enviar por sms

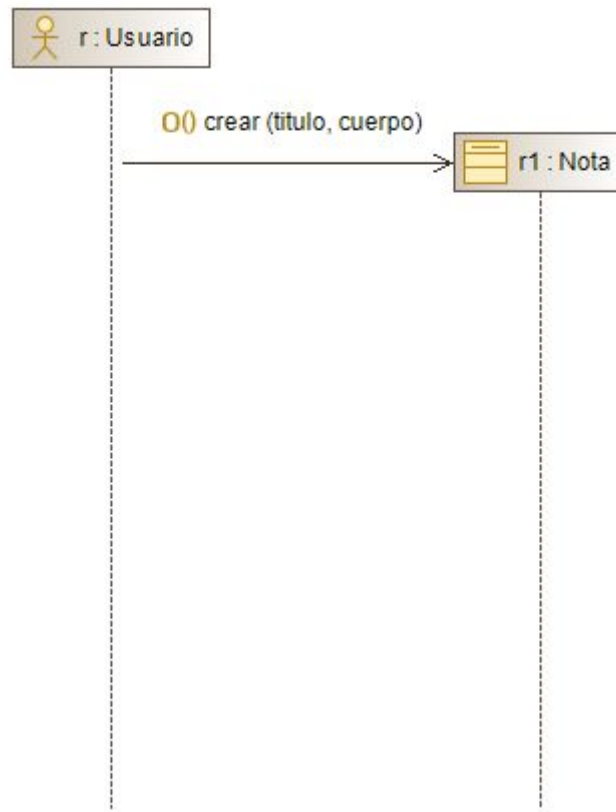
4.-Diagramas fase de análisis

4.1-Diagrama de clases

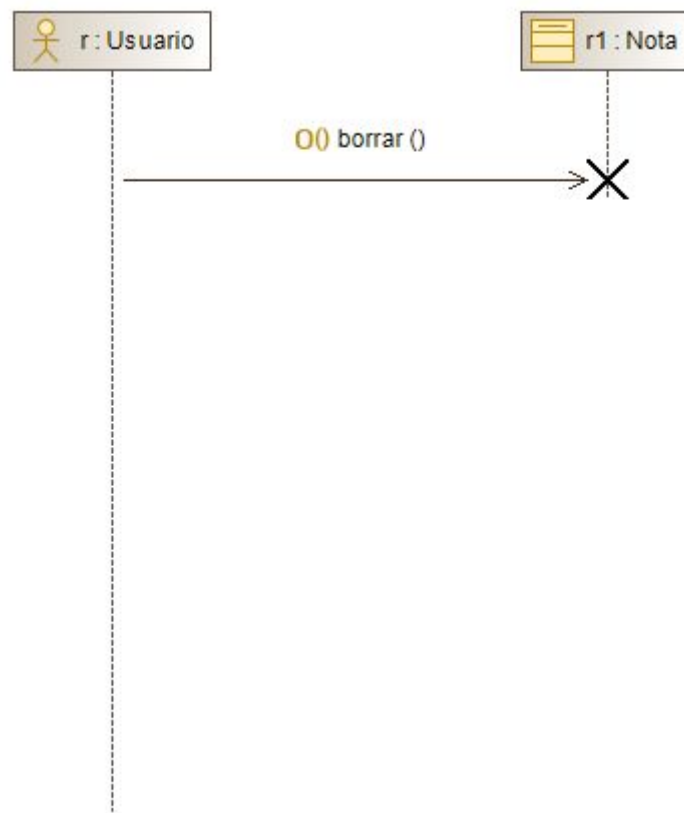


4.2-Diagramas de secuencia

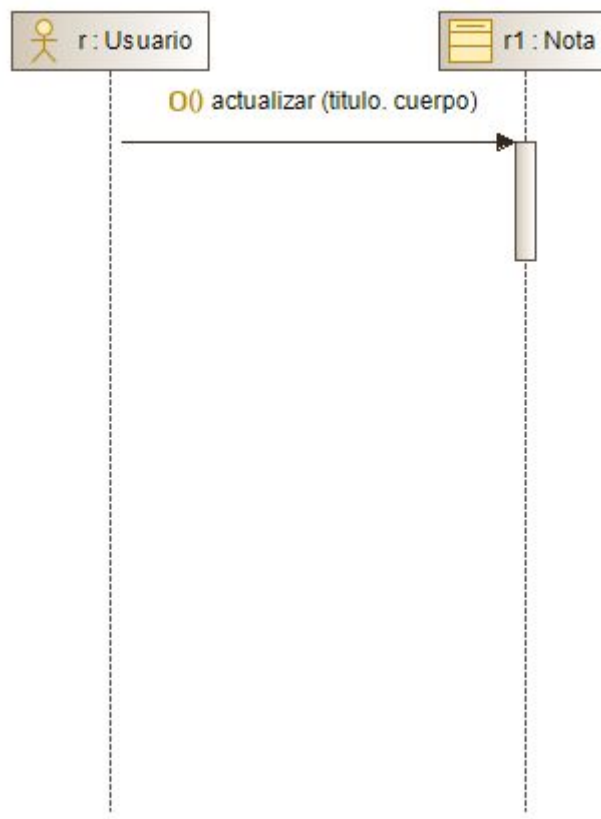
Crear nota



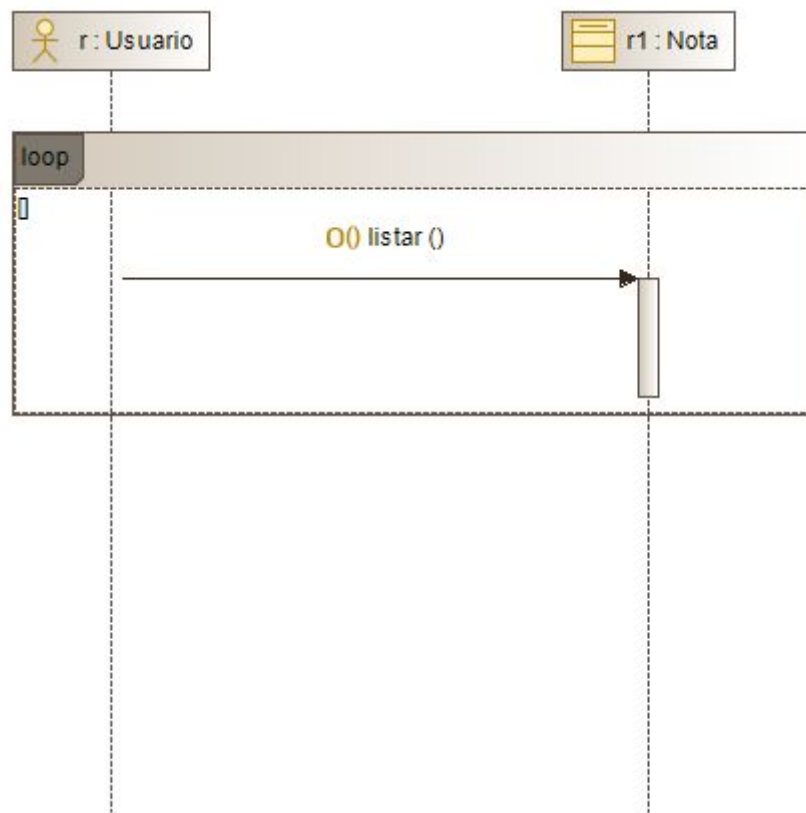
Borrar nota



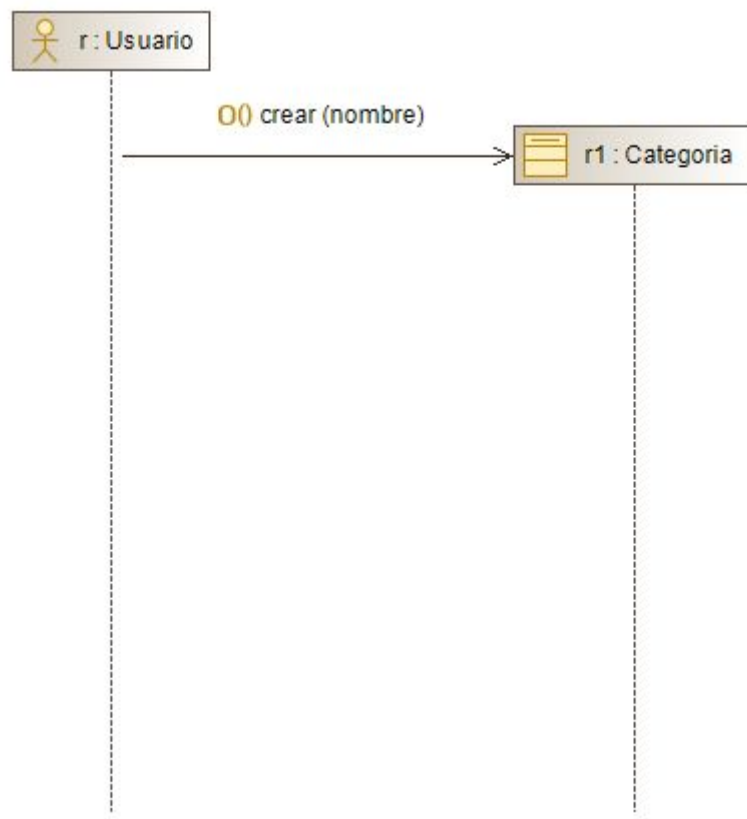
Actualizar nota



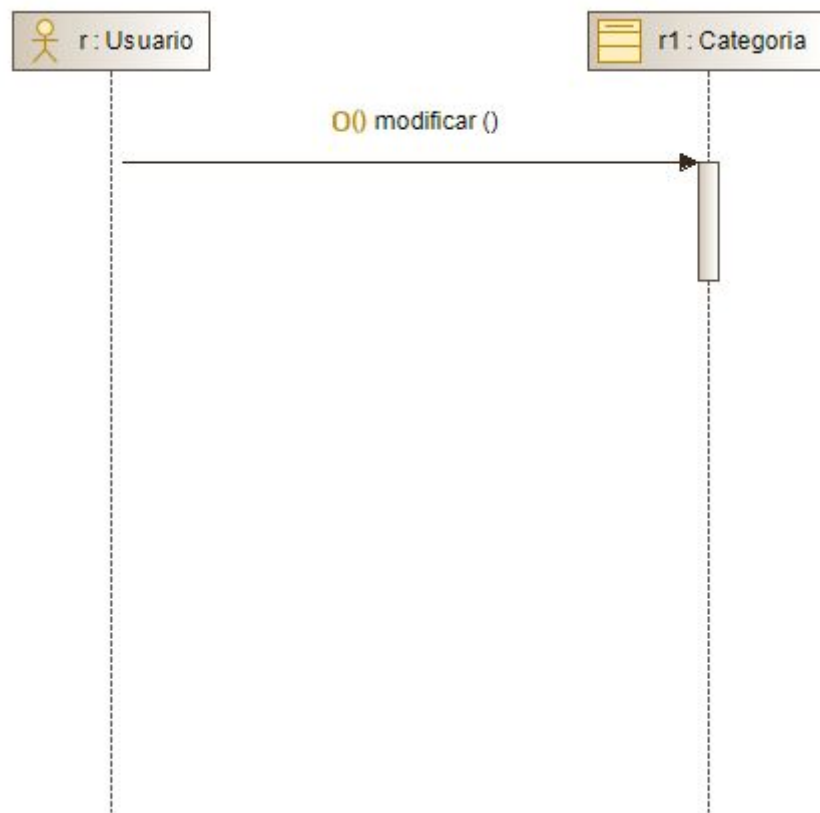
Consultar lista de notas ya existentes



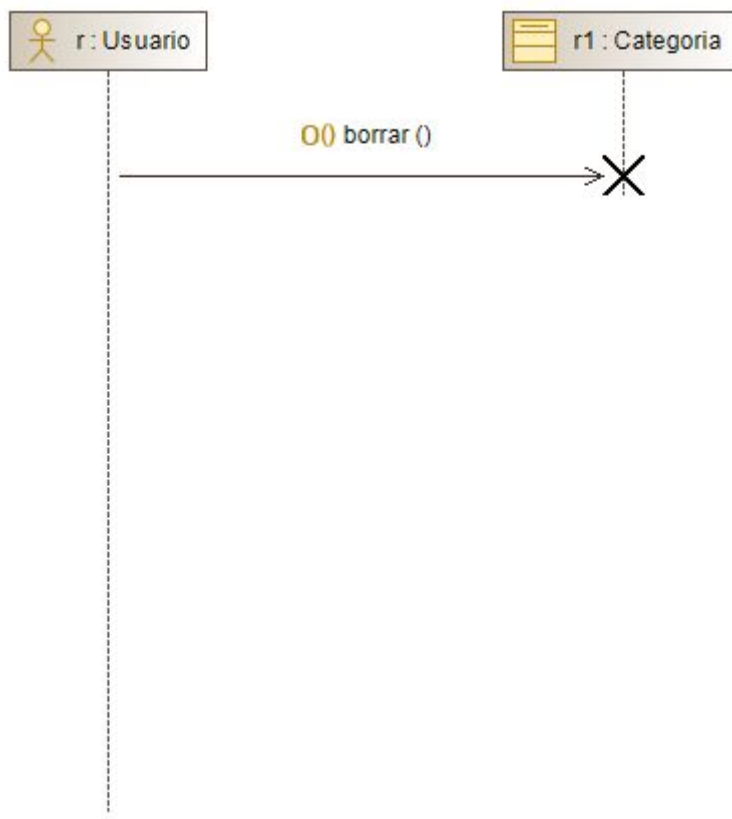
Crear categoría



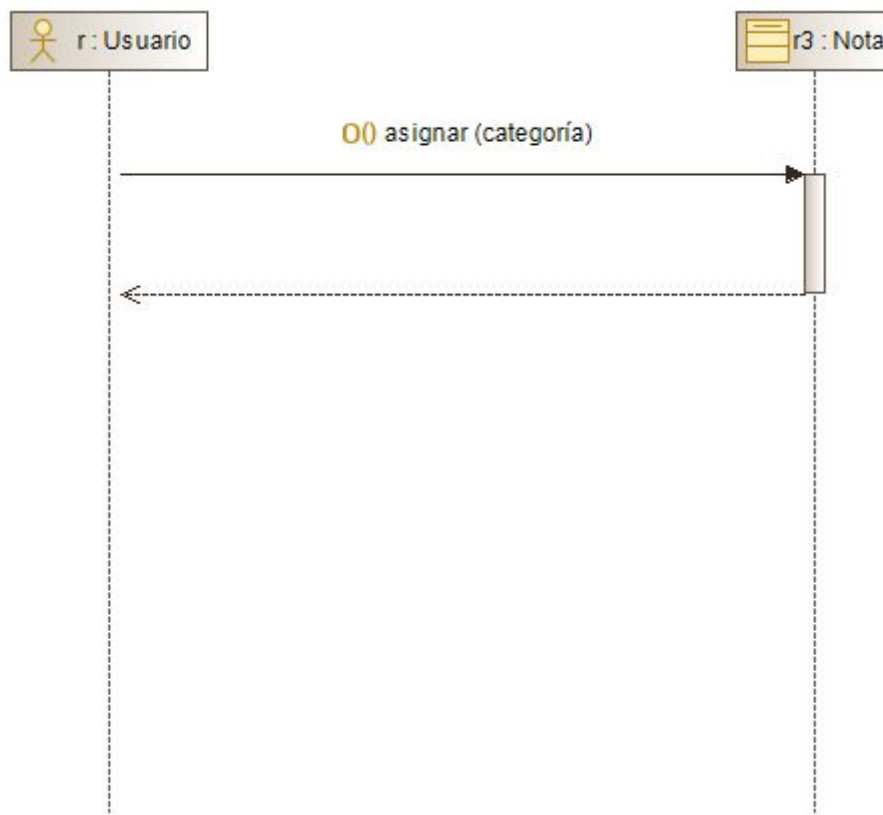
Modificar categoría



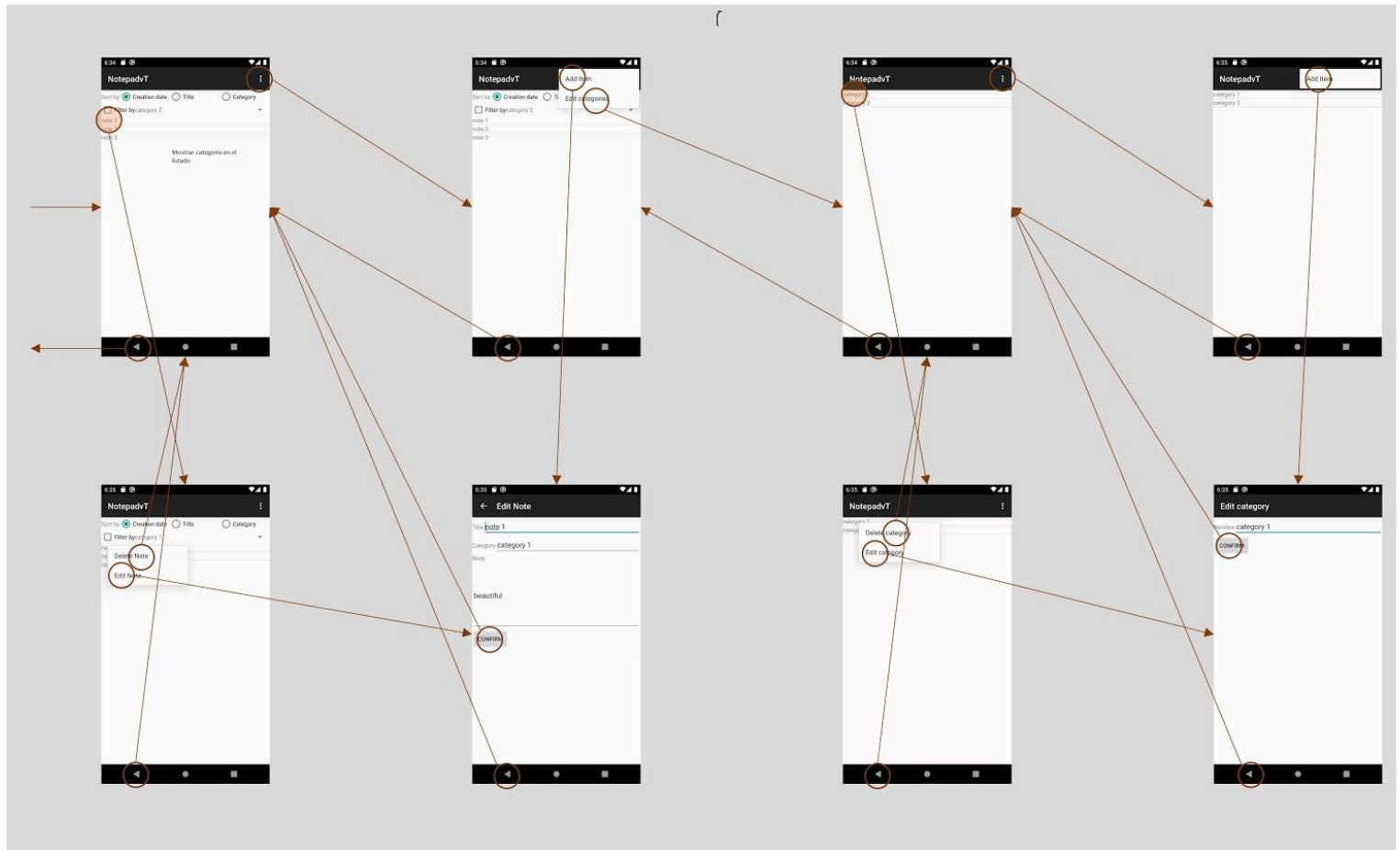
Eliminar categoría



Asignar categoría a nota ya existente

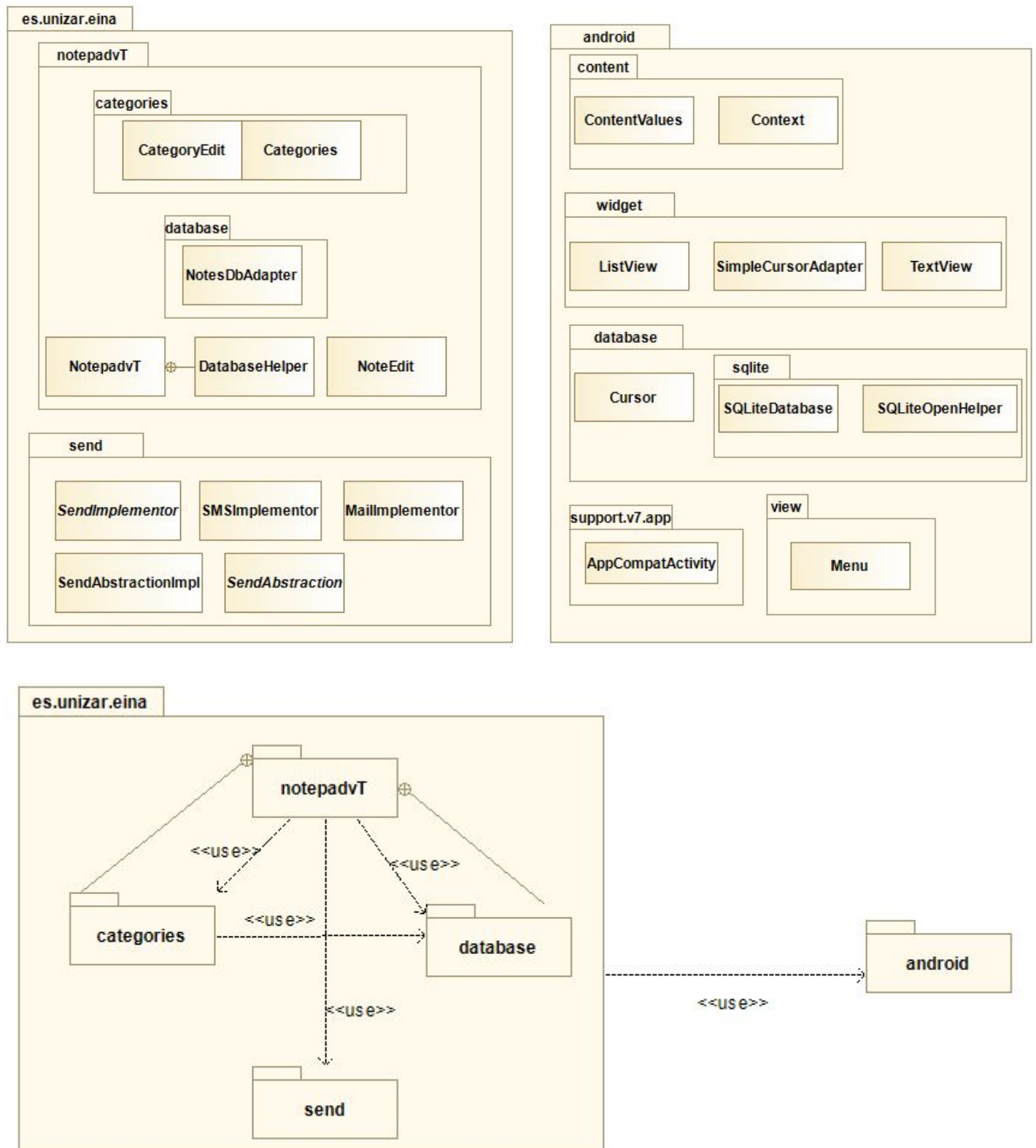


4.3-Mapa de navegación

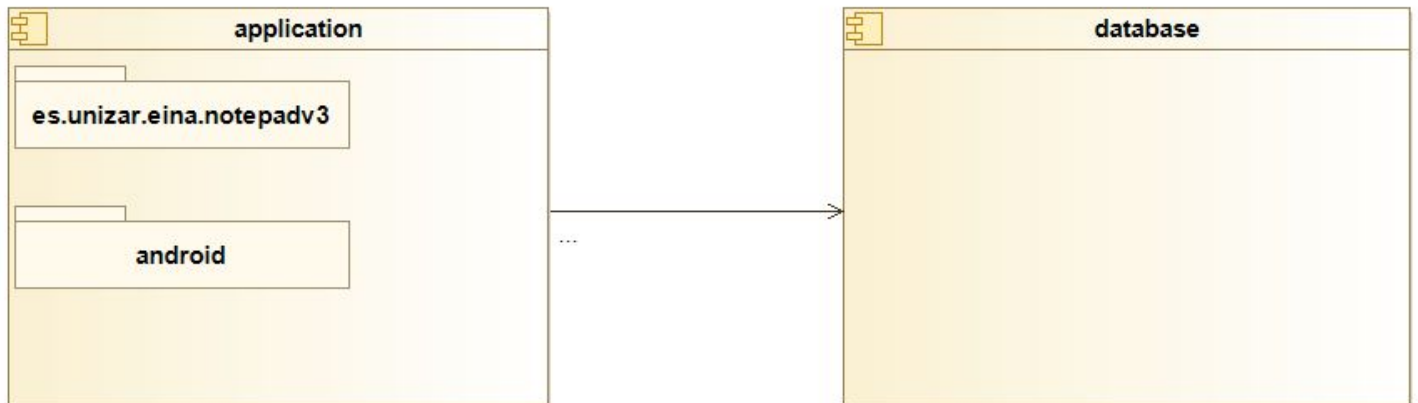


5.-Diagramas de fase de diseño del sistema

5.1-Diagrama de paquetes



5.2-Diagrama de componentes



5.3-Diagrama de despliegue

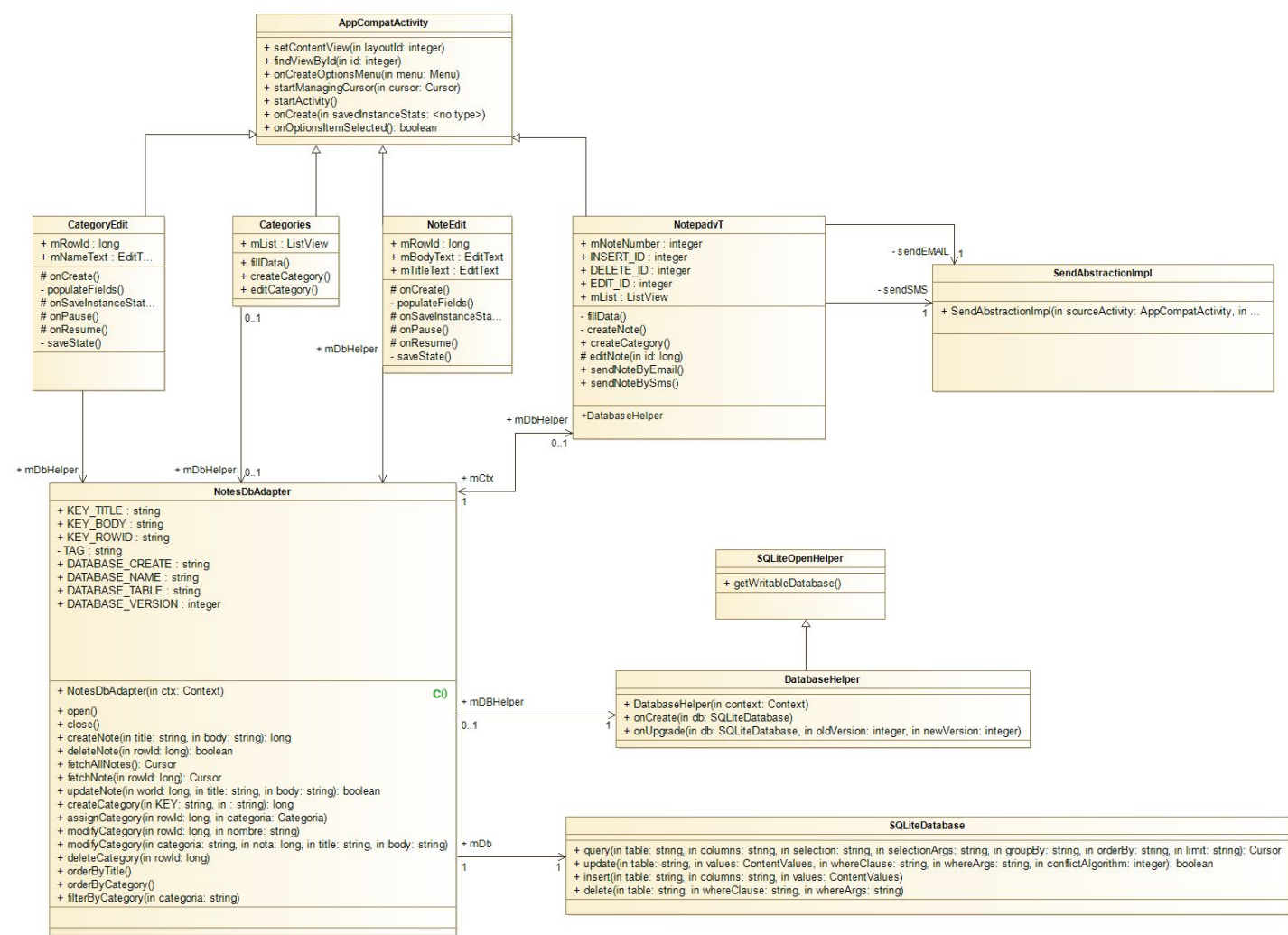


6.-Diagramas fase de diseño de objetos

6.1-Selección patrón de diseño

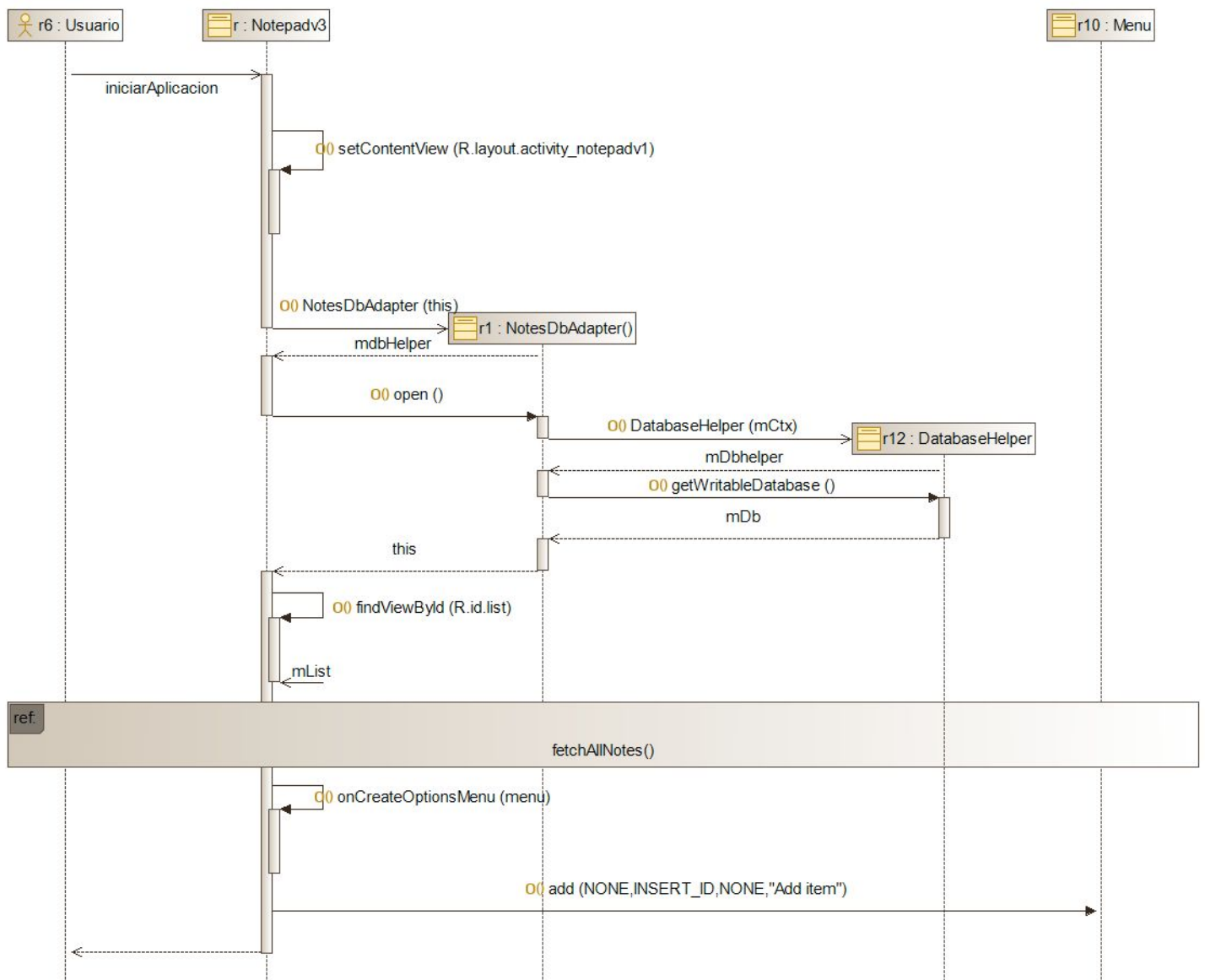
Siguiendo las recomendaciones del guión de prácticas se ha aplicado el patrón bridge, que nos permite separar las peticiones de SMS y EMAIL mediante una estructura de herencia y clases abstractas.

6.2-Diagrama de clases

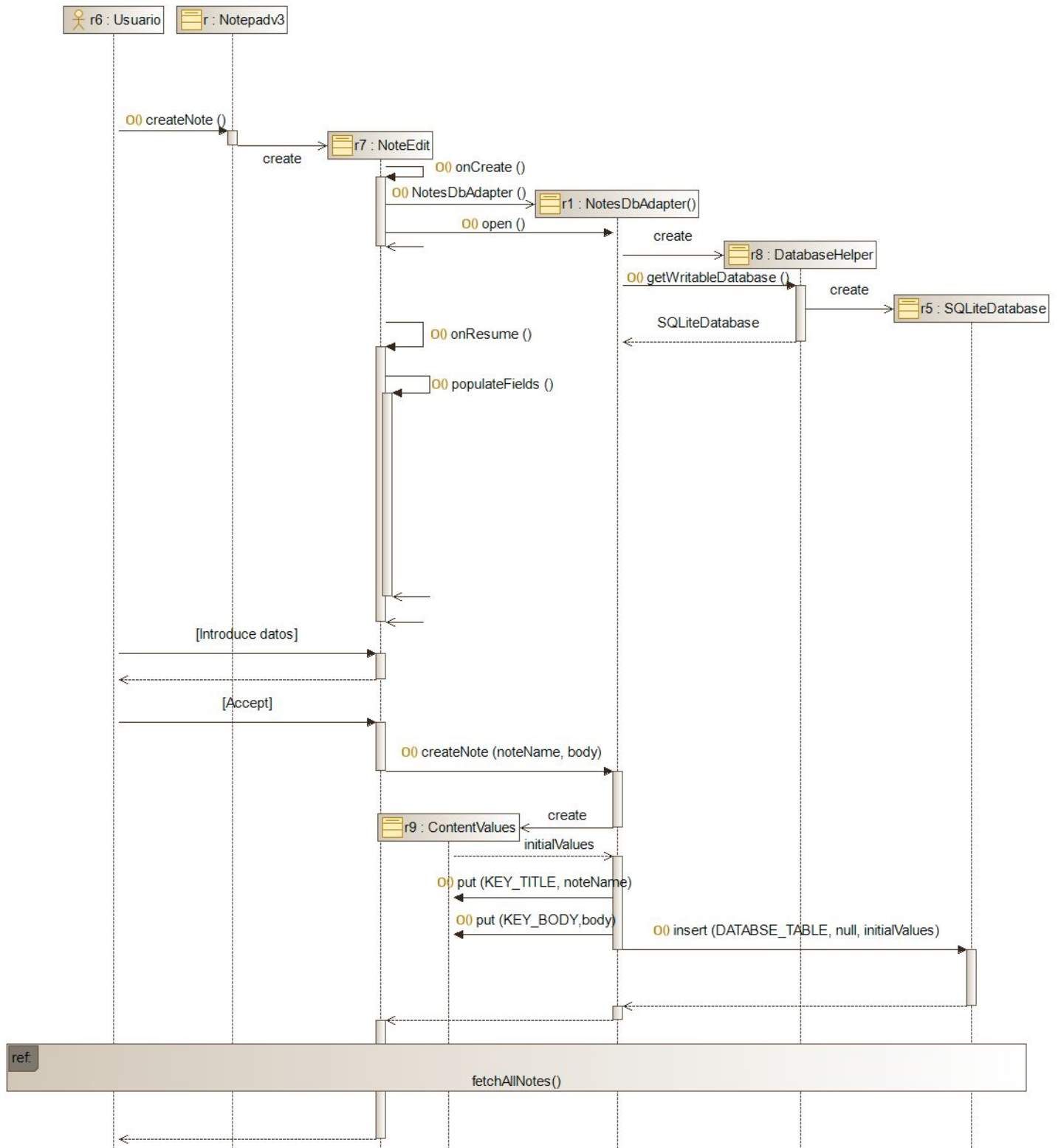


6.3-Diagramas de secuencia

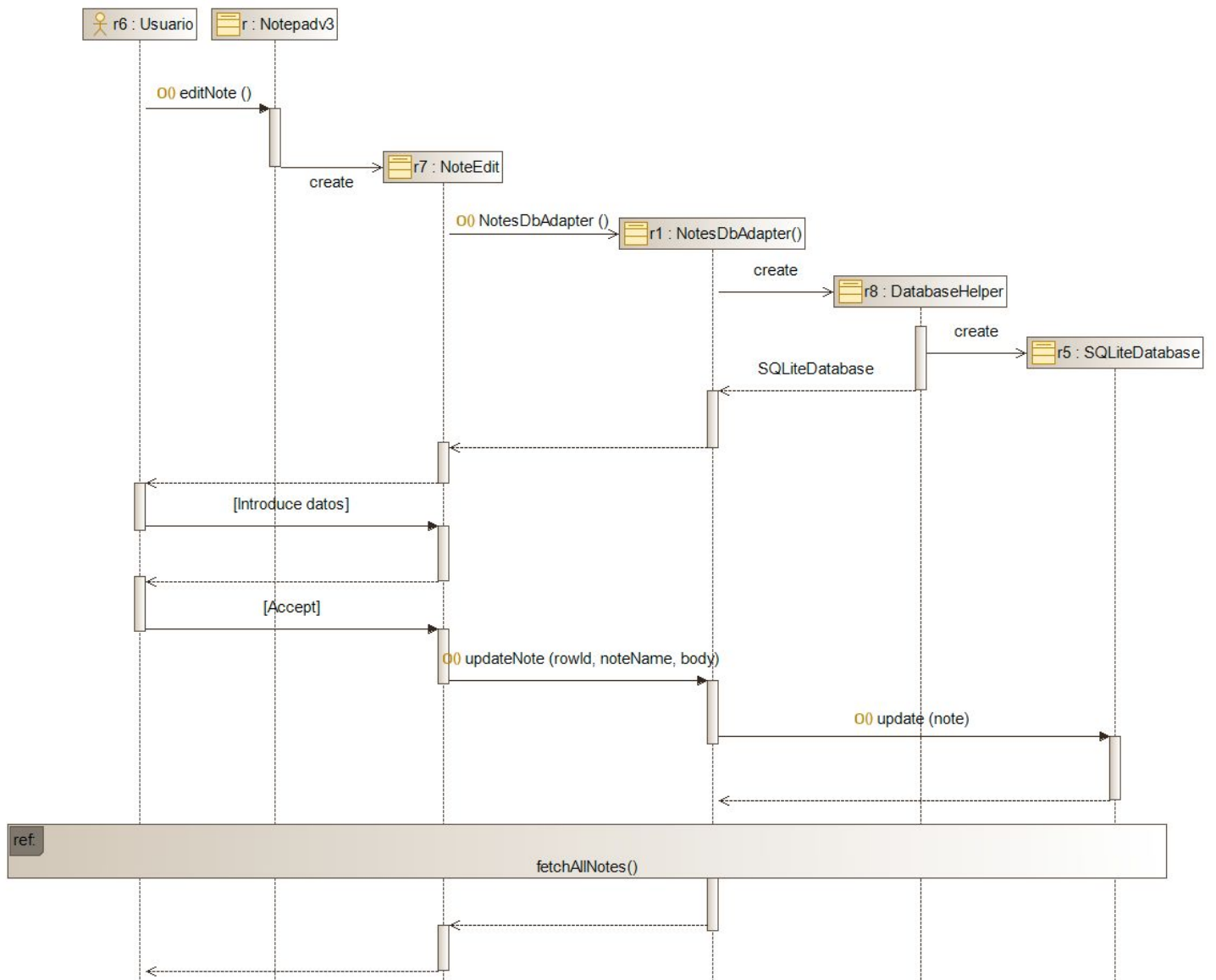
Inicialización de la aplicación



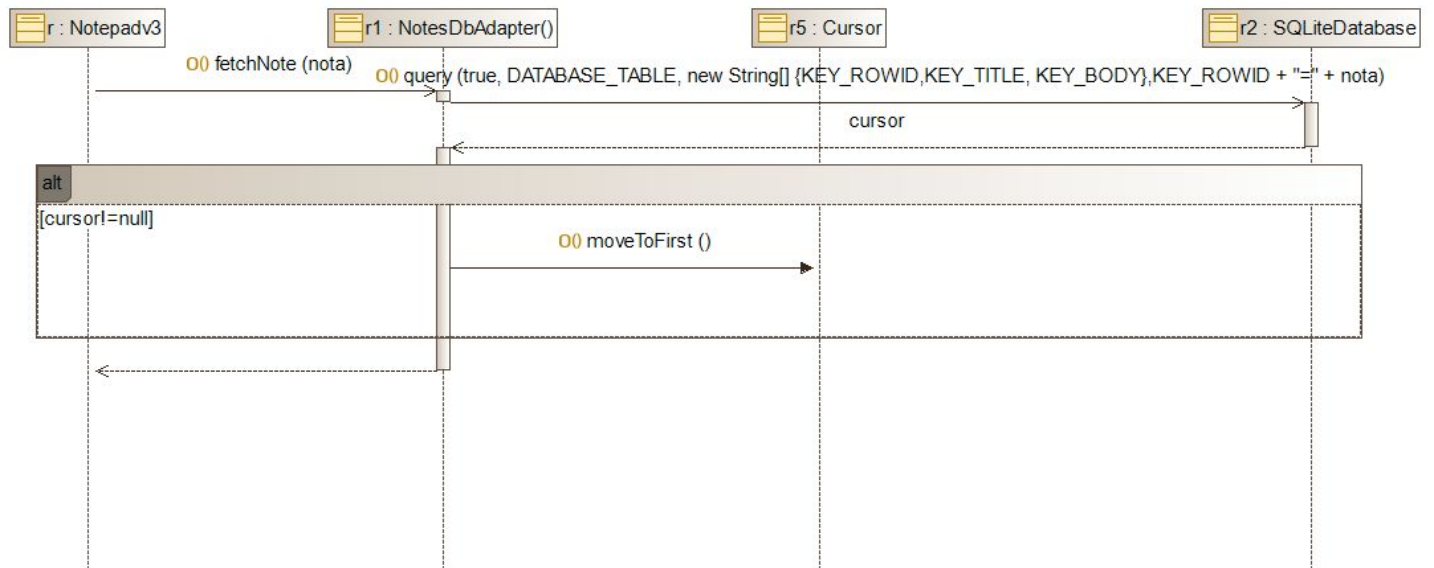
Crear nota



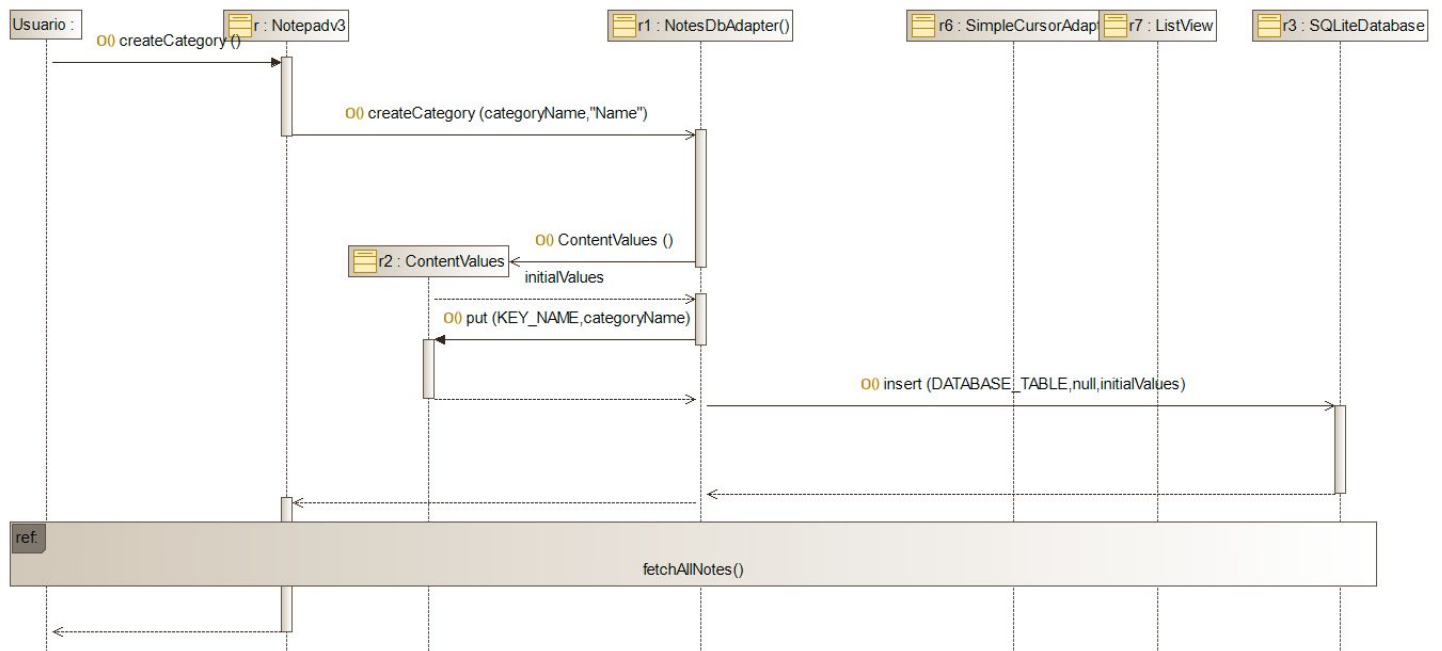
Actualizar nota



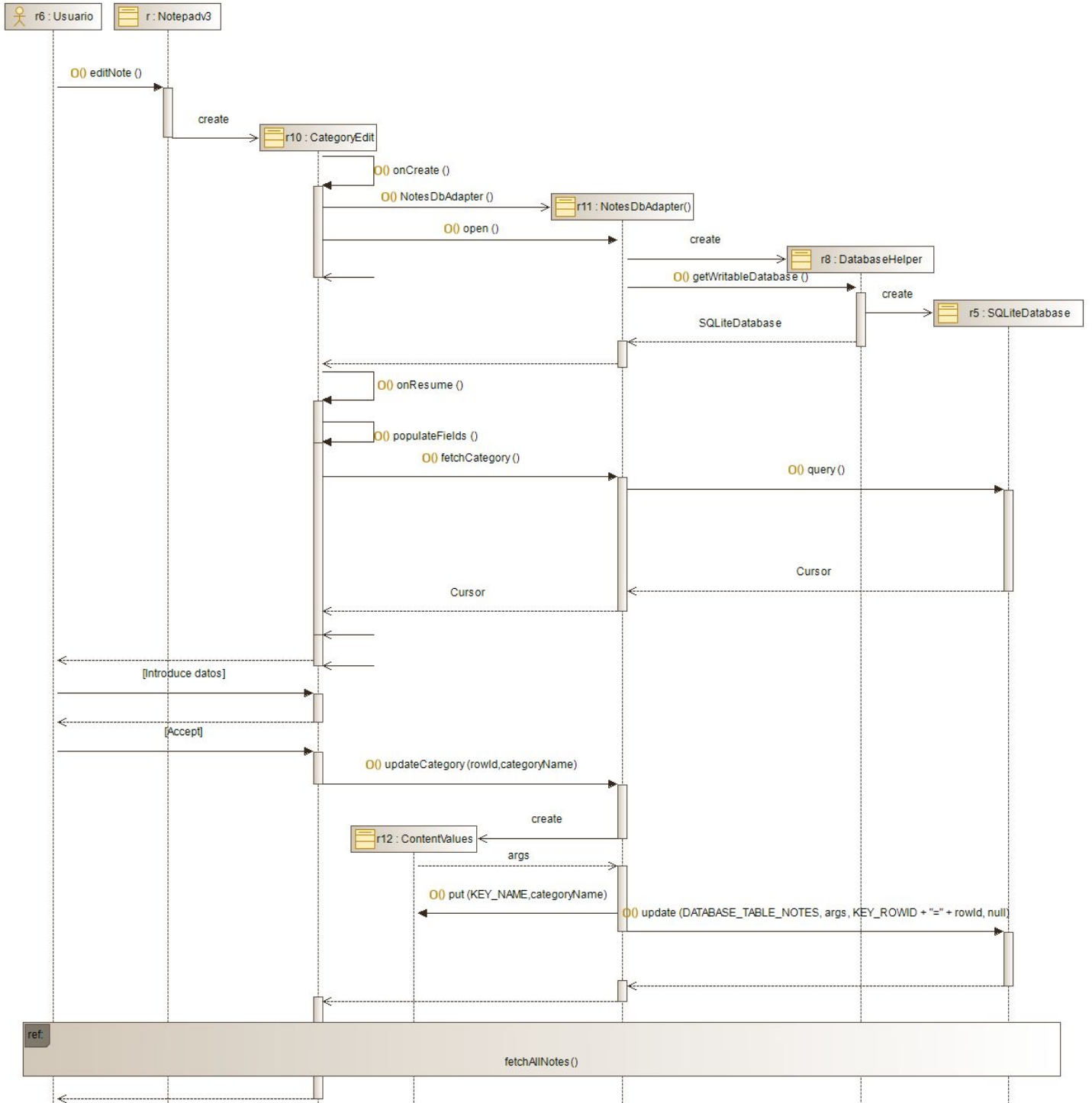
Consultar lista de notas ya existentes



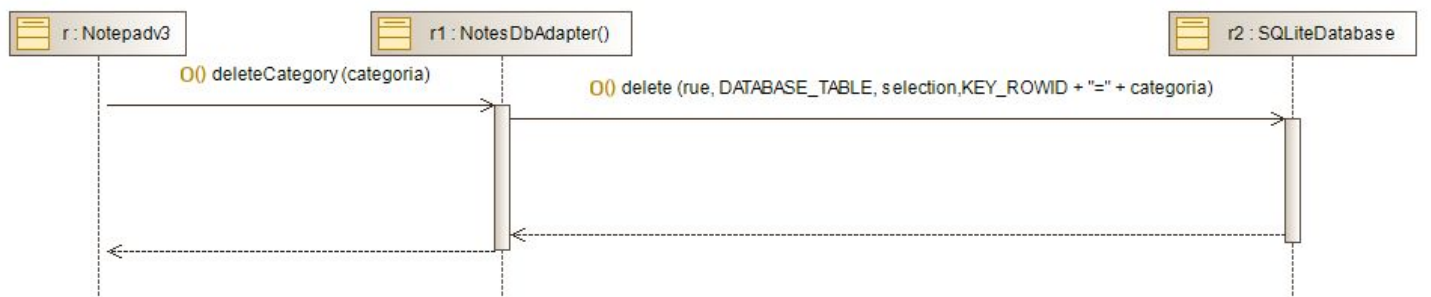
Crear categoría



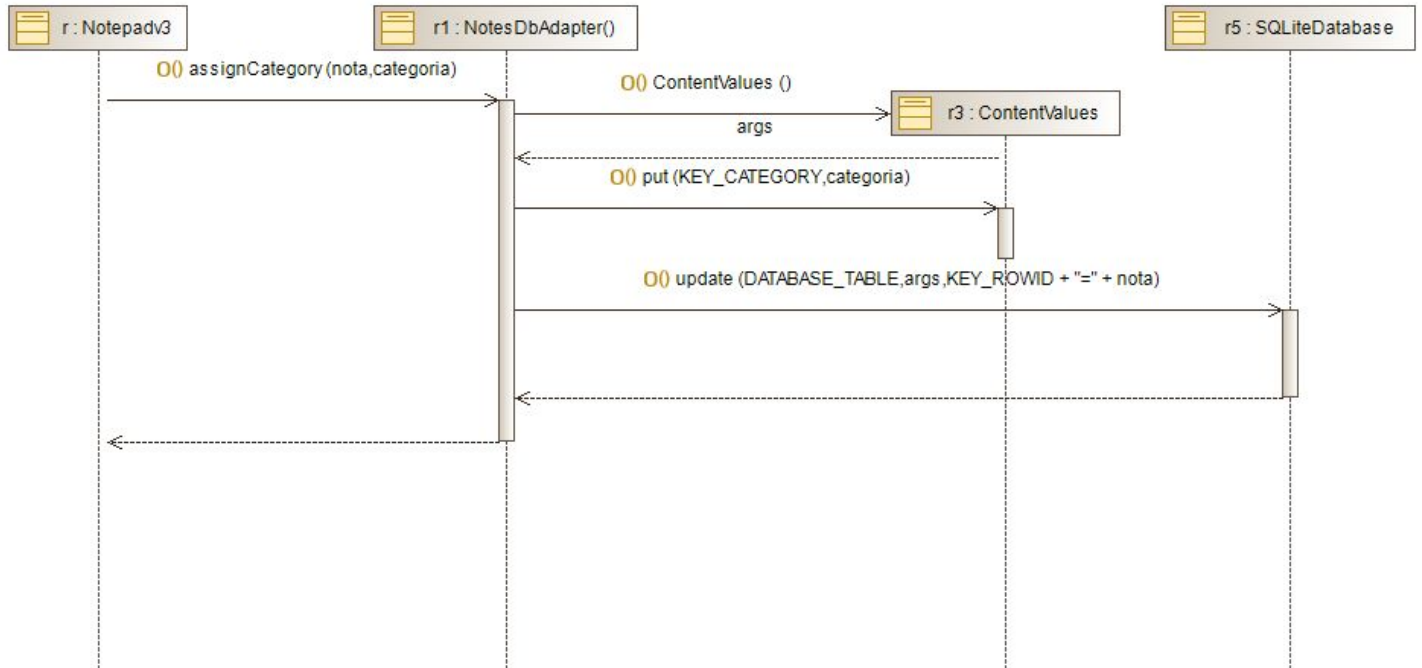
Modificar categoría



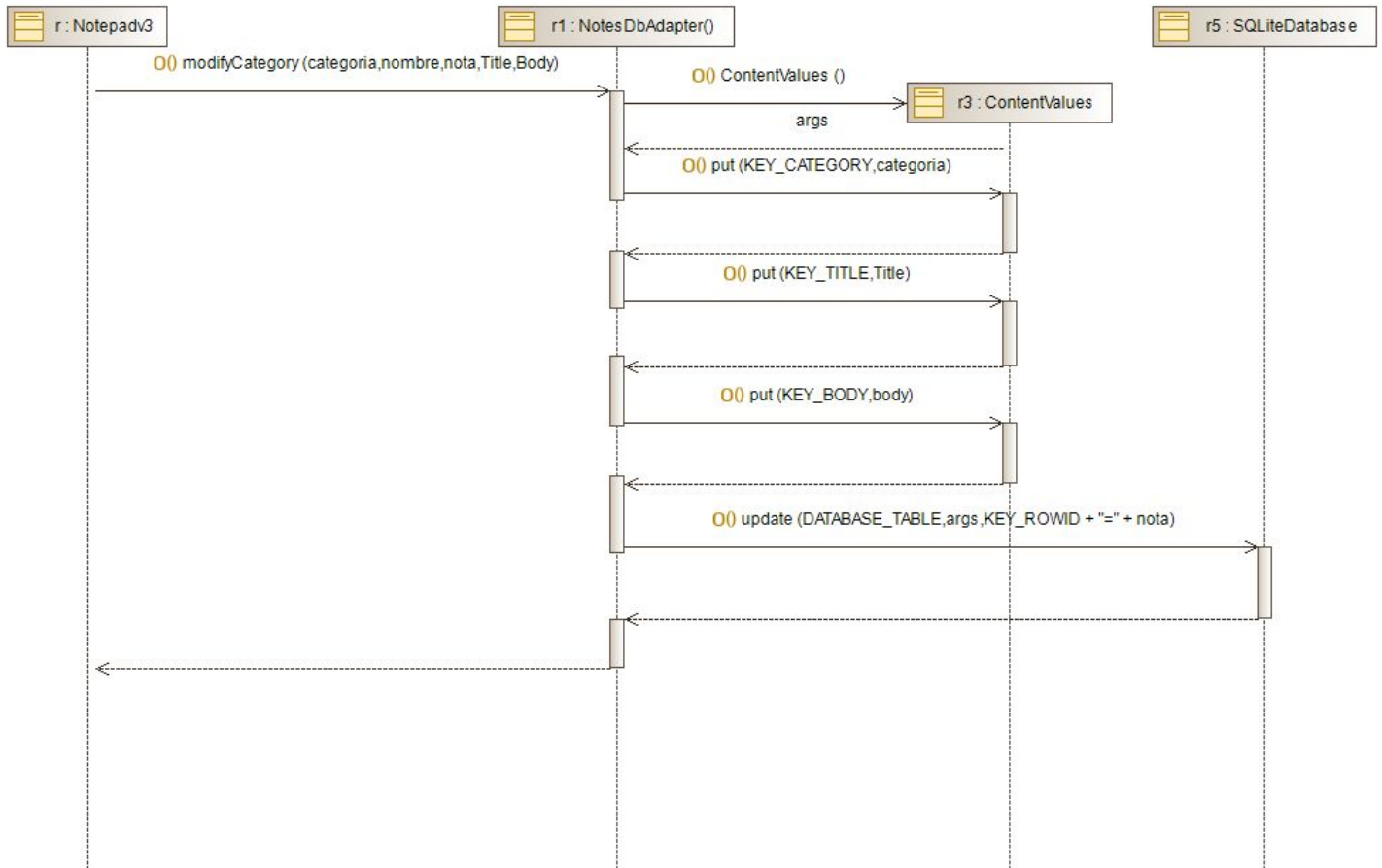
Eliminar categoría



Asignar categoría a nota ya existente



Modificar categoría asignada a nota existente



Ordenar lista de notas por categoría



Ordenar lista de notas por título



Filtrar lista de notas por categoría

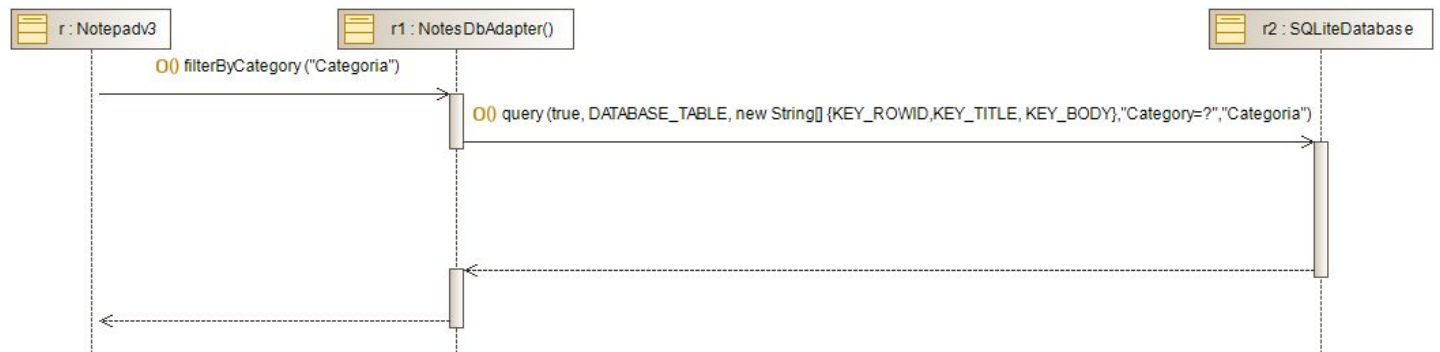
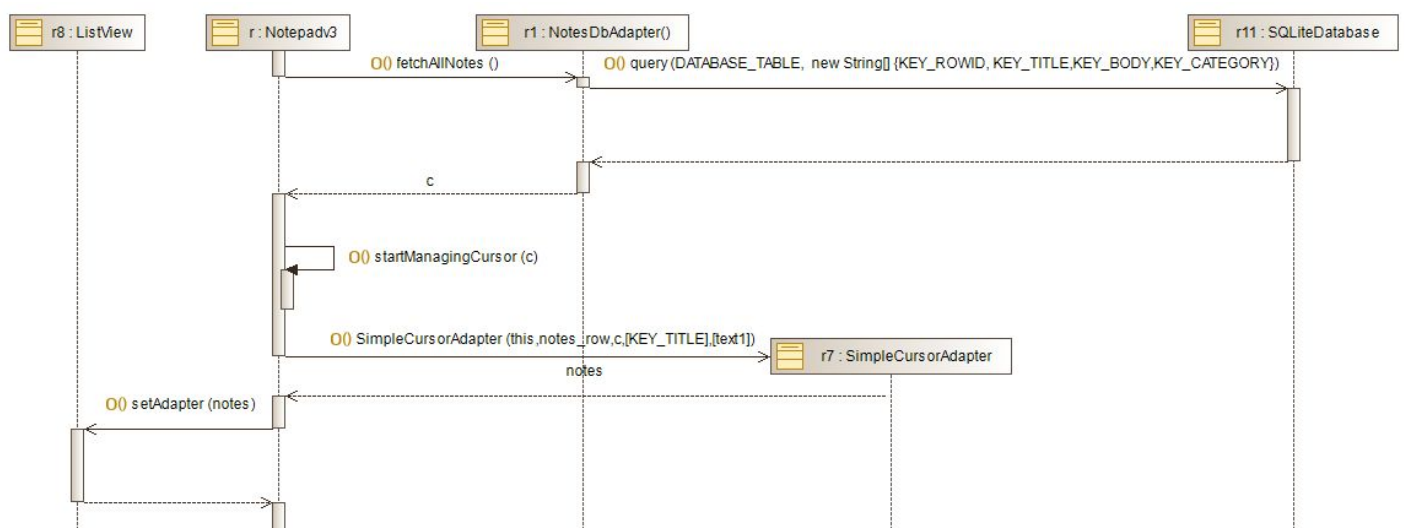
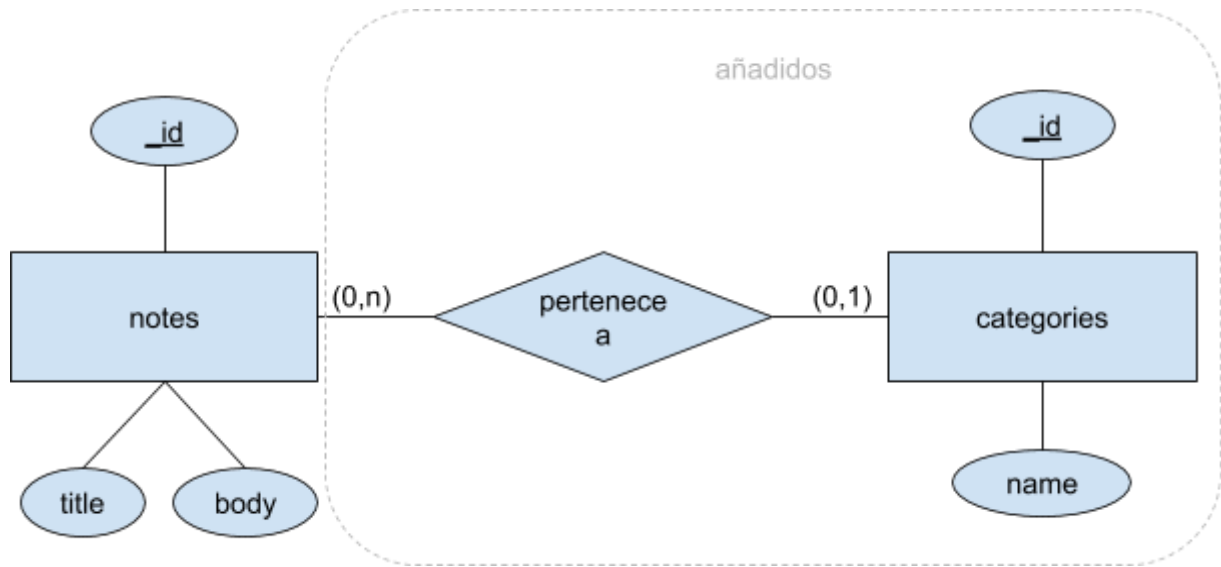


Diagrama de fetchAllNotes

Aparece referenciado en varios diagramas de secuencia.



6.4-Modelo entidad-relación



7.-Pruebas

7.1-Pruebas de caja negra

-Funcion public long createNote(String title, String body, Integer category)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>title</u>	<u>1: title.length()>0</u> <u>2: title!=null</u>	<u>3: title.length()<=0</u> <u>4: title==null</u>
<u>body</u>	<u>5: title!=null</u>	<u>6: title==null</u>
<u>category</u>	<u>7: category>=0</u> <u>8: category=null</u>	<u>9: category<0</u>

<u>Entrada</u>			<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>title</u>	<u>body</u>	<u>category</u>		
Casos de prueba para clases de equivalencia válidas (n° mínimo de casos de prueba)				
<u>"Hola"</u>	<u>"Hola"</u>	<u>1</u>	<u>1,2,5,7</u>	<u>rowId</u>
<u>"Hola"</u>	<u>"Hola"</u>	<u>null</u>	<u>1,2,5,8</u>	<u>rowId</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)				
<u>null</u>	<u>"Hola"</u>	<u>1</u>	<u>4</u>	<u>-1</u>
<u>"Hola"</u>	<u>null</u>	<u>1</u>	<u>6</u>	<u>-1</u>
<u>" "</u>	<u>"Hola"</u>	<u>1</u>	<u>3</u>	<u>-1</u>
<u>"Hola"</u>	<u>"Hola"</u>	<u>-1</u>	<u>9</u>	<u>-1</u>

-Funcion public boolean deleteNote(long rowId)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>rowId</u>	<u>1: rowId>=0</u>	<u>2: rowId<0</u>

<u>Entrada</u>	<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>rowId</u>		
Casos de prueba para clases de equivalencia válidas (n° mínimo de casos de prueba)		
<u>2</u>	<u>1</u>	<u>true</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)		
<u>-1</u>	<u>2</u>	<u>false</u>

-Funcion public boolean updateNote(long rowId, String title, String body, Integer category)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>title</u>	<u>1: title.length()>0</u> <u>2: title!=null</u>	<u>3: title.length()<=0</u> <u>4: title==null</u>
<u>body</u>	<u>5: title!=null</u>	<u>6: title==null</u>
<u>rowId</u>	<u>7: rowId>=0</u>	<u>8: rowId<0</u>
<u>category</u>	<u>9: category>=0</u> <u>10: category=null</u>	<u>11: category<0</u>

<u>Entrada</u>				<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>title</u>	<u>body</u>	<u>rowId</u>	<u>category</u>		
Casos de prueba para clases de equivalencia válidas (n° mínimo de casos de prueba)					
<u>"Hola"</u>	<u>"Hola"</u>	<u>1</u>	<u>1</u>	<u>1,2,5,7,9</u>	<u>true</u>
<u>"Hola"</u>	<u>"Hola"</u>	<u>1</u>	<u>null</u>	<u>1,2,5,7,10</u>	<u>true</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)					
<u>null</u>	<u>"Hola"</u>	<u>1</u>	<u>1</u>	<u>4</u>	<u>false</u>
<u>"Hola"</u>	<u>null</u>	<u>1</u>	<u>1</u>	<u>6</u>	<u>false</u>
<u>" "</u>	<u>"Hola"</u>	<u>1</u>	<u>1</u>	<u>3</u>	<u>false</u>
<u>"Hola"</u>	<u>"Hola"</u>	<u>-1</u>	<u>1</u>	<u>3</u>	<u>false</u>
<u>"Hola"</u>	<u>"Hola"</u>	<u>1</u>	<u>-1</u>	<u>11</u>	<u>false</u>

-Funcion public Cursor fetchAllNotes(String orderBy, String category)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>orderBy</u>	<u>1: orderBy=KEY_TITLE</u> <u>2: orderBy=KEY_CATEGORY</u> <u>3: orderBy=null</u>	<u>4: orderBy="ERROR"</u>
<u>category</u>	<u>5: category.length()>0</u> <u>6: category.length()==0</u> <u>7: category=null</u>	

<u>Entrada</u>		<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>orderBy</u>	<u>Category</u>		

Casos de prueba para clases de equivalencia válidas (n° mínimo de casos de prueba)			
<u>KEY_TITLE</u>	<u>"categoría"</u>	<u>1,5</u>	<u>Cursor</u>
<u>KEY_CATEGORY</u>	<u>" "</u>	<u>2,6</u>	<u>Cursor</u>
<u>null</u>	<u>null</u>	<u>3,7</u>	<u>Cursor</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)			
<u>"ERROR"</u>	<u>null</u>	<u>4</u>	<u>null</u>

-Funcion public Cursor fetchNote(long rowId)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>rowId</u>	<u>1: rowId>=0</u>	<u>2: rowId<0</u>

<u>Entrada</u>	<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>rowId</u>		
Casos de prueba para clases de equivalencia válidas (nº mínimo de casos de prueba)		
<u>1</u>	<u>1</u>	<u>Cursor</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)		
<u>-1</u>	<u>2</u>	<u>null</u>

-Funcion public long createCategory(String name)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>Name</u>	<u>1: name.length()>0</u> <u>2: name!=null</u>	<u>3: name.length()<=0</u> <u>4: name==null</u>

<u>Entrada</u>	<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>Name</u>		
Casos de prueba para clases de equivalencia válidas (n° mínimo de casos de prueba)		
<u>"Hola"</u>	<u>1,2</u>	<u>rowId</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)		
<u>null</u>	<u>4</u>	<u>-1</u>
<u>""</u>	<u>3</u>	<u>-1</u>

-Funcion public boolean deleteCategory(long rowId)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>rowId</u>	<u>1: rowId>=0</u>	<u>2: rowId<0</u>

<u>Entrada</u>	<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>Cuerpo</u>		
Casos de prueba para clases de equivalencia válidas (nº mínimo de casos de prueba)		
<u>2</u>	<u>1</u>	<u>true</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)		
<u>-1</u>	<u>2</u>	<u>false</u>

-Funcion public Cursor fetchCategory(long rowId)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>rowId</u>	<u>1: rowId>=0</u>	<u>2: rowId<0</u>

<u>Entrada</u>	<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>rowId</u>		
Casos de prueba para clases de equivalencia válidas (n° mínimo de casos de prueba)		
<u>1</u>	<u>1</u>	<u>Cursor</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)		
<u>-1</u>	<u>2</u>	<u>null</u>

-Funcion public Integer fetchCategory(string name)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>Name</u>	<u>1: name.length()>0</u> <u>2: name!=null</u>	<u>3: name.length()<=0</u> <u>4: name==null</u>

<u>Entrada</u>	<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>Name</u>		
Casos de prueba para clases de equivalencia válidas (nº mínimo de casos de prueba)		
<u>"Hola"</u>	<u>1,2</u>	<u>rowId</u>
Casos de prueba para clases de equivalencia no válidas (uno por clase)		

<u>null</u>	<u>4</u>	<u>null</u>
<u>""</u>	<u>3</u>	<u>null</u>

-Funcion public boolean updateCategory(long rowId, String name)

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>Name</u>	<u>1: title.length()>0</u> <u>2: title!=null</u>	<u>3: title.length()<=0</u> <u>4: title==null</u>
<u>rowId</u>	<u>5: rowId>=0</u>	<u>6: rowId<0</u>

<u>Entrada</u>		<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>Name</u>	<u>rowId</u>		
Casos de prueba para clases de equivalencia válidas (n° mínimo de casos de prueba)			
<u>"Hola"</u>	<u>1</u>	<u>1,2,5</u>	<u>true</u>

Casos de prueba para clases de equivalencia no válidas (uno por clase)			
<u>null</u>	<u>1</u>	<u>4</u>	<u>false</u>
<u>"Hola"</u>	<u>-1</u>	<u>6</u>	<u>false</u>
<u>""</u>	<u>1</u>	<u>3</u>	<u>false</u>

7.2-Prueba de volumen

Para las pruebas de volumen se añadió un requisito no funcional, “El sistema no va a almacenar más de 1000 notas”, que se ha comprobado mediante otro test como los realizados en la prueba de caja negra del apartado anterior.

<u>Entrada</u>	<u>Clases de equivalencia válidas</u>	<u>Clases de equivalencia no válidas</u>
<u>n (nº notas a crear)</u>	<u>1: n<1000</u>	<u>2: n>=1000</u>

<u>Entrada</u>	<u>Objetivo: clases cubiertas</u>	<u>Resultado</u>
<u>n</u>		
Casos de prueba para clases de equivalencia válidas (nº mínimo de casos de prueba)		
<u>500</u>	<u>1</u>	<u>operación realizada</u>

Casos de prueba para clases de equivalencia no válidas (uno por clase)		
<u>1000</u>	<u>2</u>	<u>operación realizada</u>

Como se puede ver en la prueba de caja negra de arriba, nuestra aplicación puede crear más de 1000 notas, no está limitado por ningún número más que por la capacidad del dispositivo. Sin embargo esto no se considera un fallo pues el requisito no es una obligación.

7.3- Análisis de los resultados

¿Funcionan los casos de prueba correspondientes a particiones de equivalencia válidas?

Los casos de prueba de las particiones de equivalencias válidas sí funcionan, para los casos de pruebas que cubren las clases válidas devuelven el resultado esperado, todos pasan los tests sin problemas.

¿Y los de particiones de equivalencia no válidas?

Los casos de prueba de las particiones de equivalencia no válidas fallan en algunos casos (cuando el campo title y body son null o cuando el title es la cadena vacía ""). Se ha modificado el código para corregir estos fallos.

¿Se ha producido algún error? ¿Ha aparecido algún problema de usabilidad?

Tras la primera ejecución de los test fallaban aquellos de las clases de equivalencia no válidas, que se corrigieron posteriormente. Tras la ejecución de los test de usabilidad y por indicación del enunciado se ha detectado que desplazarse en la lista de notas es tedioso, por lo que se ha implementado la característica que hace que la lista se auto desplace a la nota editada/creada/borrada. Todos los errores se han corregido y las pruebas son satisfactorias.

7.4-Pruebas de sobrecarga (monkey)

Resultados de aplicar monkey con 50000 eventos `adb shell monkey -p es.unizar.eina.notepadvT -v -s 1573333539477 50000`



Monkey finaliza sin errores y devuelve por pantalla:

Events injected: 50000

:Sending rotation degree=0, persist=false

:Dropped: keys=43 pointers=36 trackballs=0 flips=0 rotations=0

Network stats: elapsed time=228894ms (0ms mobile, 0ms wifi, 228894ms not connected)

// Monkey finished

Por lo tanto podemos suponer que nuestra aplicación pasa la prueba de sobrecarga sin problemas, ya que con 50000 eventos realizados por monkey no sucede ningún error ni se cierra la aplicación, como podemos observar en las imágenes toda categoría y nota es creada correctamente.

7.5- Descripción pruebas

Se han desarrollado dos tipos de pruebas (tres contando la ejecución de monkey).

- 1) Pruebas de base de datos: realizadas utilizando el framework junit disponible de forma nativa en android studio, accediendo directamente a la clase DBAdapter sin necesidad de iniciar la aplicación (junit genera un context equivalente, pero la interfaz no se llega a mostrar). Con esto se han probado los tests de caja negra. El código de aplicación no ha sido modificado para la realización de las pruebas, y por ello también se han incluido utilidades para test manuales que permiten crear muchas notas y borrar todas las notas.
- 2) Pruebas de interfaz: realizadas utilizando el framework espresso disponible de forma nativa en android studio. Estas pruebas han simulado los casos de uso de creación/borrado/modificación de notas, para comprobar que visualmente la aplicación se comporta adecuadamente.

8.-Resultados y conclusiones.

Siguiendo las indicaciones de los profesores se ha logrado desarrollar en Android Studio el código de las funcionalidades extras requeridas, y comprobando mediante pruebas que éste funcionaba correctamente. El manejo de este programa ha sido sencillo y se valora positivamente la facilidad y cantidad de atajos que proporciona, siendo una herramienta adecuada para este tipo de proyectos. Además tiene integración directa con git lo que facilita el trabajo en equipo.

También se ha logrado realizar los diagramas que se requerían mediante el software Modelio. Sin embargo a diferencia del anterior el manejo de este software no ha estado exento de dificultades e inconvenientes, debido a las trabas que proporciona y a la falta de funcionalidades. Además al no tener integración con git y tener un formato de ficheros propietario trabajar en equipo supone asegurarse de que solo una persona trabaja en el proyecto simultáneamente, e ir enviando o sincronizando la carpeta con cuidado de que ningún fichero fuera borrado en el proceso (lo que ha sucedido en alguna ocasión). Se ha valorado negativamente el uso de este programa, pues además no se ha logrado ver el verdadero potencial de la realización de diagramas de este tipo (sobre todo de análisis) ya que su uso en ambientes reales (no académicos) queda relegado como explicación, documentación de software existente o incluso para bocetos personales; pero no como finalidad principal.

De todas formas se entiende que al ser académico el objetivo sea tomar experiencia en el diseño e implementación de diagramas de las distintas fases del proceso. Y si bien esto sí se consigue, en lugar de animar a seguir utilizando diagramas (recomendables y casi obligatorios en los puntos mencionados previamente) su efecto es el contrario.

9.-Bibliografía

<https://www.modelio.org/quick-start-pages-35.html>

<https://stackoverflow.com/>

<https://developer.android.com/studio/intro>

<https://github.com/>