

IES Francisco de los Ríos

Ciclo Formativo de Grado Superior

Desarrollo de Aplicaciones Multiplataforma

Proyecto Integrado

Lanword

Autor : Víctor Manuel Ortiz Guardado

Tutor : Rafael Luis Morales Márquez

Marzo, 2015

Tabla de contenidos

1.Descripción del problema.....	3
2.Objetivos del proyecto.....	3
3.Recursos necesarios.....	3
4.Planificación temporal.....	4
5.Diseño de la aplicación.....	5
5.1.Diseño de la arquitectura.....	5
5.1.1.Requisitos fundamentales.....	6
5.1.2.Desarrollo de requisitos fundamentales.....	7
5.1.2.1.Diagrama de casos de uso – Administración de palabras, grupos e idiomas.....	7
5.1.2.2.Diagrama de casos de uso – juego de traducir palabra.....	47
5.1.3.Diseño de componentes.....	52
5.1.4.Diseño de estructura de clases y paquetes.....	52
5.1.4.1.Diagrama de clases – modelo.....	54
5.1.4.2.Diagrama de clases – interfaces.bd.....	63
5.1.4.3.Diagrama de clases – controladores.administracion.....	70
5.1.4.4.Diagrama de clases – cotroladores.juego.....	75
5.2.Diseño de la base de datos.....	77
5.3.Codificación.....	79
5.3.1.Lenguaje de programación y IDE utilizado.....	79
5.3.2.SGDB utilizado.....	79
5.3.3.Estructura de ficheros del proyecto.....	80
6.Anexos.....	82
6.1.Sentencias SQL del DDL.....	82

1. Descripción del problema

Se quiere hacer un programa que ayude al usuario a aprender palabras de otros idiomas. El programa debe de permitir al usuario introducir palabras, agruparlas en grupos definidos por el usuario, agruparlas por idiomas, y con las palabras introducidas realizar juegos para recordarlas.

2. Objetivos del proyecto

El usuario debe poder:

- Añadir, modificar, eliminar y traducir palabras. El usuario deberá de especificar el idioma de las palabras.
- Agrupar palabras en grupos. Los grupos los utilizará un juego para usar las palabras. Por ejemplo, si el usuario quisiera examinarse de un subconjunto de palabras de un idioma solamente, y no de todo el idioma.
- Jugar a un juego para recordar palabras. Este juego consistirá en mostrar palabras y el usuario tendrá que escribir una traducción. Al final del juego mostrará la puntuación y se podrá iniciar de nuevo.

3. Recursos necesarios

Los requisitos hardware mínimos para que el proyecto funcione son:

- Procesador Prentium 2 a 266Mhz.
- Espacio en el disco 200MB.
- RAM 128 MB.
- Sistema operativo de 64 bits.

Los recursos software que necesitará el proyecto para ser ejecutado:

- Java JRE 8.

Los recursos sotfware que utilizará el proyecto son:

- SQLite, para la base de datos.
- Java JDK 8.

4. Planificación temporal

La planificación que he estimado es la siguiente:

Semana	Acciones previstas
1-2	Análisis: <ul style="list-style-type: none">• Captación de requisitos fundamentales.• Diseño de los componentes de la aplicación.• Diseño de clases inicial.
3-5	Diseño: <ul style="list-style-type: none">• Desarrollo de los requisitos fundamentales.• Diseño de la base de datos.
6-7	Codificación: <ul style="list-style-type: none">• Codificación de módulos independientes.• Integración de los módulos entre sí.
8-10	Codificación: <ul style="list-style-type: none">• Diseño y codificación de la interfaz gráfica.
11-12	Documentación: <ul style="list-style-type: none">• Documentación del manual de instalación.• Documentación del manual de usuario.• Valoración personal, mejoras y posibles actualizaciones.

5. Diseño de la aplicación

Este apartado se dividirá en dos; diseño de la arquitectura, diseño de datos y decisiones del diseño.

En el diseño de la arquitectura definiré, todos los modelos conceptuales para el desarrollo de la aplicación. Este componente se ha diseñado siguiendo el patrón de arquitectura del software MVC (Modelo-Vista-Controlador). Por lo tanto, en el diseño de la arquitectura es la parte modelo. En la codificación se desarrollarán la vista y el controlador, siguiendo las directrices del modelo, y los requisitos fundamentales (casos de uso).

En este proyecto voy a utilizar una base de datos relacional. Por lo tanto, en el diseño de datos se definirán el diagramas de ER (Entidad-Relación), junto con la especificación de los elementos y la lógica de su manipulación.

En el apartado decisiones del diseño podrás ver una lista enumerada de todas las decisiones que voy a tomar en el diseño de la aplicación.

5.1. Diseño de la arquitectura

El diseño de la arquitectura del proyecto, lo dividiré en los siguientes apartados:

- los requisitos fundamentales de la aplicación, representados por diagramas de casos de uso;
- el diseño de los componentes, representado por un diagrama de componentes;
- el diseño de paquetes;
- el diseño de las clases y sus relaciones, representado por diagramas de clases;
- el desarrollo de los requisitos fundamentales, mediante la especificación de los casos de uso y diagramas de secuencia.

5.1.1. Requisitos fundamentales

Siguiendo los objetivos proyecto, los requisitos para la parte de administración de los grupos, idiomas y palabras, se reflejan en el siguiente diagrama.

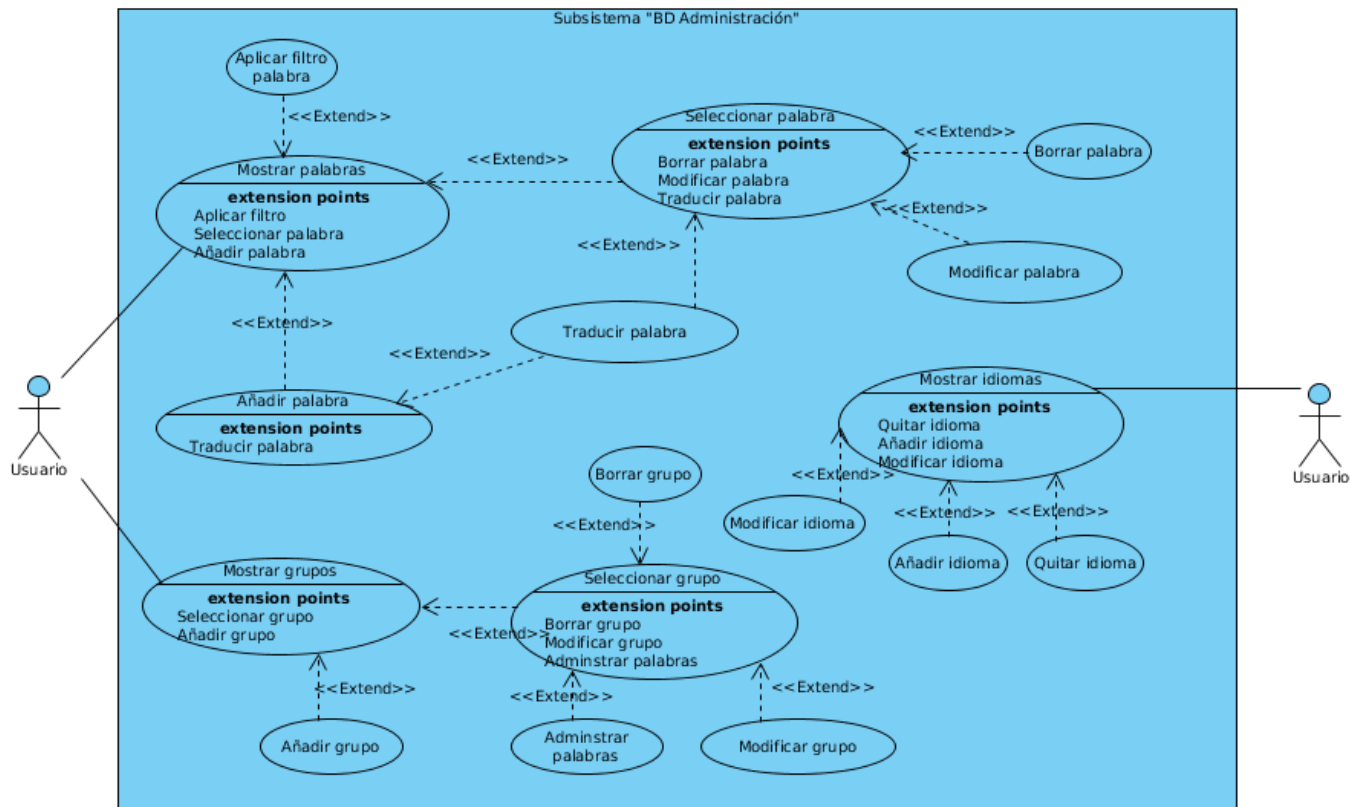


Diagrama 1: Requisitos de administración de grupos, palabras y idiomas.

Los requisitos fundamentales para el juego son los siguientes:

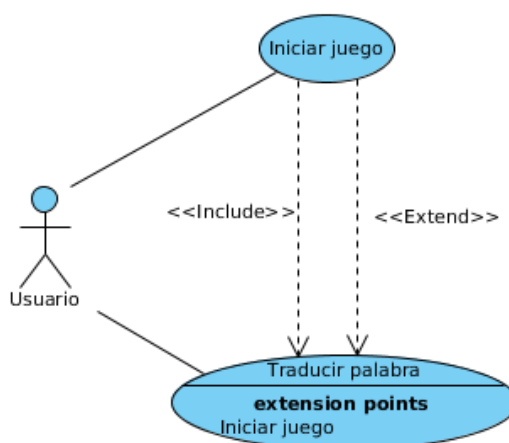


Diagrama 2: Requisitos del juego de traducir palabras

Como se pueden ver en los diagramas 1 y 2, el usuario final podrá, a grandes rasgos, administrar los componentes claves de la aplicación. Que son, administrar las palabras, grupos, idiomas y jugar al juego de traducir palabras. En cada bloque de administración podrá hacer lo siguiente:

- Ver las palabras existentes, añadir una nueva, filtrar las palabras para buscar alguna o seleccionar una palabra. Cuando el usuario selecciona una palabra, podrá borrarla, traducirla o modificarla.
- Ver los grupos existentes, añadir un nuevo grupo y seleccionar uno existente. Cuando se seleccione, el usuario podrá modificarlo, eliminarlo o administrar sus palabras.
- Ver los idiomas existentes, añadir uno nuevo, o seleccionar alguno para modificarlo, o eliminarlo.
- Jugar al juego de traducir palabras, en el que podrá elegir un grupo o un idioma, y un idioma de traducción. Seguidamente podrá introducir traducciones de las palabras que se le vayan mostrando, hasta que llegue a traducir 10 y se mostrará la puntuación.

5.1.2. Desarrollo de requisitos fundamentales.

En esta sección explicaré más detalladamente cada caso de uso, junto con su diagrama de secuencia. Puesto que he realizado este diseño antes de documentarlo (no lo he diseñado a la vez que lo documento), las clases a las que se hagan referencia a en los diagramas, las podrás ver en el apartado *5.1.4 Diseño de clases y paquetes*.

Empezaré por definir qué hace cada caso de uso del diagrama 1 (administración de palabras, grupos e idiomas), y después, los casos de uso del diagrama 2 (juego de traducir palabras).

5.1.2.1. Diagrama de casos de uso – Administración de palabras, grupos e idiomas.

Los caso de uso que hay en este diagrama son :

- Mostrar palabras.
- Aplicar filtro palabra.
- Añadir palabra.
- Seleccionar palabra.
- Traducir palabra.
- Modificar palabra.
- Borrar palabra.

- Mostrar grupos.
- Añadir grupo.
- Seleccionar grupo.
- Administrar palabras.
- Modificar grupo.
- Borrar grupo.
- Mostrar idiomas.
- Modificar idioma.
- Quitar idioma.

A continuación mostraré una tabla con la definición del cada caso de uso y su diagrama de secuencia.

Mostrar palabras																										
Descripción	El sistema muestra todas las palabras al usuario, dando la posibilidad de filtrarlas, de seleccionar alguna para más funcionalidad o de añadir más palabras. Cada una de estas opciones perteneces respectivamente a los casos de uso "Aplicar filtro palabra", "Seleccionar palabra" y "Añadir palabra".																									
Precondiciones	Ninguna.																									
Postcondiciones	<ul style="list-style-type: none"> • Opción "Aplicar filtro", habilitada. • Opción "Añadir palabra", habilitada. • Posibilidad de seleccionar palabra, habilitada. 																									
Flujo de eventos normal	<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>1</td><td>Selecciona la opción <i>mostrar palabras</i>.</td><td></td></tr> <tr> <td>2</td><td></td><td>Recupera la información del filtro de palabras, si lo hubiera.</td></tr> <tr> <td>3</td><td></td><td>Consulta la base de datos para obtener las palabras.</td></tr> <tr> <td>4</td><td></td><td>Muestra las palabras.</td></tr> <tr> <td>5</td><td></td><td>Habilita la opción de <i>Aplicar filtro</i>.</td></tr> <tr> <td>6</td><td></td><td>Habilita la opción de <i>Seleccionar una palabra</i>.</td></tr> <tr> <td>7</td><td></td><td>Habilita la opción de <i>Añadir una palabra</i> nueva.</td></tr> </table>			Usuario	Sistema	1	Selecciona la opción <i>mostrar palabras</i> .		2		Recupera la información del filtro de palabras, si lo hubiera.	3		Consulta la base de datos para obtener las palabras.	4		Muestra las palabras.	5		Habilita la opción de <i>Aplicar filtro</i> .	6		Habilita la opción de <i>Seleccionar una palabra</i> .	7		Habilita la opción de <i>Añadir una palabra</i> nueva.
	Usuario	Sistema																								
1	Selecciona la opción <i>mostrar palabras</i> .																									
2		Recupera la información del filtro de palabras, si lo hubiera.																								
3		Consulta la base de datos para obtener las palabras.																								
4		Muestra las palabras.																								
5		Habilita la opción de <i>Aplicar filtro</i> .																								
6		Habilita la opción de <i>Seleccionar una palabra</i> .																								
7		Habilita la opción de <i>Añadir una palabra</i> nueva.																								

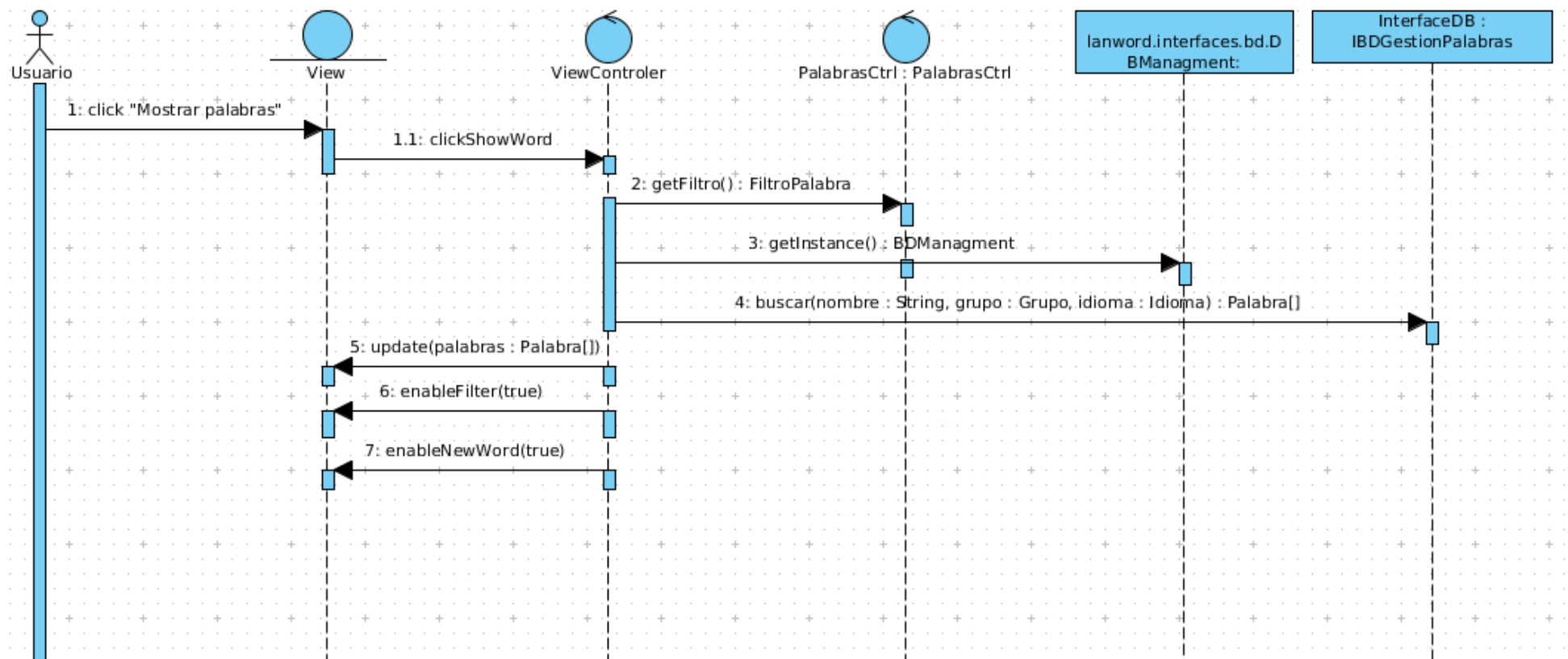


Diagrama de secuencia – escenario muestra palabras

Aplicar filtro palabra																													
Descripción	El sistema ofrece al usuario la posibilidad de filtrar la lista de palabras mostradas según su idioma, grupo o cadena que contenga su nombre.																												
Precondiciones	Ninguna.																												
Postcondiciones	<ul style="list-style-type: none"> Configuración de filtro establecida. 																												
Flujo de eventos normal		<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>1</td><td>El usuario selecciona <i>aplicar filtro</i>.</td><td></td></tr> <tr> <td>2</td><td></td><td>Habilita medios de entrada para que el usuario introduzca una cadena, seleccione un grupo y seleccione un idioma.</td></tr> <tr> <td>3</td><td>Introduce una cadena en el campo nombre.</td><td></td></tr> <tr> <td>4</td><td>Selecciona un grupo.</td><td></td></tr> <tr> <td>5</td><td>Selecciona un idioma.</td><td></td></tr> <tr> <td>6</td><td>Pulsa sobre <i>aplicar filtro</i></td><td></td></tr> <tr> <td>7</td><td></td><td>Guarda la configuración establecida por el usuario.</td></tr> <tr> <td>8</td><td></td><td>Instancia el caso de uso <i>mostrar palabras</i>.</td></tr> </table>		Usuario	Sistema	1	El usuario selecciona <i>aplicar filtro</i> .		2		Habilita medios de entrada para que el usuario introduzca una cadena, seleccione un grupo y seleccione un idioma.	3	Introduce una cadena en el campo nombre.		4	Selecciona un grupo.		5	Selecciona un idioma.		6	Pulsa sobre <i>aplicar filtro</i>		7		Guarda la configuración establecida por el usuario.	8		Instancia el caso de uso <i>mostrar palabras</i> .
	Usuario	Sistema																											
1	El usuario selecciona <i>aplicar filtro</i> .																												
2		Habilita medios de entrada para que el usuario introduzca una cadena, seleccione un grupo y seleccione un idioma.																											
3	Introduce una cadena en el campo nombre.																												
4	Selecciona un grupo.																												
5	Selecciona un idioma.																												
6	Pulsa sobre <i>aplicar filtro</i>																												
7		Guarda la configuración establecida por el usuario.																											
8		Instancia el caso de uso <i>mostrar palabras</i> .																											

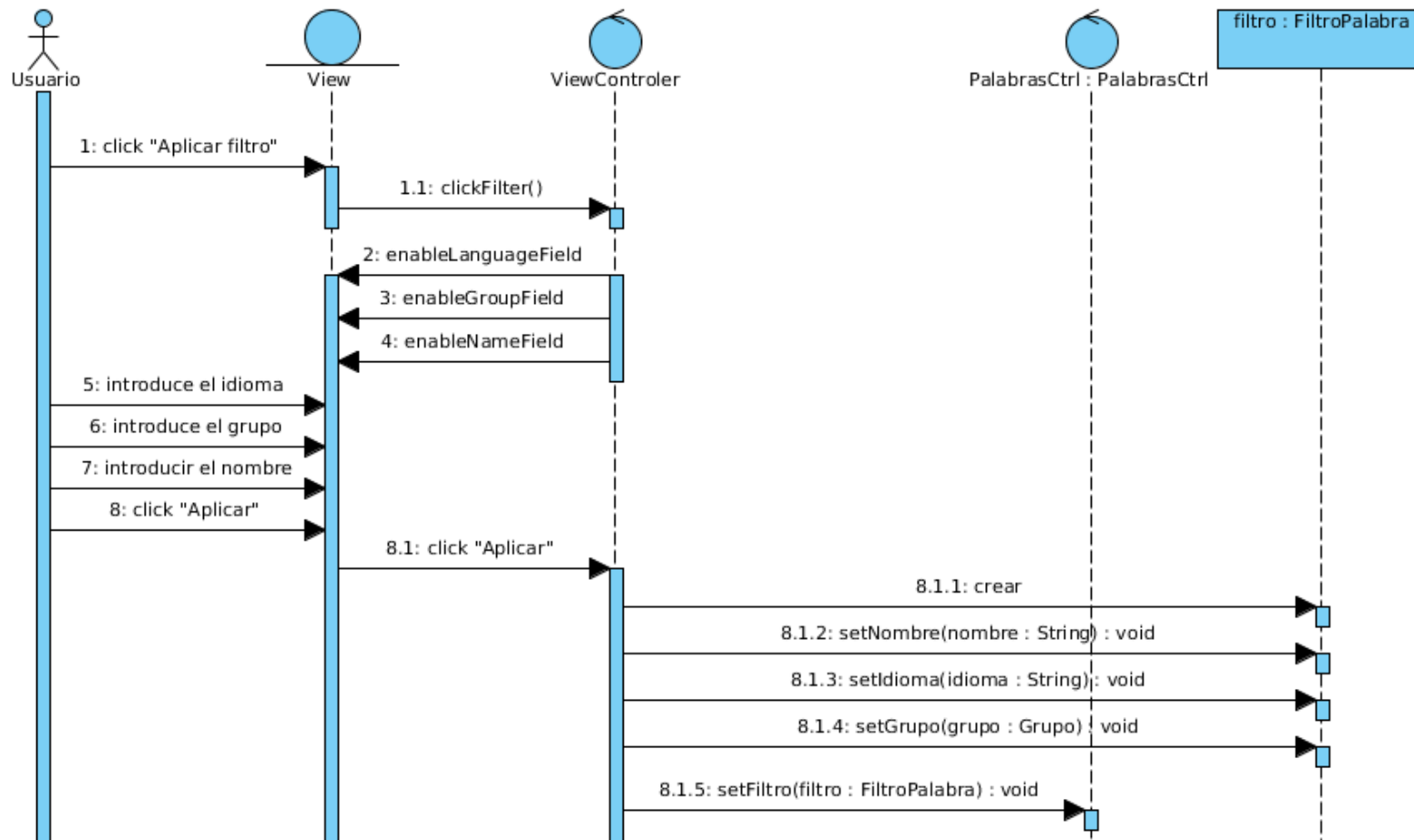


Diagrama de sequencia del escenario de aplicar el filtro de palabras

Añadir palabra			
Descripción	EL sistema le mostrará al usuario un formulario con los campos necesarios para rellenar una palabra, y un <i>checkbox</i> que si lo marca, cuando pulse sobre crear la palabra, saldrá la pantalla de traducción.		
	Cuando pulsa sobre crear, el sistema comprobará que la palabra no exista en la base de datos y que los campos sean correctos. Una vez creada la palabra, limpia el formulario y deja al usuario introducir otra palabra. Así hasta que el se cierre la vista.		
Precondiciones	Ninguna.		
Postcondiciones	<ul style="list-style-type: none"> Palabra almacenada en la fuente de datos. 		
Flujo de eventos normal		Usuario	Sistema
	1	Selecciona la opción <i>añadir palabra</i> .	
	2		Muestra un formulario con todos los atributos de la palabra.
	3		Muestra un <i>checkbox</i> para traducir la palabra.
	4	Rellena los campos	
	5		Comprueba que la palabra no exista.
	6		Escribe en la fuente de datos la palabra.
	7		Limpia el formulario y vuelve al paso 2.
Flujo alternativo El usuario traduce la palabra.		Usuario	Sistema
	1	Selecciona la opción <i>añadir palabra</i> .	
	2		Muestra un formulario con todos los atributos de la palabra.
	3		Muestra un <i>checkbox</i> para traducir la palabra.
	4	Rellena los campos	
	5	Marca la opción de traducir la palabra	
	6		Comprueba que la palabra no exista.

	7		Escribe en la fuente de datos la palabra.
	8		Instancia el caso de uso <i>traducir palabra</i>.
	9		Limpia el formulario y vuelve al paso 2.
Flujo alternativo La palabra existe		Usuario	Sistema
	1	Selecciona la opción <i>añadir palabra</i> .	
	2		Muestra un formulario con todos los atributos de la palabra.
	3		Muestra un <i>checkbox</i> para traducir la palabra.
	4	Rellena los campos	
	5		Comprueba que la palabra no exista.
	6		Vuelve al paso 4.

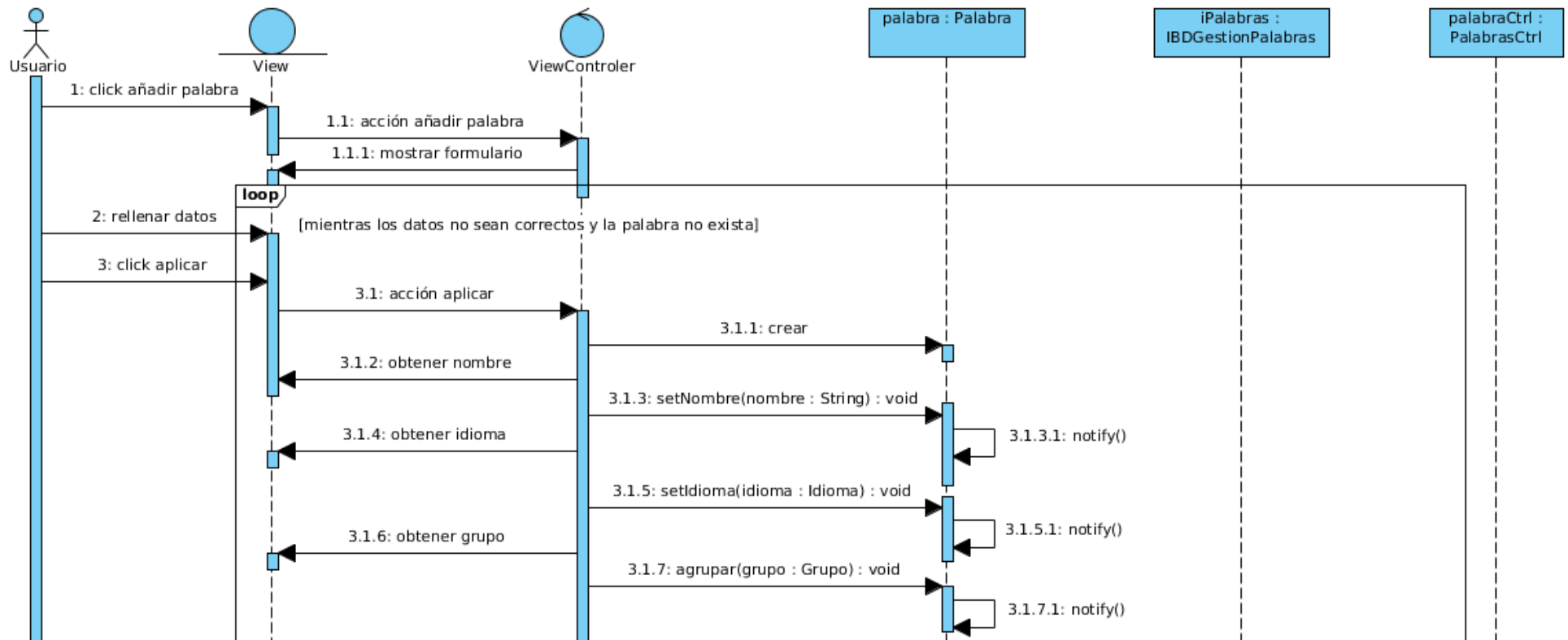


Diagrama de secuencia (parte 2) : Añadir palabra.

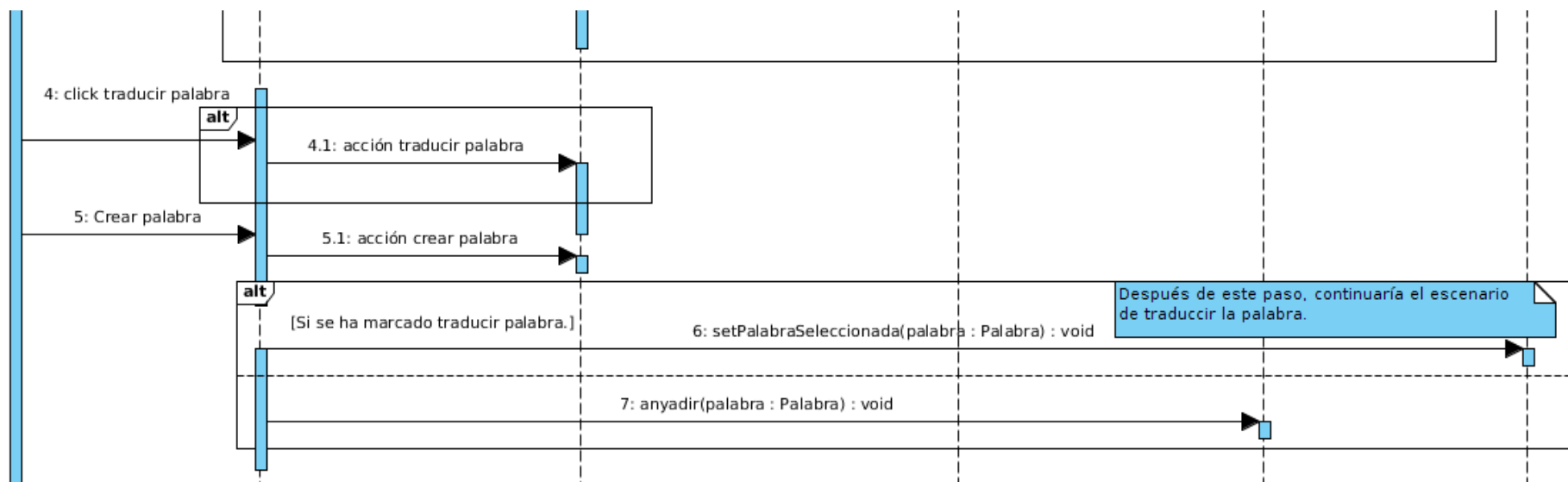


Diagrama de secuencia (parte 2) : Añadir palabra.

Seleccionar palabra																				
Descripción	El usuario selecciona una palabra y el sistema marca esta palabra como "seleccionada". Cuando esto sucede, el sistema habilita las opciones "Borrar palabra", "Modificar palabra" y "Traducir palabra"; que también corresponden a sus respectivos casos de uso.																			
Precondiciones	<ul style="list-style-type: none"> Se han mostrado las palabras. 																			
Postcondiciones	<ul style="list-style-type: none"> Se ha marcado una palabra como seleccionada. Opción <i>Borrar palabra</i>, habilitada. Opción <i>Modificar palabra</i>, habilitada. Opción <i>Traducir palabra</i>, habilitada. 																			
Flujo de eventos normal	<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>1</td><td>Selecciona una palabra.</td><td></td></tr> <tr> <td>2</td><td></td><td>Marca la palabra como seleccionada.</td></tr> <tr> <td>3</td><td></td><td>Habilitar la opción <i>borrar palabra</i>.</td></tr> <tr> <td>4</td><td></td><td>Habilitar la opción <i>modificar palabra</i>.</td></tr> <tr> <td>5</td><td></td><td>Habilitar la opción <i>traducir palabra</i>.</td></tr> </table>			Usuario	Sistema	1	Selecciona una palabra.		2		Marca la palabra como seleccionada.	3		Habilitar la opción <i>borrar palabra</i> .	4		Habilitar la opción <i>modificar palabra</i> .	5		Habilitar la opción <i>traducir palabra</i> .
	Usuario	Sistema																		
1	Selecciona una palabra.																			
2		Marca la palabra como seleccionada.																		
3		Habilitar la opción <i>borrar palabra</i> .																		
4		Habilitar la opción <i>modificar palabra</i> .																		
5		Habilitar la opción <i>traducir palabra</i> .																		

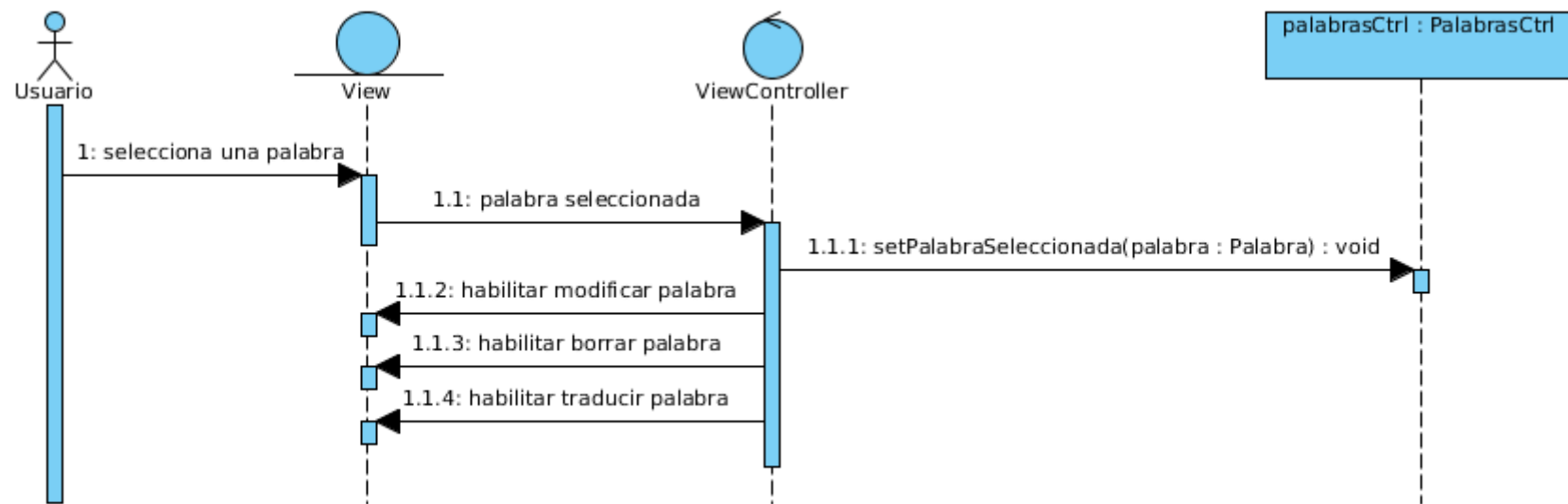


Diagrama de secuencia : Seleccionar palabra

Traducir palabra																	
Descripción	El sistema pide al usuario que filtre por un idioma. Estos idiomas son distintos al de la palabra. Cuando se seleccione el idioma, el sistema mostrará todas las palabras no relacionadas con la palabra. Dando la posibilidad al usuario de quitar traducciones y añadir traducciones.																
Precondiciones	<ul style="list-style-type: none"> Palabra marcada como seleccionada. 																
Postcondiciones	<ul style="list-style-type: none"> Se ha escrito una traducción en la fuente de datos, o se ha eliminado. 																
Flujo de eventos normal	<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>1</td><td>Selecciona la opción <i>traducir palabra</i>.</td><td></td></tr> <tr> <td>2</td><td></td><td>Pide al usuario el idioma de traducción.</td></tr> <tr> <td>3</td><td>Inserta el idioma.</td><td></td></tr> <tr> <td>4</td><td></td><td>Muestra las palabras traducibles al idioma seleccionado y las palabras traducidas ya.</td></tr> </table>			Usuario	Sistema	1	Selecciona la opción <i>traducir palabra</i> .		2		Pide al usuario el idioma de traducción.	3	Inserta el idioma.		4		Muestra las palabras traducibles al idioma seleccionado y las palabras traducidas ya.
	Usuario	Sistema															
1	Selecciona la opción <i>traducir palabra</i> .																
2		Pide al usuario el idioma de traducción.															
3	Inserta el idioma.																
4		Muestra las palabras traducibles al idioma seleccionado y las palabras traducidas ya.															
Flujo alternativo El usuario inserta una traducción	<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>5</td><td>Selecciona una posible traducción</td><td></td></tr> <tr> <td>6</td><td>Pulsa sobre <i>traducir</i>.</td><td></td></tr> <tr> <td>7</td><td></td><td>Escribe la traducción en la fuente de datos.</td></tr> <tr> <td>8</td><td></td><td>Vuelve al paso 4, del flujo normal.</td></tr> </table>			Usuario	Sistema	5	Selecciona una posible traducción		6	Pulsa sobre <i>traducir</i> .		7		Escribe la traducción en la fuente de datos.	8		Vuelve al paso 4, del flujo normal.
	Usuario	Sistema															
5	Selecciona una posible traducción																
6	Pulsa sobre <i>traducir</i> .																
7		Escribe la traducción en la fuente de datos.															
8		Vuelve al paso 4, del flujo normal.															
Flujo alternativo El usuario borra una traducción	<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>5</td><td>Selecciona una traducción existente.</td><td></td></tr> <tr> <td>6</td><td>Pulsa sobre <i>eliminar</i>.</td><td></td></tr> <tr> <td>7</td><td></td><td>Borra la traducción de la fuente de datos.</td></tr> <tr> <td>8</td><td></td><td>Vuelve al paso 4, del flujo normal.</td></tr> </table>			Usuario	Sistema	5	Selecciona una traducción existente.		6	Pulsa sobre <i>eliminar</i> .		7		Borra la traducción de la fuente de datos.	8		Vuelve al paso 4, del flujo normal.
	Usuario	Sistema															
5	Selecciona una traducción existente.																
6	Pulsa sobre <i>eliminar</i> .																
7		Borra la traducción de la fuente de datos.															
8		Vuelve al paso 4, del flujo normal.															

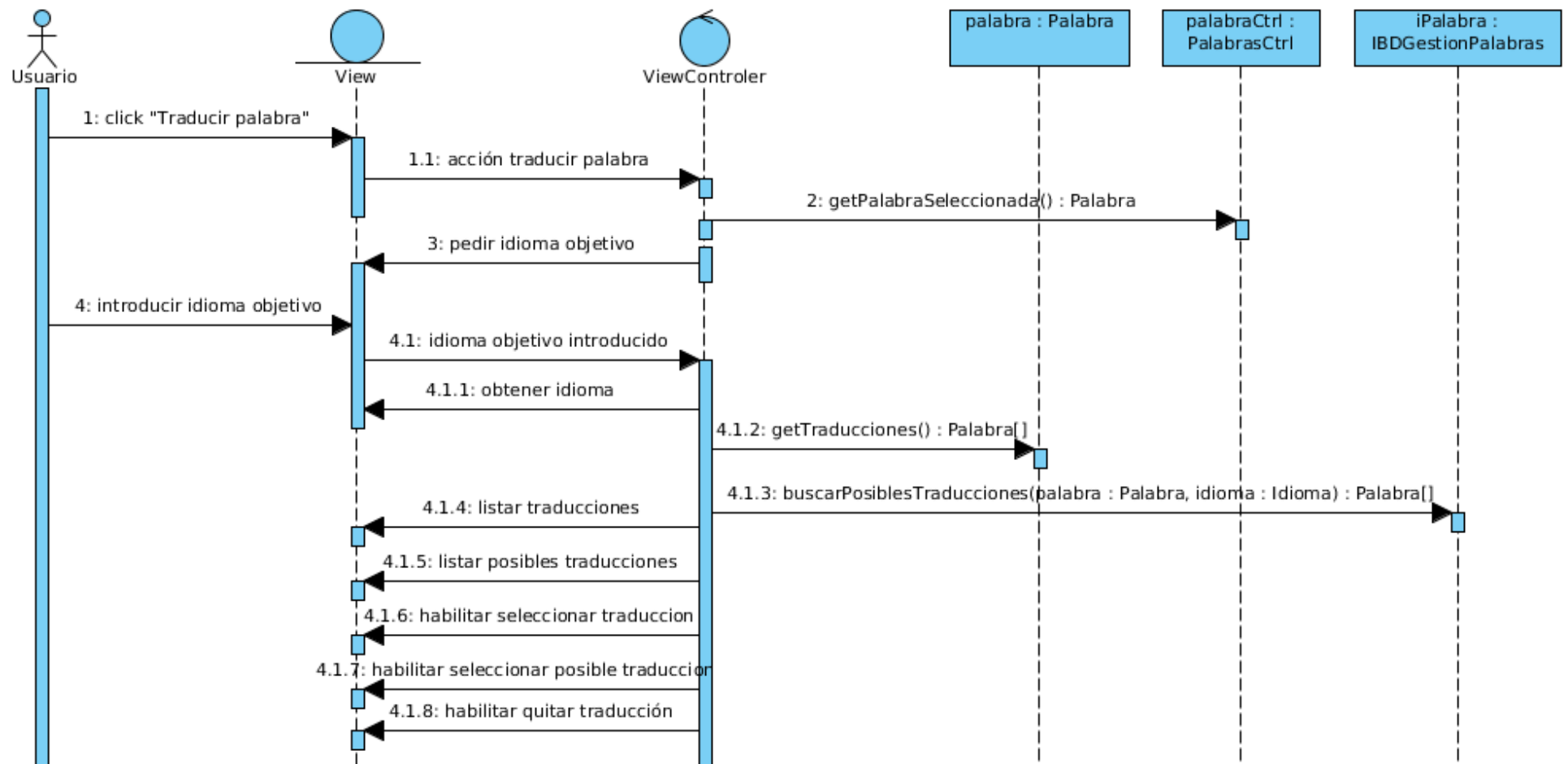


Diagrama de secuencia : Traducir palabra (Parte 1)

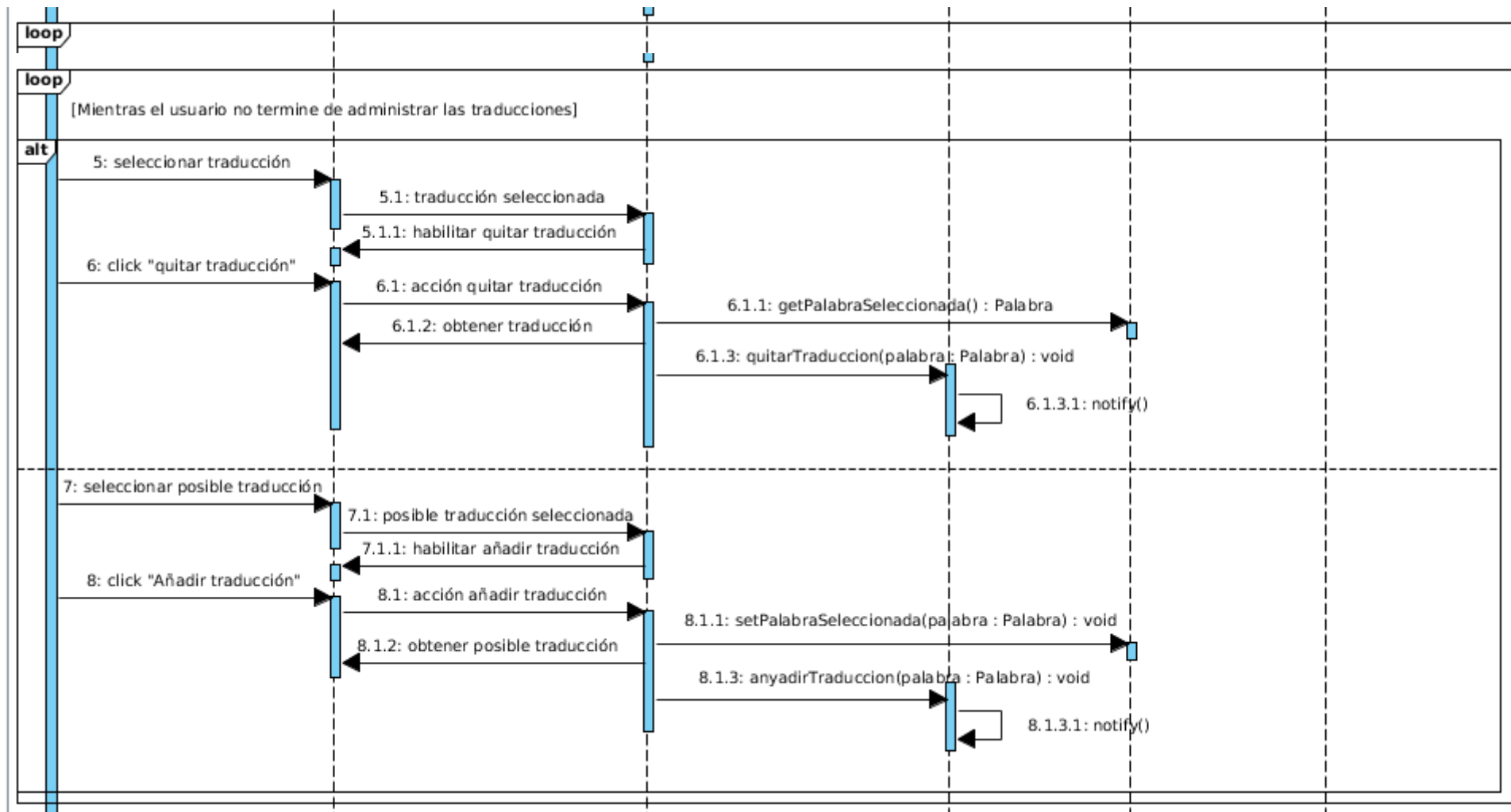


Diagrama de secuencia : Traducir palabra (Parte 2)

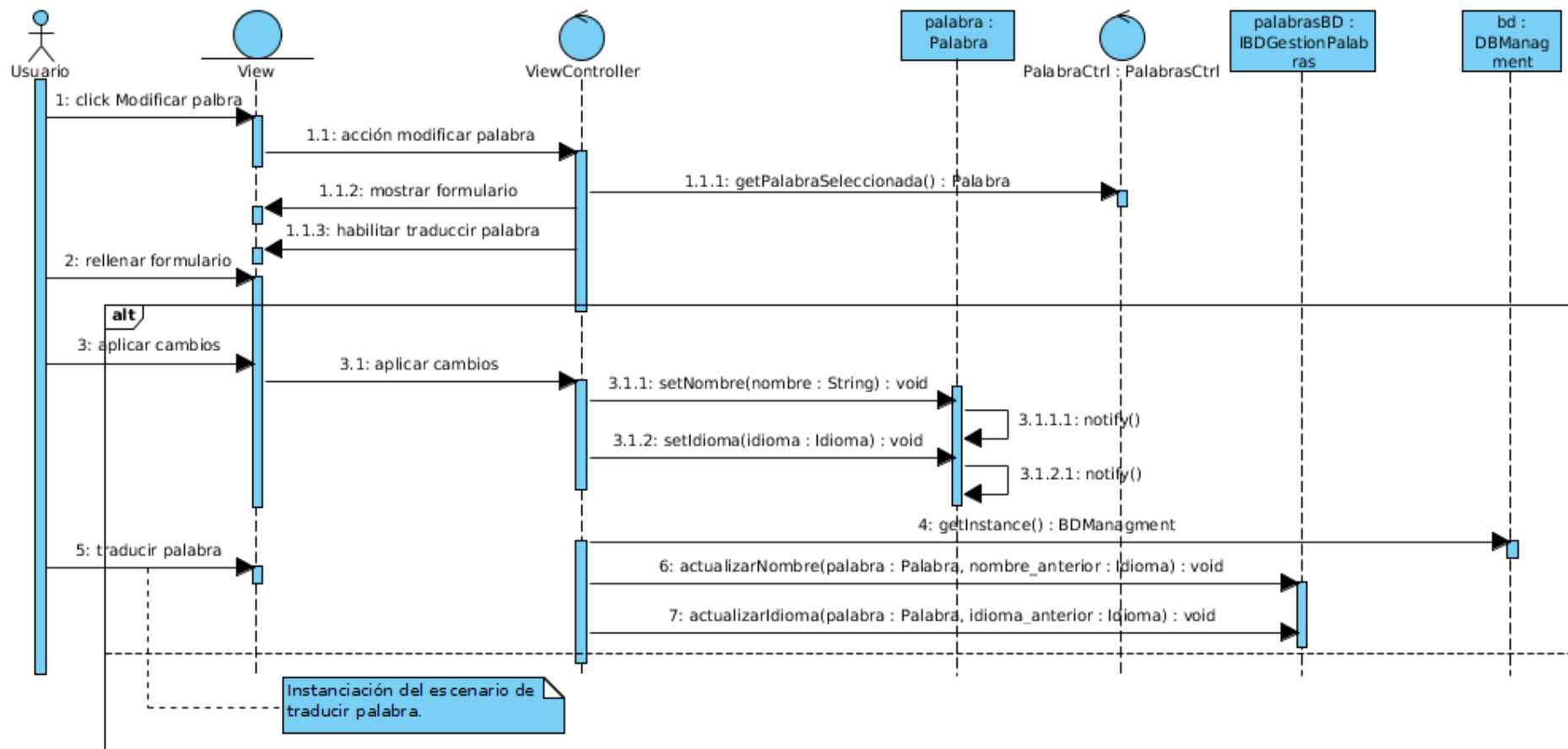


Diagrama de secuencia : Modificar palabras

Borrar palabra																				
Descripción	El sistema busca la palabra seleccionada en la fuente de datos.																			
Precondiciones	<ul style="list-style-type: none"> Palabra marcada como seleccionada. 																			
Postcondiciones	<ul style="list-style-type: none"> Palabra seleccionada borrada de la fuente de datos. Palabra desmarcada como seleccionada. 																			
Flujo de eventos normal																				
<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>1</td><td>Selecciona la opción <i>borrar palabra</i>.</td><td></td></tr> <tr> <td>2</td><td></td><td>Solicita confirmación.</td></tr> <tr> <td>3</td><td>Acepta confirmación.</td><td></td></tr> <tr> <td>4</td><td></td><td>Borra la palabra seleccionada de la fuente de datos.</td></tr> <tr> <td>5</td><td></td><td>Quitar como seleccionada la palabra.</td></tr> </table>				Usuario	Sistema	1	Selecciona la opción <i>borrar palabra</i> .		2		Solicita confirmación.	3	Acepta confirmación.		4		Borra la palabra seleccionada de la fuente de datos.	5		Quitar como seleccionada la palabra.
	Usuario	Sistema																		
1	Selecciona la opción <i>borrar palabra</i> .																			
2		Solicita confirmación.																		
3	Acepta confirmación.																			
4		Borra la palabra seleccionada de la fuente de datos.																		
5		Quitar como seleccionada la palabra.																		

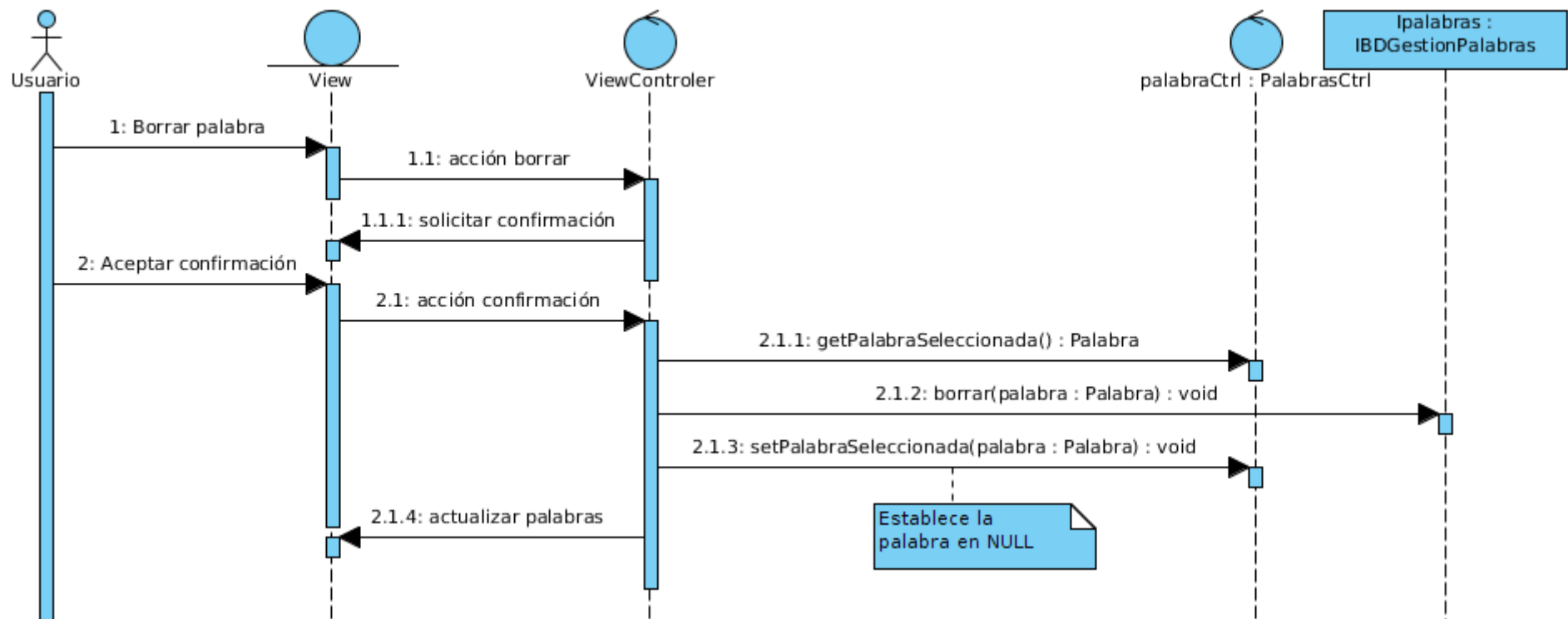


Diagrama de secuencia : Borrar palabra

Mostrar grupos		
Descripción	El sistema busca la palabra seleccionada en la fuente de datos. El sistema muestra al usuario todos los grupos en la fuente de datos. Habilitando las opciones de añadir un nuevo grupo y seleccionar un grupo.	
Precondiciones	Ninguna.	
Postcondiciones	<ul style="list-style-type: none"> • Opción <i>añadir grupo</i>, habilitada. • Posibilidad de seleccionar algún grupo, habilitada. 	
Flujo de eventos normal		
	1	Elige la opción de <i>mostrar grupos</i> .
	2	Consulta en la fuente de datos todos los grupos.
	3	Muestra todos los grupos.
	4	Habilita la opción <i>añadir grupo</i> .

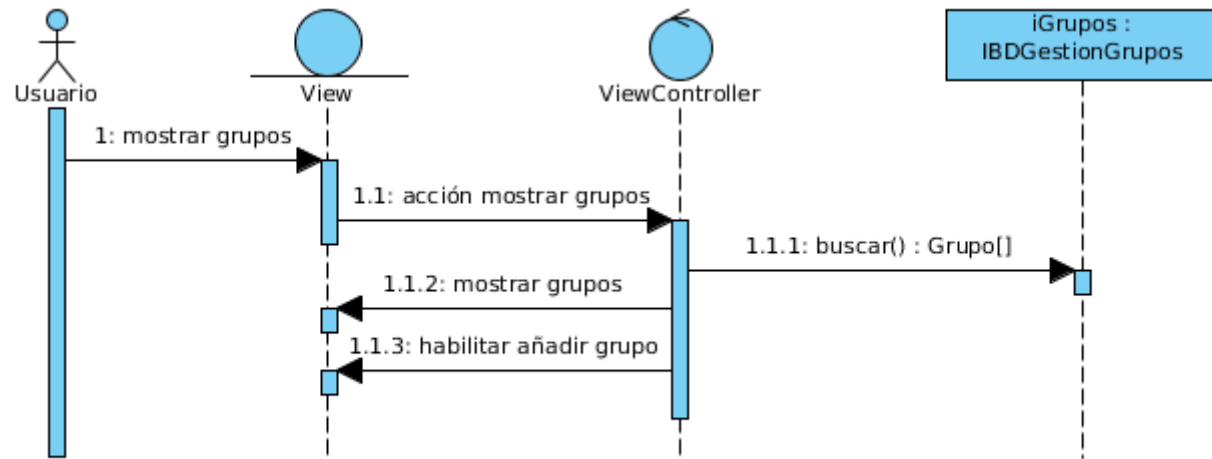


Diagrama de secuencia: Mostrar grupos.

Añadir grupo																				
Descripción	El sistema muestra al usuario un formulario, que podrá rellenar y cuando termine, el sistema aplicará los cambios en la base de datos.																			
Precondiciones	Ninguna.																			
Postcondiciones	<ul style="list-style-type: none"> Grupo escrito en la fuente de datos. 																			
Flujo de eventos normal	<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>1</td><td>Elige la opción de <i>nuevo grupo</i>.</td><td></td></tr> <tr> <td>2</td><td></td><td>Muestra un formulario con todos los campos del grupo.</td></tr> <tr> <td>3</td><td>Rellena los campos.</td><td></td></tr> <tr> <td>4</td><td>Pulsa sobre <i>Aplicar cambios</i>.</td><td></td></tr> <tr> <td>5</td><td></td><td>Escribe el grupo en la fuente de datos.</td></tr> </table>			Usuario	Sistema	1	Elige la opción de <i>nuevo grupo</i> .		2		Muestra un formulario con todos los campos del grupo.	3	Rellena los campos.		4	Pulsa sobre <i>Aplicar cambios</i> .		5		Escribe el grupo en la fuente de datos.
	Usuario	Sistema																		
1	Elige la opción de <i>nuevo grupo</i> .																			
2		Muestra un formulario con todos los campos del grupo.																		
3	Rellena los campos.																			
4	Pulsa sobre <i>Aplicar cambios</i> .																			
5		Escribe el grupo en la fuente de datos.																		

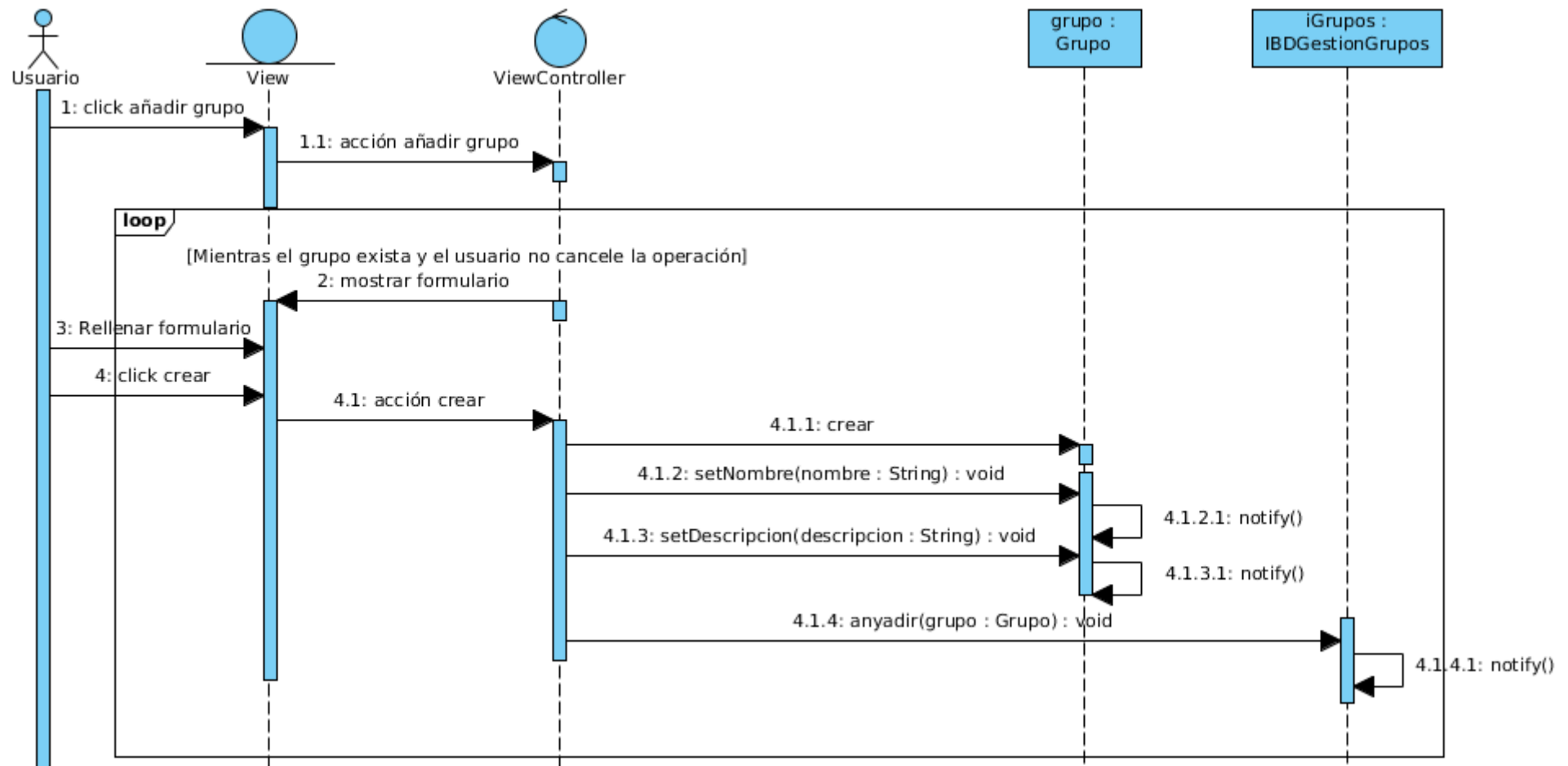


Diagrama de secuencia : Añadir grupo

Seleccionar grupo														
Descripción	El sistema habilita las opciones de modificar grupo, borrar grupo, administrar sus palabras y modificar grupo, cuando el usuario selecciona un grupo.													
Precondiciones	Ninguna.													
Postcondiciones	<ul style="list-style-type: none"> • Opción <i>borrar grupo</i>, habilitada. • Opción <i>modificar grupo</i>, habilitada. • Opción <i>administrar palabras</i>, habilitada. • Grupo seleccionado marcado. 													
Flujo de eventos normal	<table> <tr> <th></th><th>Usuario</th><th>Sistema</th></tr> <tr> <td>1</td><td>Selecciona un grupo.</td><td></td></tr> <tr> <td>2</td><td></td><td>Marca el grupo como seleccionado para todo el sistema.</td></tr> <tr> <td>3</td><td></td><td>Habilita la opción <i>borrar grupo</i>, <i>modificar grupo</i> y <i>administrar palabras</i>.</td></tr> </table>			Usuario	Sistema	1	Selecciona un grupo.		2		Marca el grupo como seleccionado para todo el sistema.	3		Habilita la opción <i>borrar grupo</i> , <i>modificar grupo</i> y <i>administrar palabras</i> .
	Usuario	Sistema												
1	Selecciona un grupo.													
2		Marca el grupo como seleccionado para todo el sistema.												
3		Habilita la opción <i>borrar grupo</i> , <i>modificar grupo</i> y <i>administrar palabras</i> .												

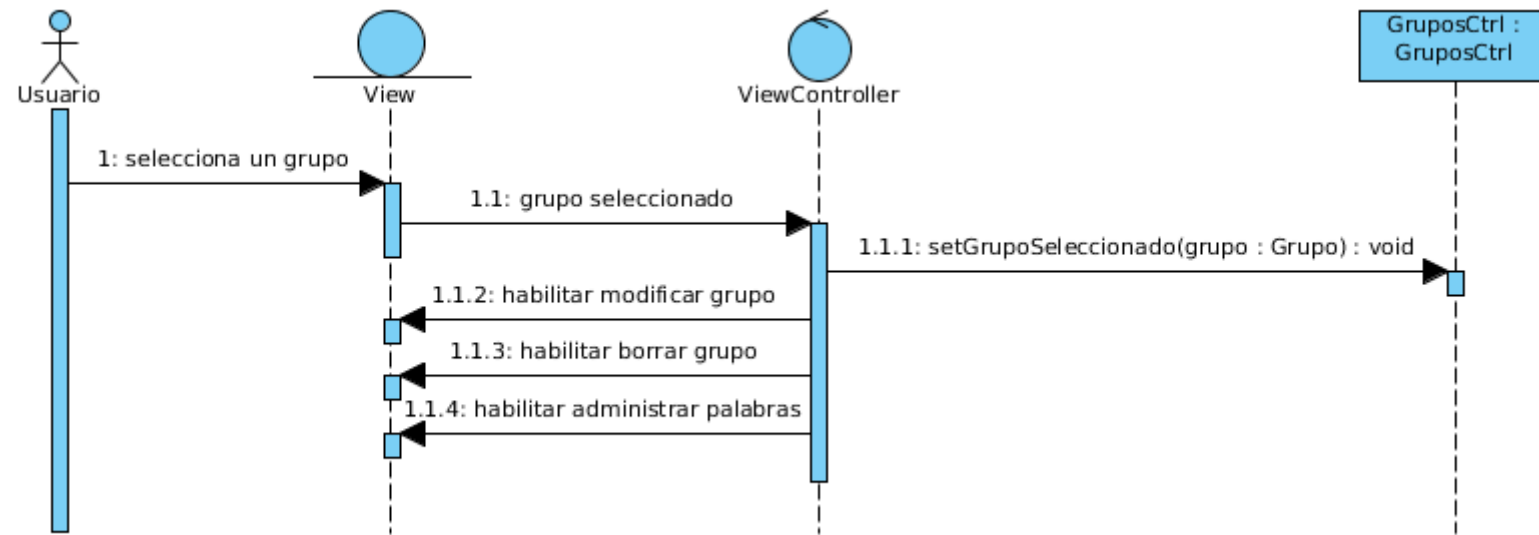


Diagrama de secuencia : Seleccionar grupo.

Administrar palabras		
Descripción	El sistema muestra todas las palabras agrupadas en el grupo seleccionado, y permite al usuario seleccionar una y desagruparla, o añadir la palabra al grupo.	
Precondiciones	<ul style="list-style-type: none"> Grupo marcado como seleccionado. 	
Postcondiciones	<ul style="list-style-type: none"> Una palabra agrupada o desagrupada, depende de los flujos alternativos. 	
Flujo de eventos normal		
Flujo alternativo Agrupar palabra		
Flujo alternativo Desagrupar palabra		

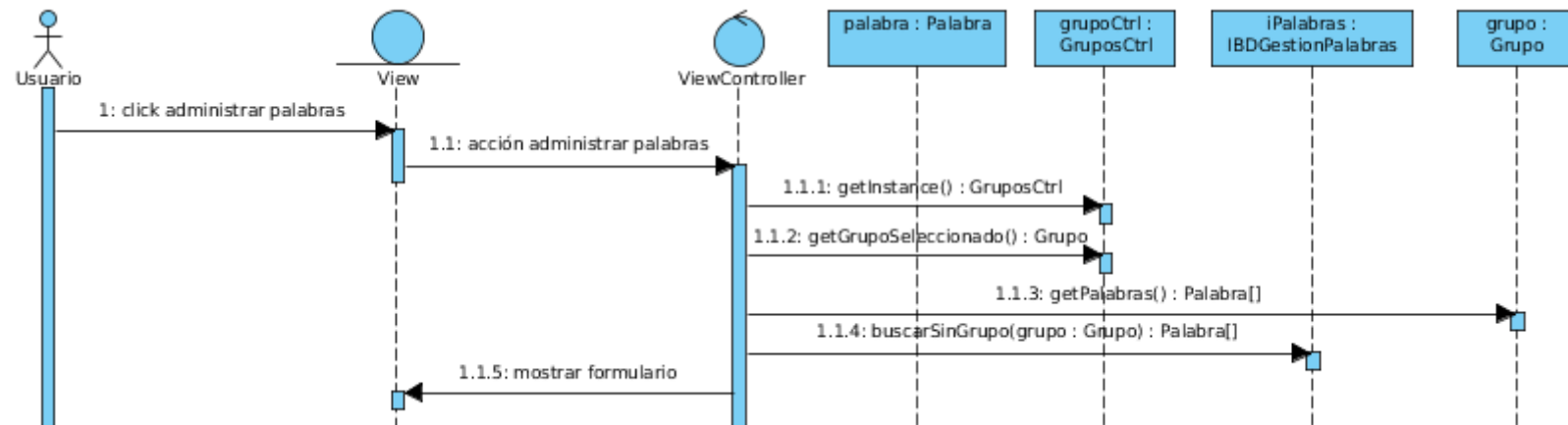


Diagrama de secuencia : Administrar palabras (Parte 1)

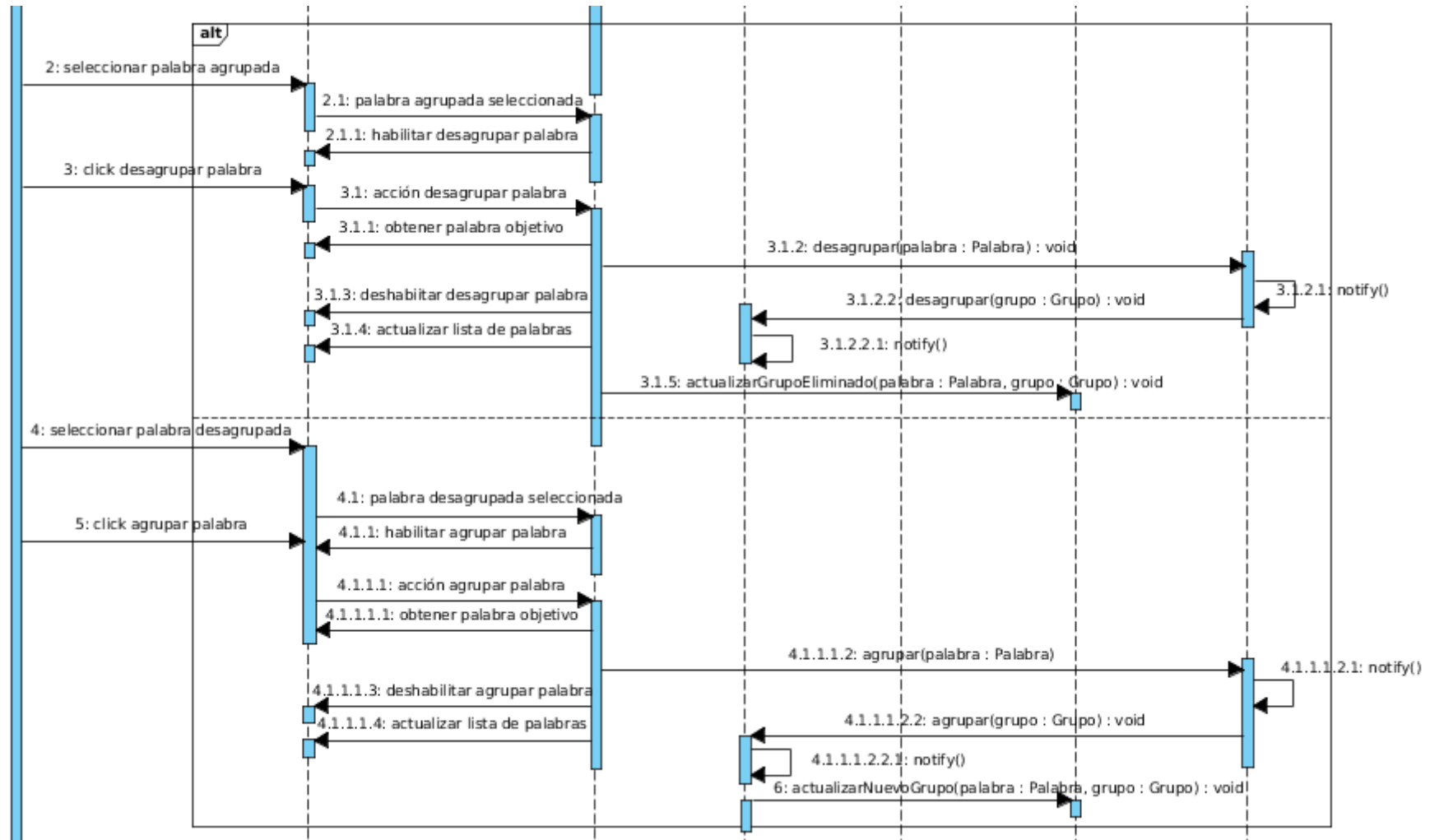


Diagrama de secuencia : Administrar palabras (Parte 2)

Modificar grupo		
Descripción	El sistema muestra al usuario un formulario donde podrá modificar los atributos del grupo. Cuando termine, el sistema aplicará los cambios en la fuente de datos y dará la posibilidad de administrar sus palabras.	
Precondiciones	<ul style="list-style-type: none"> Grupo marcado como seleccionado. 	
Postcondiciones	<ul style="list-style-type: none"> Actualización del grupo escrito en la fuente de datos. 	
Flujo de eventos normal		
	1	Selecciona la opción <i>modificar grupo</i> .
	2	Muestra un formulario con todos los atributos del grupo, para que el usuario pueda modificarlos.
	3	Rellena los campos.
	4	Pulsa sobre aplicar cambios.
	5	Se escriben la actualización en la fuente de datos.

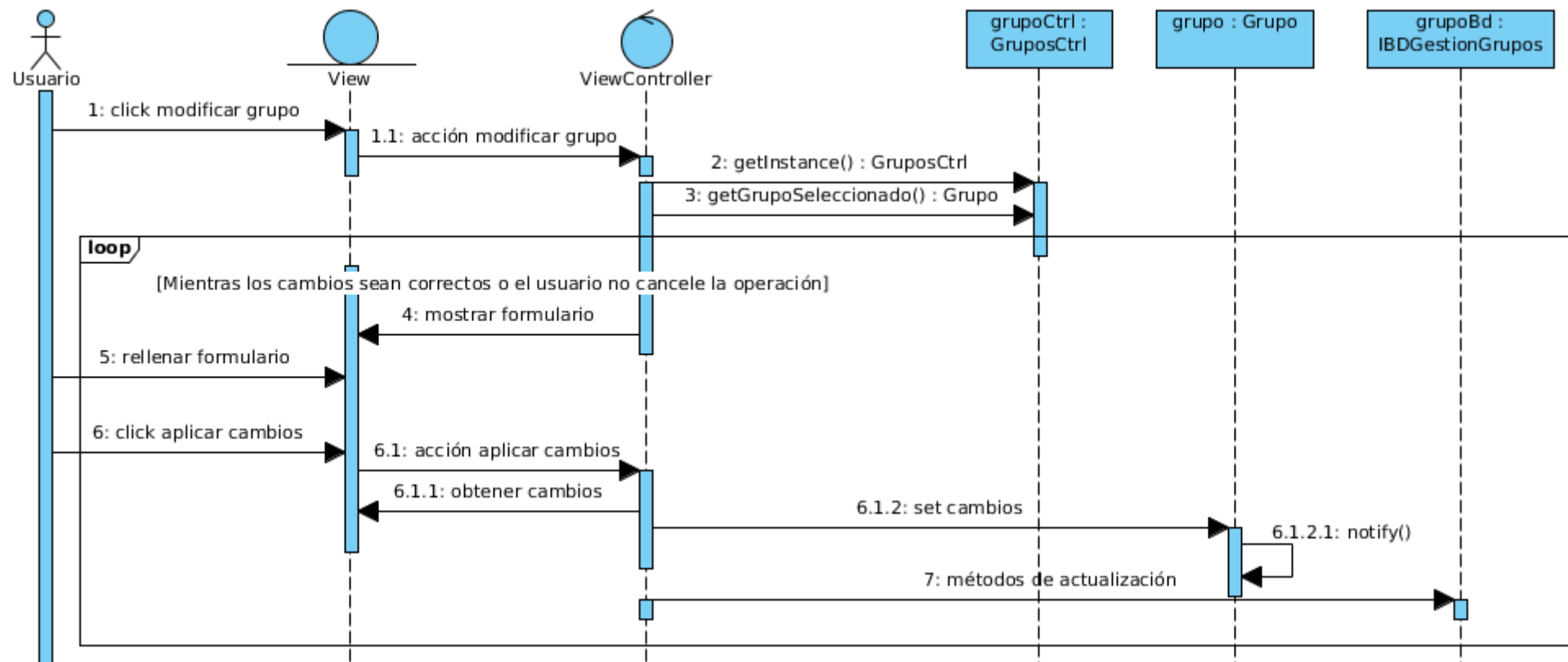
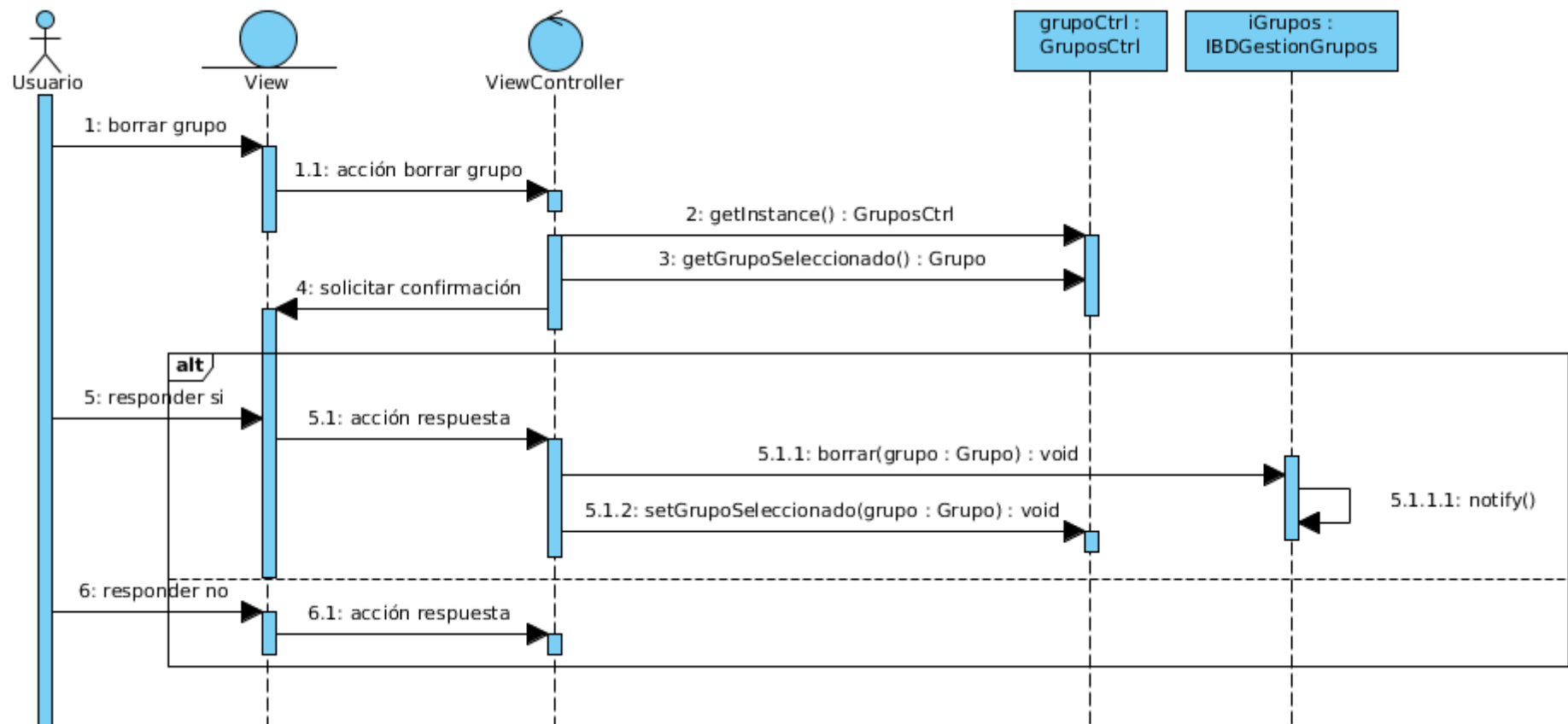


Diagrama de secuencia: Modificar grupo

Borrar grupo		
Descripción	El sistema elimina el grupo seleccionado de la fuente de datos.	
Precondiciones	<ul style="list-style-type: none"> Grupo marcado como seleccionado. 	
Postcondiciones	<ul style="list-style-type: none"> Grupo seleccionado borrado de la base de datos. 	
Flujo de eventos normal		

	Usuario	Sistema
1	Selecciona la opción <i>borrar grupo</i> .	
2		Solicita confirmación.
3	Acepta la confirmación.	
4		Elimina el grupo e la fuente de datos.
5		Desmarca el grupo como seleccionado para el sistema.



Diagramas de secuencia : Borrar grupo

Mostrar idiomas		
Descripción	El sistema muestra todos los idiomas que existen en la fuente de datos al usuario, y permite "Añadir un idioma". Como alternativa, si selecciona un idioma, permite borrarlo.	
Precondiciones	Ninguna.	
Postcondiciones	<ul style="list-style-type: none">• Posibilidad de seleccionar una palabra añadida.• Opción <i>añadir idioma</i>, habilitada.• Opción <i>modificar idioma</i> y <i>borrar idioma</i>, habilitada si se produce la alternativa.• Idioma marcado como seleccionado, si se cumple la alternativa.	
Flujo de eventos normal		
Flujo alternativo El usuario selecciona un idioma		

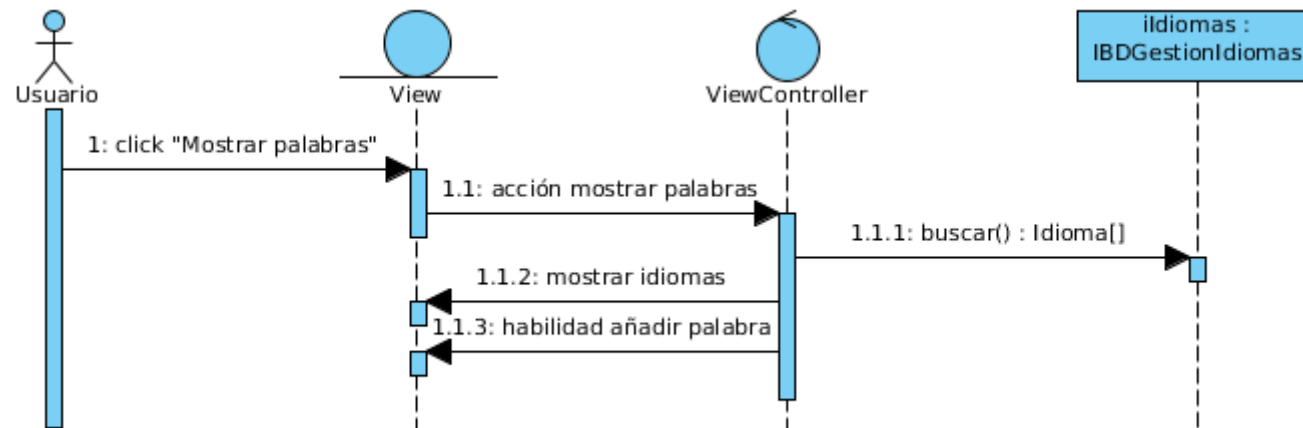


Diagrama de secuencia: Mostrar idiomas

Añadir idioma		
Descripción	El sistema muestra al usuario un formulario con los campos del idioma y lo escribe en la fuente de datos.	
Precondiciones	<ul style="list-style-type: none"> Un idioma marcado como seleccionado previamente. 	
Postcondiciones	<ul style="list-style-type: none"> Un idioma escrito en la fuente de datos. 	
Flujo de eventos normal		
	1	Selecciona la opción <i>añadir idioma</i> .
	2	Muestra un formulario con los campos necesarios para el idioma.
	3	Rellena los campos.
	4	Comprueba que el idioma no exista.
	5	Escribe en la fuente de datos el idioma introducido por el usuario.
Flujo alternativos El idioma existe		
	1	Selecciona la opción <i>añadir idioma</i> .
	2	Muestra un formulario con los campos necesarios para el idioma.
	3	Rellena los campos.
	4	Comprueba que el idioma no exista.
	5	Vuelve al paso 2.

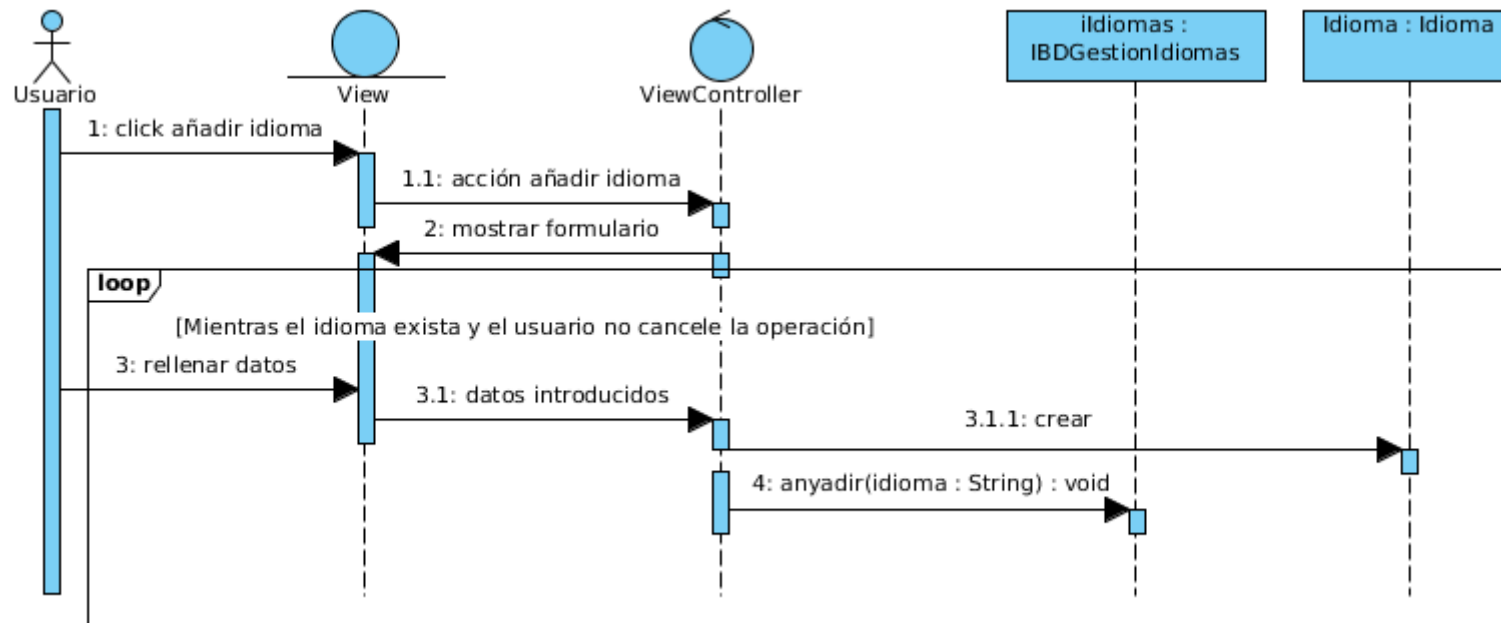


Diagrama de secuencia : Añadir idioma

Modificar idioma		
Descripción	El sistema muestra al usuario un formulario con los campos del idioma seleccionado rellenos. El usuario podrá modificarlos y aplicar los cambios en la fuente de datos.	
Precondiciones	<ul style="list-style-type: none"> Idioma marcado como seleccionado. 	
Postcondiciones	<ul style="list-style-type: none"> Idioma seleccionado actualizado en la fuente de datos. 	
Flujo de eventos normal		
	1	Selecciona la opción <i>modificar grupo</i> .
	2	Muestra un formulario con los datos del idioma y medios para modificarlos.
	3	Rellena los campos.
	4	Escribe la actualización del idioma en la fuente de datos.

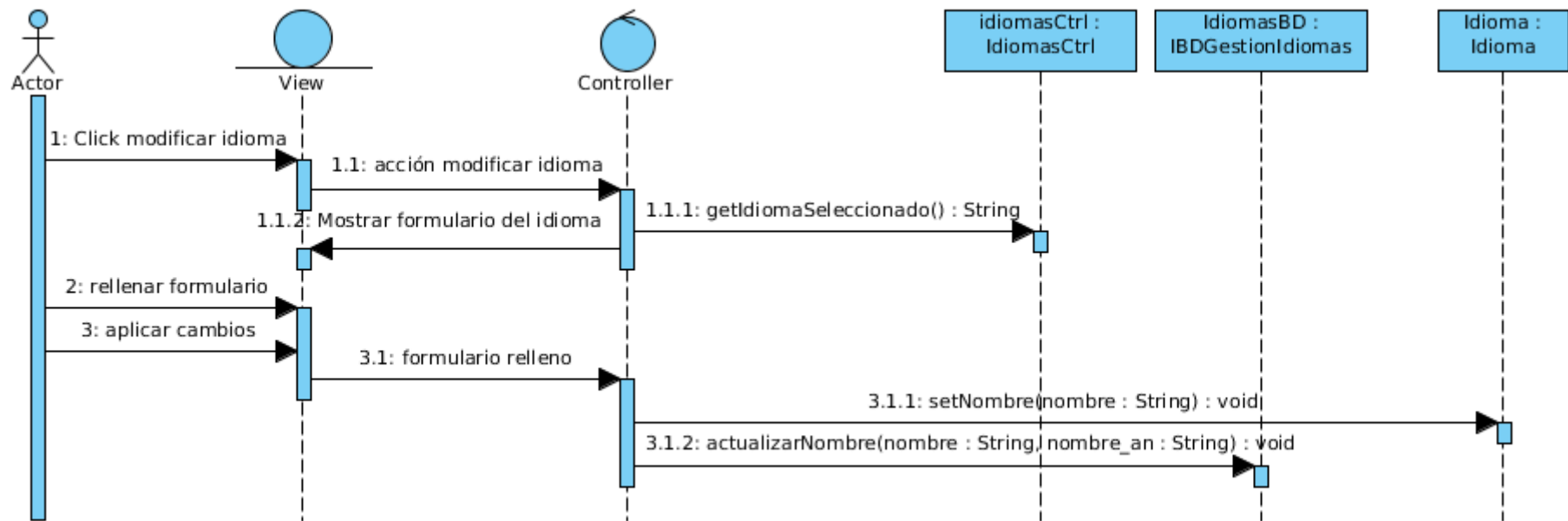


Diagrama de secuencia : Modificar idioma

Borrar idioma		
Descripción	El sistema permite al usuario borrar una palabra seleccionada.	
Precondiciones	<ul style="list-style-type: none"> Idioma marcado como seleccionado. 	
Postcondiciones	<ul style="list-style-type: none"> Idioma seleccionado borrado de la fuente de datos. Idioma seleccionado desmarcado del sistema. 	
Flujo de eventos normal		
	1	Selecciona la opción <i>borrar grupo</i> .
	2	Pide confirmación.
	3	Acepta el procedimiento.
	4	Borra de la fuente de datos la palabra seleccionada.
	5	Desmarca el idioma seleccionado del sistema.

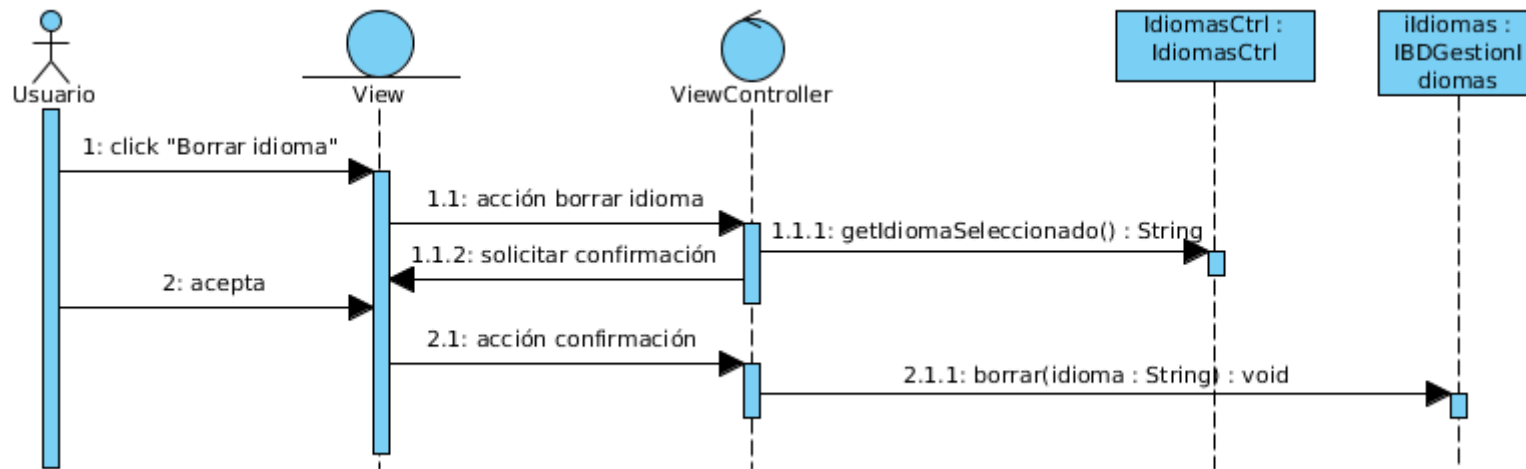


Diagrama de secuencia : Borrar idioma

5.1.2.2. Diagrama de casos de uso – juego de traducir palabra

Los casos de uso que aparecen en este diagrama son:

- Inicia juego.
- Traducir palabra.

Inicia juego		
Descripción	El sistema muestra al usuario un grupo o idioma para filtrar las palabras que saldrán en el juego, y prepara el juego para su transcurso.	
Precondiciones	<ul style="list-style-type: none"> • El juego no está iniciado. 	
Postcondiciones	<ul style="list-style-type: none"> • El juego iniciado. • Partida creada con las palabras aleatorias. • Primera palabra marcada. 	
Flujo de eventos normal		
	1	Usuario Selecciona la opción <i>iniciar juego</i> .
	2	Sistema Consulta en la fuente de datos todos los idiomas
	3	Consulta en la fuente de datos todos los grupos.
	4	Muestra los dos resultados para que el usuario seleccione un idioma o grupo objetivo y un idioma de traducción.
	5	Selecciona un grupo o idioma.
	6	Selecciona el idioma de traducción.
	7	Comprueba que las dos opciones no sean la misma.
	8	Consulta en la fuente de datos todas las palabras filtradas por la opción del usuario y que no sean del idioma de traducción.
	9	Comprueba si hay suficientes palabras para empezar la partida.

Flujo de eventos normal		Usuario	Sistema
	10		Crea la partida con el resultado de la consulta anterior.
	11		Marca la primera palabra
	12		Instancia al caso de uso "Traducir palabra".

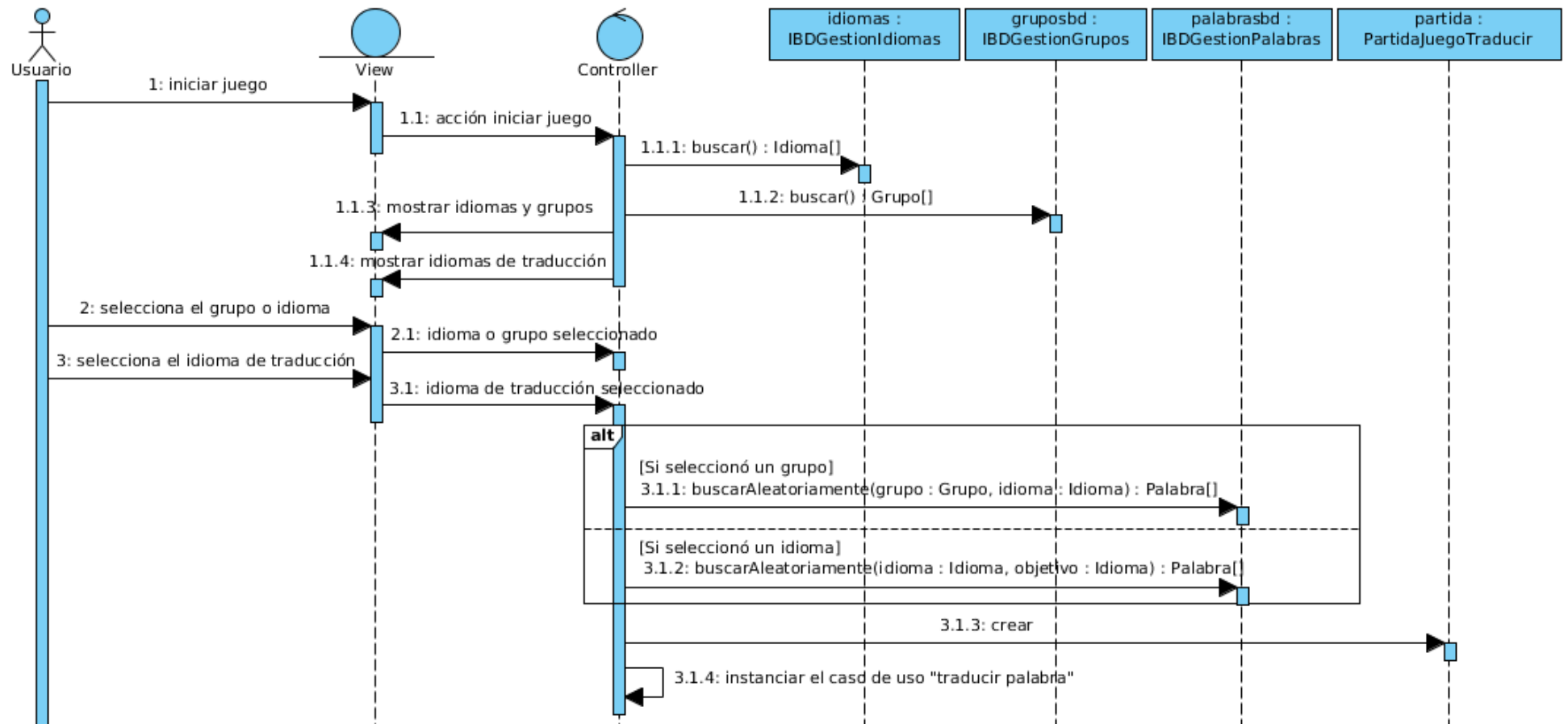


Diagrama de secuencia : Iniciar juego

Traducir palabras			
Descripción	El sistema muestra la palabra que el usuario tiene que traducir, y permite que introduzca su traducción.		
Precondiciones	<ul style="list-style-type: none"> El juego iniciado. Una palabra marcada. 		
Postcondiciones	<ul style="list-style-type: none"> Un acierto o un fallo apuntado en la partida. Siguiente palabra marcada. 		
Flujo de eventos normal			
		Usuario	Sistema
	1	Selecciona <i>siguiente palabra</i> .	
	2		Obtiene la palabra marcada.
	3		Muestra la palabra.
	4		Habilita un campo para que introduzca la traducción.
	5	Introducir traducción.	
	6		Si es correcta, se apunta como acierto; si no, se apunta como fallo.
	7		Comprueba si hay siguiente palabra.
	8		Marca la siguiente palabra.
Flujo alternativo No hay siguiente palabra. (Fin de partida)	9		Vuelve al paso 2.
		Usuario	Sistema
	7		Comprueba si hay siguiente palabra.
	8		Muestra la puntuación de la partida.
	9		Habilita la opción <i>reiniciar la partida</i> .
	10	Selecciona la opción <i>reiniciar partida</i> .	
11		Instancia el caso de uso <i>iniciar juego</i> .	

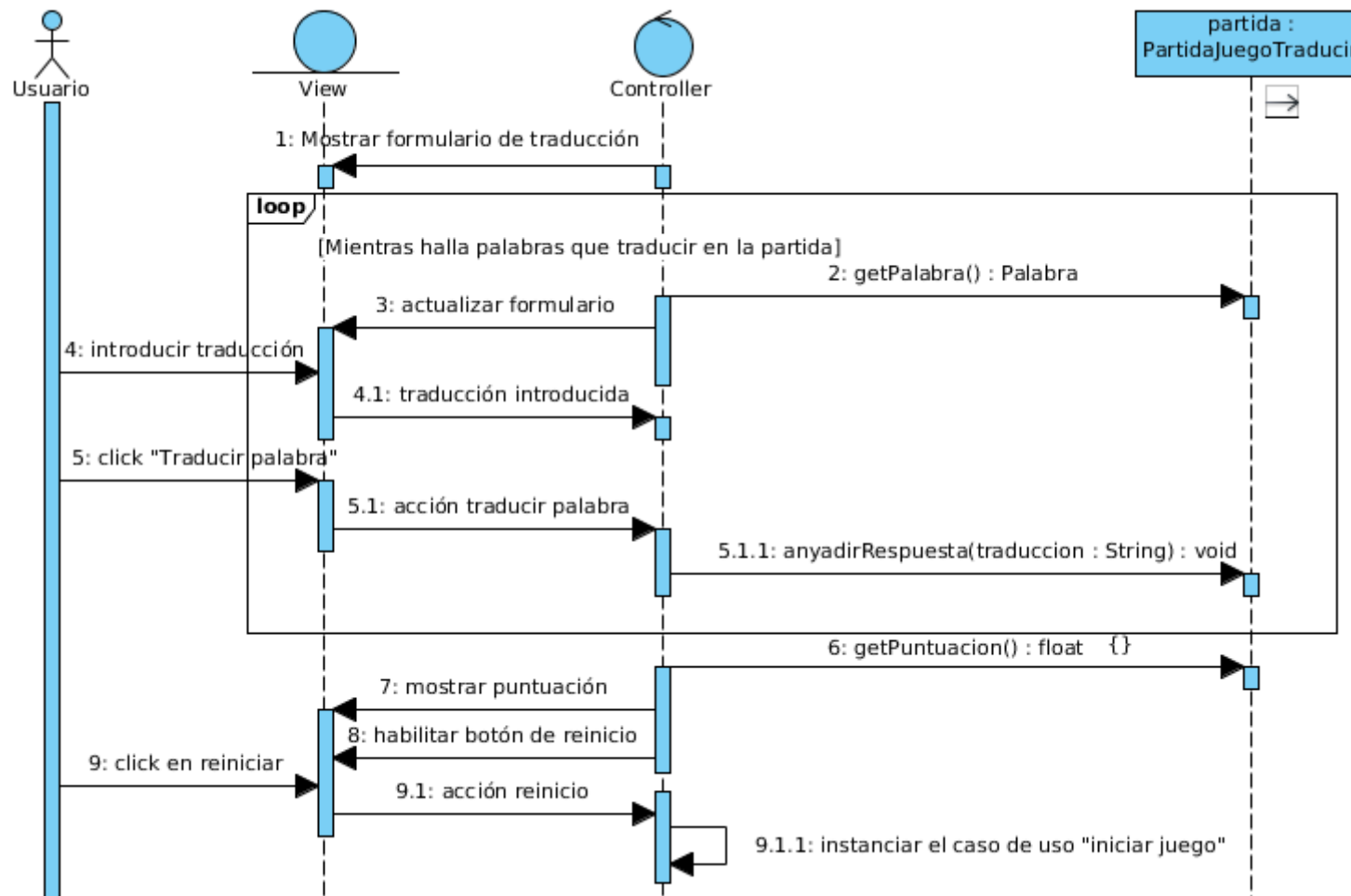


Diagrama de secuencia : traducir palabras

5.1.3. Diseño de componentes

La visión final del proyecto física es la siguiente:



Diagrama 3: diagrama de componentes.

Como se puede ver en el diagrama, el proyecto dispondrá de un ejecutable que se comunicará con el fichero “palabras.db3” a través de la interfaz “BDManagment”. La interfaz usará el driver conveniente para comunicarse con la base de datos, que en este caso, asumo que es SQLite3. Del diseño de la interfaz hablaré más detenidamente en su diseño.

5.1.4. Diseño de estructura de clases y paquetes

En el diseño de paquetes he establecido la siguiente estructura:

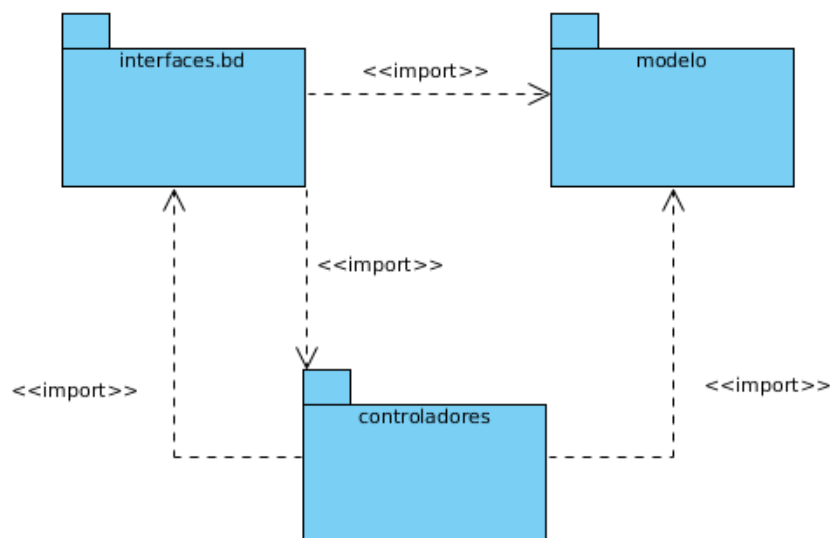


Diagrama 4: diagrama de paquetes

Como se puede ver en el diagrama, aparecen 3 paquetes: *interfaces.bd*, *modelo* y *controladores*. Dentro del paquete *controladores* hay dos paquetes más, *controadores.administracion* y *controladores.juego*. La estructura, junto con su descripción de qué agrupa, es la siguiente:

- *modelo*

Contiene todas las clases de datos del dominio del problema, con su propia semántica y algunas clases de utilidad; como la persistencia de datos, etc.

- *interfaces.bd*

Contiene las clases abstractas e interfaces para comunicarse con la fuente de datos.

- *controladores*

Contiene dos paquetes más para controlar el comportamiento del programa, según los diagramas de secuencia. Estos se han dividido en dos partes, *juegos* y *administracion*.

- *controladores.juegos*

Contiene las clases necesarias para llevar a cabo el juego de traducir palabras, reflejado en el diagrama 2; casos de uso del juego de traducir palabras.

- *controladores.administracion*

Contiene las clases necesarias para llevar a cabo la administración según los requisitos del diagrama 1; casos de uso de administración de palabras, grupos y idiomas.

A continuación, entraré en detalles de sobre las clases que hay en cada paquete, menos en *controladores* que no hay ninguna clase, ya que agrupa sólo paquetes.

5.1.4.1. Diagrama de clases – modelo

En el siguiente diagrama se muestran las relaciones de las clases *java.util.Observable*, *ClasePersistente*, *DatosObjeto*, *Idioma*, *Palabra* y *Grupo*.

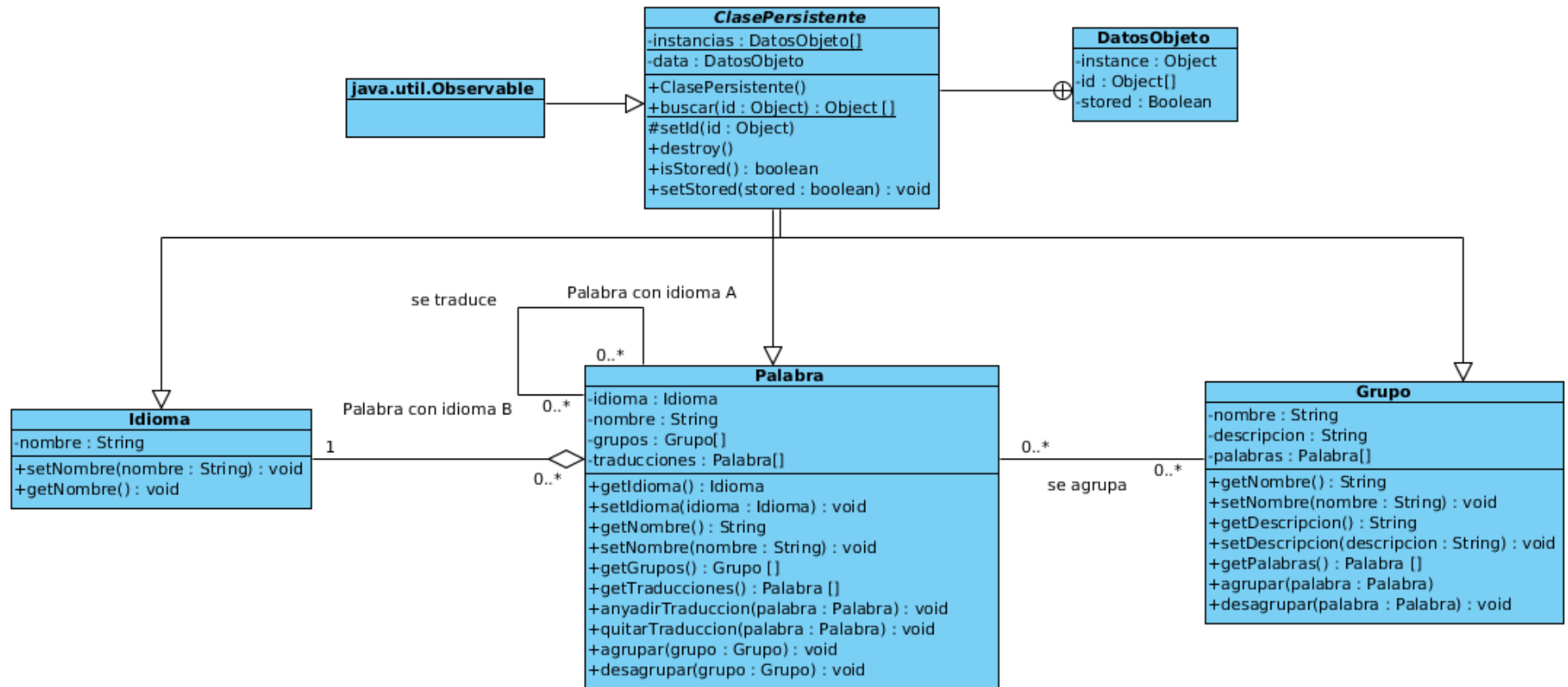
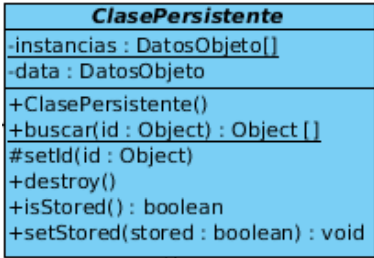



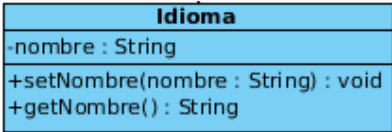
Diagrama 5: Diagrama de clases del paquete modelo

A continuación explicaré para qué sirve cada una de ellas:

ClasePersistente	
Imagen	 <pre> classDiagram class ClasePersistente { -instancias : DatosObjeto[] -data : DatosObjeto +ClasePersistente() +buscar(id : Object) : Object [] #setId(id : Object) +destroy() +isStored() : boolean +setStored(stored : boolean) : void } </pre>
Descripción	<p>Representa un objeto que podrá hacerse persistente en memoria y controlará, mediante su ID, que solamente halla uno en ejecución. Proporciona métodos para la búsqueda y la destrucción de objetos en memoria.</p> <p>Las clases hijas deberán de establecer su id, mediante el método apropiado.</p>
Atributos	<ul style="list-style-type: none"> • <u>instancias : DatosObjeto []</u> Contiene todas las instancias de todos los objetos persistentes del programa. • <u>data : DatosObjeto</u> Contiene los datos relevantes para esta clase del objeto persistente. Como su ID, su instancia y si está guardado en algún medio persistente.
Métodos	<ul style="list-style-type: none"> • <u>buscar(id : Object [])</u> Array con el id del objeto buscado. Es un array porque, por ejemplo, palabra tiene dos identificadores, su nombre y su idioma. • <u>destroy() : void</u> Desvincula el objeto con el array de objetos. Así, el recolector de basura de Java se encargará de liberarlo de memoria. • <u>setId(id : Object []) : void</u> Establece el ID del objeto. Este método tiene que ser usado por una clase hija, y es necesario para buscar el objeto en memoria. • <u>IsStored() : boolean</u> Permite conocer si el objeto está almacenado en alguna fuente. • <u>setStored(stored : boolean) : void</u> Permite establecer si el objeto se ha borrado o si se a añadido a la fuente de datos.

Relaciones	<ul style="list-style-type: none"> • <i>ObjetoPersistente</i> → <i>Idioma, Palabra y Grupo</i> Con esta relación hago persistentes a las clases idioma, grupo y palabra. Que son las más importantes, debido al ámbito del problema. Así, si se quiere buscar una palabra, no se tendrá que consultar la fuente de datos, si no, la memoria. • <i>ObjetoPersistente</i> → <i>DatosObjeto</i> ObjetoPersistente usa DatosObjeto para almacenar los datos necesarios para la búsqueda de objetos en memoria. • <i>ObjetoPersistente</i> → <i>java.util.Observable</i> Con esta relación establezco la posibilidad de que cualquiera clase que esté interesada en los cambios del objeto persistente.
Comentarios	<p>Esta clase está creada para optimizar la búsqueda en la fuente de datos. Puesto que de no existir esta clase, cada vez que se hiciera una consulta a la fuente de datos, se crearía un nuevo objeto, dando la posibilidad de que hubiera dos objetos iguales en memoria.</p> <p>Por ejemplo; imagine que hacemos dos consultas, en las que preguntamos, “dame todas las palabras del grupo informática y programación”. Y en los dos grupos está la palabra “disco duro”. Sin esta clase, se crearían dos objetos con el nombre disco duro, lo que repercutiría en un gasto de memoria.</p> <p>Todo el funcionamiento de esta clase será explicado detenidamente en el apartado de codificación de este documento.</p>

DatosObjeto	
Imagen	 <pre> classDiagram class DatosObjeto { -instance : Object -id : Object[] -stored : Boolean } </pre>
Descripción	Es una clase interna de <i>ObjetoPersistente</i> y aloja la información relevante para la búsqueda de un objeto en memoria. Esta información es su ID, si está almacenado en la fuente de datos, o no, y su instancia.
Atributos	<ul style="list-style-type: none"> • <u>instancia : Object</u> Guarda la instancia del objeto que se ha instanciado. • <u>id : Object[]</u> Contiene los objetos identificadores del objeto persistente. Estos son necesarios para encontrar el objeto en memoria. • <u>stored : boolean</u> Guarda si el objeto está almacenado en alguna fuente de datos.
Relaciones	<ul style="list-style-type: none"> • <i>DatosObjeto</i> → <i>ObjetoPersistente</i> Es una clase interna de <i>ObjetoPersistente</i>, así lo refleja esta relación. Puesto que el propósito de esta clase es proporcionar información a su clase contenedora.
Comentarios	<p>Esta clase la he creado para organizar un poco la clase <i>ObjetoPersistente</i>. Así, alojando la información en esta clase, puedo pasar toda la información de método a método, sin pasar una lista de atributos.</p> <p>Como se puede observar, en UML no he reflejado ningún <i>setters</i> and <i>getters</i>, puesto que no es necesario, la clase contenedora tiene total acceso a esta clase.</p>

Idioma	
Imagen	 <pre> classDiagram class Idioma { -nombre : String +setNombre(nombre : String) : void +getNombre() : String } </pre>
Descripción	Representa un idioma de las palabras, del que queremos conocer su nombre.
Atributos	<ul style="list-style-type: none"> • <u>nombre : String</u> Nombre del idioma.
Métodos	<ul style="list-style-type: none"> • <u>setNombre(nombre : String)</u> Permite cambiar el nombre del idioma por el pasado como argumento. • <u>getNombre() : String</u> Permite conocer el nombre del idioma.
Relaciones	<ul style="list-style-type: none"> • <i>Idioma → Palabra</i> Un idioma lo tienen varias palabras. • <i>Idioma → ObjetoPersistente</i> Con esta relación, el programa no tendrá que crear un nuevo idioma por cada palabra con el mismo. Si no, que todas las palabras tendrán la misma instancia.
Comentarios	Inicialmente, esta clase no existía. Ya que yo pensé que un atributo como el idioma, debería ser parte de la palabra. Pero, una vez puesta en marcha la codificación, me interesaba que si se cambia el idioma, se cambie en todas las palabras. Esto es posible si todas tienen la misma instancia.

Palabra				
magen	<table><tr><th>Palabra</th></tr><tr><td>-idioma : Idioma -nombre : String -grupos : Grupo[] -traducciones : Palabra[]</td></tr><tr><td>+getIdioma() : Idioma +setIdioma(idioma : Idioma) : void +getNombre() : String +setNombre(nombre : String) : void +getGrupos() : Grupo [] +getTraducciones() : Palabra [] +anyadirTraduccion(palabra : Palabra) : void +quitarTraduccion(palabra : Palabra) : void +agrupar(grupo : Grupo) : void +desagrupar(grupo : Grupo) : void</td></tr></table>	Palabra	-idioma : Idioma -nombre : String -grupos : Grupo[] -traducciones : Palabra[]	+getIdioma() : Idioma +setIdioma(idioma : Idioma) : void +getNombre() : String +setNombre(nombre : String) : void +getGrupos() : Grupo [] +getTraducciones() : Palabra [] +anyadirTraduccion(palabra : Palabra) : void +quitarTraduccion(palabra : Palabra) : void +agrupar(grupo : Grupo) : void +desagrupar(grupo : Grupo) : void
Palabra				
-idioma : Idioma -nombre : String -grupos : Grupo[] -traducciones : Palabra[]				
+getIdioma() : Idioma +setIdioma(idioma : Idioma) : void +getNombre() : String +setNombre(nombre : String) : void +getGrupos() : Grupo [] +getTraducciones() : Palabra [] +anyadirTraduccion(palabra : Palabra) : void +quitarTraduccion(palabra : Palabra) : void +agrupar(grupo : Grupo) : void +desagrupar(grupo : Grupo) : void				
Descripción	Representa una palabra, de la que se quiere conocer su nombre, su idioma, sus grupos y sus traducciones en otro idioma distinto.			
Atributos	<ul style="list-style-type: none"><u>idioma : Idioma</u> Almacena el idioma de la palabra.<u>nombre : String</u> Almacena el nombre de la palabra.<u>grupos : Grupo[]</u> Almacena todos los grupos en los que aparece la palabra.<u>traducciones : Palabra[]</u> Almacena todas las palabras que traducen la palabra a otros idiomas.			
Métodos	<ul style="list-style-type: none"><u>setNombre(nombre : String)</u> Permite cambiar el nombre de la palabra.<u>getNombre() : String</u> Permite conocer el nombre de la palabra.<u>setIdioma(idioma : Idioma) : void</u> Permite establecer un nuevo idioma para la palabra. Esto provocará que se borren todas las traducciones de la palabra. Puesto que si esto ocurre, no tienen sentido las palabras que hubiera.<u>getIdioma() : void</u> Permite conocer el idioma de la palabra.<u>anyadirTraduccion(palabra : Palabra) : void</u> Permite añadir una traducción a la palabra. La traducción debe de ser de distinto idioma a la palabra. Este método también enlaza la traducción a la palabra.			

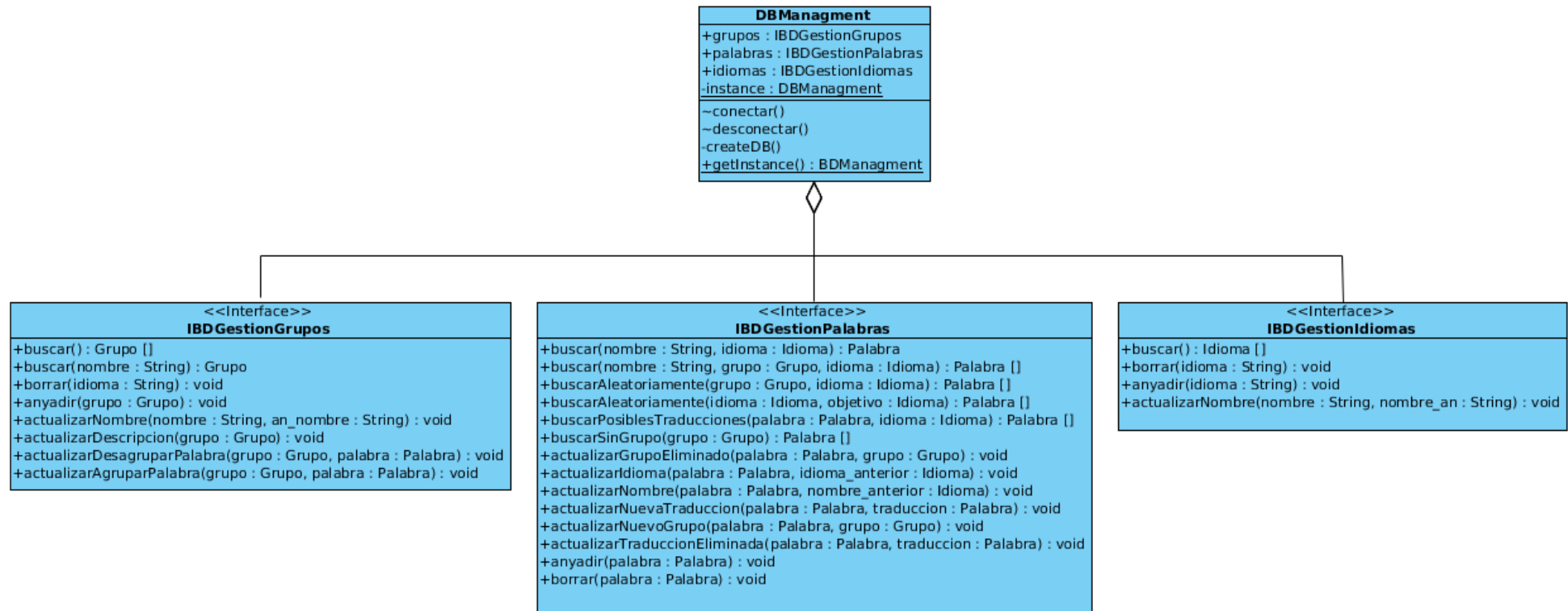
<p>Métodos</p>	<ul style="list-style-type: none"> • <u><i>quitarTraduccion(palabra : Palabra) : void</i></u> Elimina de las traducciones la palabra pasada como argumento, este método también lo hará en sentido contrario, es decir; que también quitará la palabra de las traducciones de la traducción. • <u><i>agrupar(grupo : Grupo) : void</i></u> Agrupar la palabra en el grupo pasado como argumento. Este método es bidireccional. Es decir, que añadirá la palabra a las palabras agrupadas del grupo. • <u><i>desagrupar(grupo : Grupo) : void</i></u> Desagrupa la palabra en el grupo pasado como argumento. Este método es bidireccional. Es decir, que eliminará la palabra de las palabras agrupadas del grupo. • <u><i>getGrupos() : Grupo[]</i></u> Permite conocer todos los grupos que agrupan a la palabra. • <u><i>getTraducciones() : Palabra[]</i></u> Permite conocer todas las traducciones de la palabra.
<p>Relaciones</p>	<ul style="list-style-type: none"> • <i>Palabra</i> → <i>Idioma</i> Un idioma lo tienen varias palabras. • <i>Palabra</i> → <i>Grupo</i> Una palabra puede aparecer agrupada en varios grupos. • <i>Idioma</i> → <i>ObjetoPersistente</i> Hace que el programa cree solamente una palabra con un nombre y un idioma en la memoria, o al menos, mantenga su instancia.

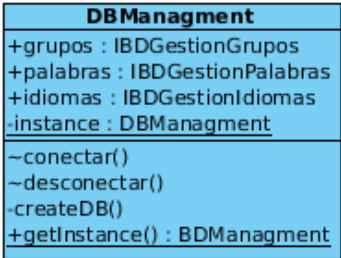
Grupo												
Imagen	<table><tr><th>Grupo</th></tr><tr><td>-nombre : String</td></tr><tr><td>-descripcion : String</td></tr><tr><td>-palabras : Palabra[]</td></tr><tr><td>+getNombre() : String</td></tr><tr><td>+setNombre(nombre : String) : void</td></tr><tr><td>+getDescripcion() : String</td></tr><tr><td>+setDescripcion(descripcion : String) : void</td></tr><tr><td>+getPalabras() : Palabra []</td></tr><tr><td>+agrupar(palabra : Palabra)</td></tr><tr><td>+desagrupar(palabra : Palabra) : void</td></tr></table>	Grupo	-nombre : String	-descripcion : String	-palabras : Palabra[]	+getNombre() : String	+setNombre(nombre : String) : void	+getDescripcion() : String	+setDescripcion(descripcion : String) : void	+getPalabras() : Palabra []	+agrupar(palabra : Palabra)	+desagrupar(palabra : Palabra) : void
Grupo												
-nombre : String												
-descripcion : String												
-palabras : Palabra[]												
+getNombre() : String												
+setNombre(nombre : String) : void												
+getDescripcion() : String												
+setDescripcion(descripcion : String) : void												
+getPalabras() : Palabra []												
+agrupar(palabra : Palabra)												
+desagrupar(palabra : Palabra) : void												
Descripción	Representa una unidad organizativa de palabras, de la que se quiere conocer su nombre, una descripción, opcional, y las palabras que agrupa.											
Atributos	<ul style="list-style-type: none">• <u>nombre : String</u> Nombre del grupo.• <u>descripcion : String</u> Descripción de qué agrupa el grupo, es opcional.• <u>palabras : Palabra[]</u> Array de palabras que agrupa el grupo.											
Métodos	<ul style="list-style-type: none">• <u>setNombre(nombre : String)</u> Permite cambiar el nombre del grupo por el pasado como argumento.• <u>getNombre() : String</u> Permite conocer el nombre del grupo.• <u>setDescripcion(descripcion : String) : void</u> Permite cambiar la descripción del grupo.• <u>getDescripcion() : String</u> Permite conocer la descripción del grupo.• <u>agrupar(palabra : Palabra) : void</u> Agrupar la palabra pasada como argumento al grupo. Este método es bidireccional, es decir, también añadirá a la palabra el grupo.• <u>desagrupar(palabra : Palabra) : void</u> Desagrupa la palabra pasada como argumento al grupo. Este método es bidireccional, es decir, también eliminará de la palabra el grupo.• <u>getPalabras() : Palabra[]</u> Permite conocer todas la palabras agrupadas por el grupo.											

Relaciones	<ul style="list-style-type: none">• <i>Grupo</i> → <i>Palabra</i> Un grupo lo agrupa a varias palabras.• <i>Grupo</i> → <i>ObjetoPersistente</i> Permite que el programa mantenga una sólo instancia en memoria del un grupo con un nombre como identificador.
-------------------	---

5.1.4.2. Diagrama de clases – interfaces.bd

En el siguiente diagrama de clases se muestran la clase abstracta *DBManagment* y las interfaces *IBDGestionGrupos*, *IBDGestionPalabras* y *IBDGestionIdiomas* :



BDManagment	
Imagen	 <pre> classDiagram class DBManagment { +grupos : IBDGestionGrupos +palabras : IBDGestionPalabras +idiomas : IBDGestionIdiomas -instance : DBManagment ~conectar() ~desconectar() ~createDB() +getInstance() : DBManagment } </pre>
Descripción	<p>Representa la clase padre de todas las administraciones relacionadas con las bases de datos del programa Lanword. La funcionalidad de esta clase se divide a la gestión de los idiomas, las palabras y los grupos. Para ello, están las interfaces, que deberán ser implementadas, <i>IBDGestionGrupos</i>, <i>IBDGestionIdiomas</i> y <i>IBDGestionPalabras</i>.</p> <p>A parte de la funcionalidad que aportan las interfaces, <i>BDManagment</i> contiene las funcionalidades que tratan directamente la conexión a la base de datos. Como por ejemplo, conectarse a ella, desconectarse o crear la base de datos.</p> <p>Es una clase <i>Singleton</i>, por lo que todo el programa tendrá acceso a una estancia.</p>
Atributos	<ul style="list-style-type: none"> • <u><i>grupos : IBDGestionGrupos</i></u> Almacena la instancia de la clase controladora del gestión de grupos. • <u><i>palabras : IBDGestionPalabras</i></u> Almacena la instancia de la clase controladora del gestión de palabras. • <u><i>idiomas : IBDGestionIdiomas</i></u> Almacena la instancia de la clase controladora del gestión de idiomas. • <u><i>instancia : BDManagment</i></u> Almacena la instancia de la propia clase, es utilizada por el método <i>getInstance()</i>.
Métodos	<ul style="list-style-type: none"> • <u><i>conectar() : void</i></u> Establece una conexión con la base de datos y la mantiene abierta. • <u><i>desconectar() : void</i></u> Pone fin a la conexión con la base de datos. • <u><i>createDB() : void</i></u> Ejecuta el DDL de la base de datos a través de la conexión abierta. • <u><i>getInstance() : BDManagment</i></u> Permite conseguir acceso a esta clase.

Relaciones	<ul style="list-style-type: none">• <i>BDManagment</i> → <i>IBDGestionIdiomas</i>, <i>IBDGestionGrupos</i> y <i>IBDGestionPalabras</i> Establece que la administración de la base de datos (<i>BDManagment</i>) tiene la funcionalidad de la administración de grupos, palabras y idiomas.
-------------------	--

IBDGestionIdiomas	
Imagen	<div> <div><<Interface>></div> <div>IBDGestionIdiomas</div> <div> +buscar() : Idioma [] +borrar(idioma : String) : void +anyadir(idioma : String) : void +actualizarNombre(nombre : String, nombre_an : String) : void </div> </div>
Descripción	Esta clase agrupa todas las funciones necesarias para obtener información de la fuente de datos, a cerca de los idiomas.
Métodos	<ul style="list-style-type: none"> • <u>buscar() : Idioma[]</u> Permite conoer todos los idiomas escritos en la base de datos. • <u>borrar(idioma : String) : void</u> Permite borrar un idioma mediante su nombre de la fuente de datos. • <u>anyadir(idioma : String) : void</u> Permite escribir una nueva palabra en la fuente de datos, mediante su nombre. • <u>actualizarNombre(nombre : String, nombre_an : String) : void</u> Permite actualizar en la fuente de datos el nombre de un idioma, conociendo también el nombre anterior.

IBDGestionPalabras	
Imagen	<pre> <<Interface>> IBDGestionPalabras +buscar(nombre : String, idioma : Idioma) : Palabra +buscar(nombre : String, grupo : Grupo, idioma : Idioma) : Palabra [] +buscarAleatoriamente(grupo : Grupo, idioma : Idioma) : Palabra [] +buscarAleatoriamente(idioma : Idioma, objetivo : Idioma) : Palabra [] +buscarPosiblesTraducciones(palabra : Palabra, idioma : Idioma) : Palabra [] +buscarSinGrupo(grupo : Grupo) : Palabra [] +actualizarGrupoEliminado(palabra : Palabra, grupo : Grupo) : void +actualizarIdioma(palabra : Palabra, idioma_anterior : Idioma) : void +actualizarNombre(palabra : Palabra, nombre_anterior : Idioma) : void +actualizarNuevaTraduccion(palabra : Palabra, traduccion : Palabra) : void +actualizarNuevoGrupo(palabra : Palabra, grupo : Grupo) : void +actualizarTraduccionEliminada(palabra : Palabra, traduccion : Palabra) : void +anyadir(palabra : Palabra) : void +borrar(palabra : Palabra) : void </pre>
Descripción	<p>Esta clase agrupa todas las funciones necesarias para obtener información de la fuente de datos, a cerca de las palabras.</p>
Métodos	<ul style="list-style-type: none"> • <u><i>buscar(nombre : String, idioma : Idioma) : Palabra</i></u> Permite obtener una palabra de la fuente de datos conociendo su nombre y su idioma. • <u><i>buscar(nombre : String, grupo : Grupo, idioma : Idioma) : Palabra[]</i></u> Permite obtener varias o ninguna palabra pasando como filtro un nombre que puede contener la palabra, un grupo o un idioma. Estos argumentos son excluyentes. Es decir, puedes filtrar combinándolos entre sí. • <u><i>buscarAleatoriamente(grupo : Grupo, idioma : Idioma) : Palabra[]</i></u> Permite obtener varias palabras o ninguna, ordenadas en un orden distinto cada vez que se llama a este método, filtradas por un grupo y que tengan traducciones al idioma pasado como argumento. • <u><i>buscarAleatoriamente(idioma : Idioma, objetivo : Idioma) : Palabra[]</i></u> Permite obtener varias palabras o ninguna, ordenadas en un orden distinto cada vez que se llama a este método, filtradas por un idioma y que tengan traducciones al idioma objetivo, pasado como argumento. • <u><i>buscarPosiblesTraducciones(palabra : Palabra, idioma : Idioma): Palabra[]</i></u> Permite obtener las posibles traducciones de la palabra pasada como argumento en la fuente de datos, en el idioma pasado como argumento. • <u><i>buscarSinGrupo(grupo : Grupo) : Palabra[]</i></u> Permite obtener todas las palabras que no estén agrupadas en el grupo pasado como argumento.

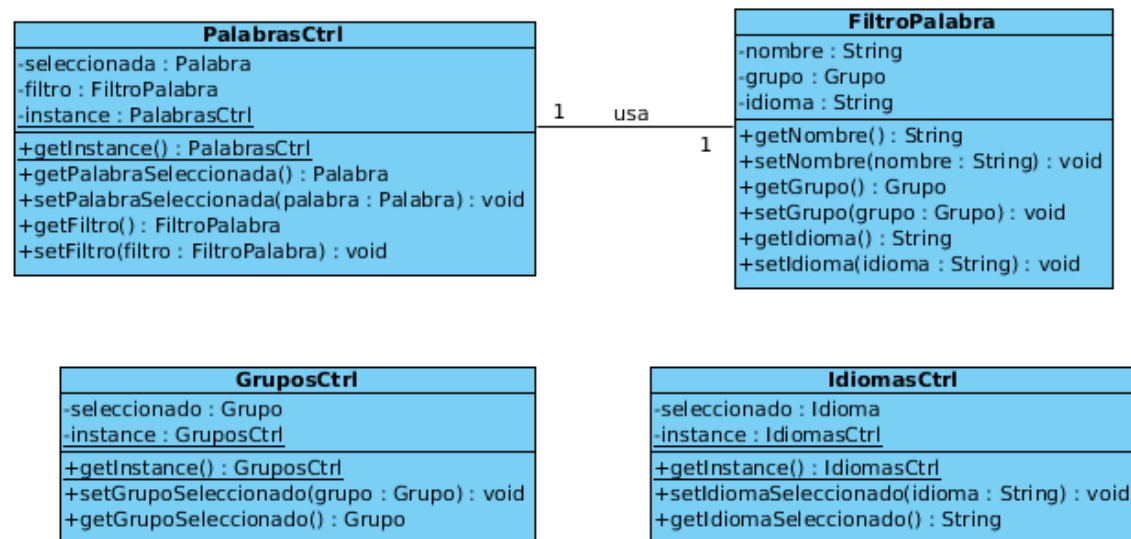
<p>Método</p>	<ul style="list-style-type: none"> • <u><i>actualizarGrupoEliminado(palabra : Palabra, grupo : Grupo) : void</i></u> Permite actualizar el grupo de la palabra en la fuente de datos. • <u><i>actualizarIdioma(palabra : Palabra, idioma_an : String) : void</i></u> Permite actualizar el idioma de la palabra en la fuente de datos, el idioma debe de estar cambiado en el objeto, antes de ejecutar este método. • <u><i>actualizarNombre(palabra : Palabra, nombre_an : String) : void</i></u> Permite actualizar el nombre de la palabra en la fuente de datos, el nombre debe ser cambiado antes de llamar a este método, y se debe pasar el nombre anterior. • <u><i>actualizarNuevaTraduccion(palabra : Palabra, traduccion : Palabra) : void</i></u> Permite actualizar una nueva traducción para la palabra pasada como argumento, en la fuente de datos. • <u><i>actualizarNuevoGrupo(palabra : Palabra, grupo : Grupo) : void</i></u> Permite actualizar un nuevo grupo para la palabra pasada como argumento, en la fuente de datos. • <u><i>actualizarTraduccionEliminada(palabra : Palabra, grupo : Grupo) : void</i></u> Permite actualizar una traducción eliminada de la palabra pasada como argumento, en la fuente de datos. • <u><i>anyadir(palabra : Palabra) : void</i></u> Permite añadir en la fuente de datos la palabra pasada como argumento. • <u><i>borrar(palabra : Palabra) : void</i></u> Permite borrar de la fuente de datos, la palabra pasada como argumento.
----------------------	---

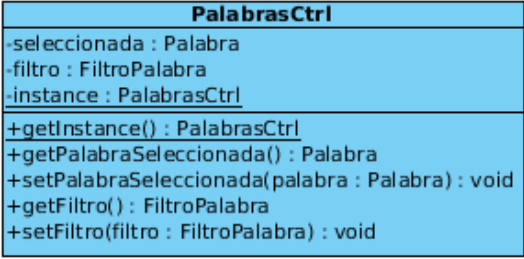
IBDGestionGrupos	
Imagen	<pre> <<Interface>> IBDGestionGrupos +buscar() : Grupo [] +buscar(nombre : String) : Grupo +borrar(grupo : Grupo) : void +anyadir(grupo : Grupo) : void +actualizarNombre(nombre : String, an_nombre : String) : void +actualizarDescripcion(grupo : Grupo) : void +actualizarDesagruparPalabra(grupo : Grupo, palabra : Palabra) : void +actualizarAgruparPalabra(grupo : Grupo, palabra : Palabra) : void </pre>
Descripción	<p>Esta clase agrupa todas las funciones necesarias para obtener información de la fuente de datos, a cerca de los grupos de palabras.</p>
Métodos	<ul style="list-style-type: none"> • <u><i>buscar() : Grupo[]</i></u> Permite conocer todos los grupos escritos en la base de datos. • <u><i>buscar(nombre : String) : Grupo</i></u> Permite obtener el grupo con el nombre pasado como argumento de la fuente de datos, si este existe. • <u><i>borrar(grupo : Grupo) : void</i></u> Permite borrar un grupos mediante su nombre de la fuente de datos. • <u><i>anyadir(grupo : Grupo) : void</i></u> Permite escribir un nuevo grupo en la fuente de datos. • <u><i>actualizarNombre(nombre : String, nombre_an : String) : void</i></u> Permite actualizar en la fuente de datos el nuevo nombre del un grupo. Se tiene que pasar el nuevo nombre y el que tenía. • <u><i>actualizarDescripcion(grupo : Grupo) : void</i></u> Permite actualizar la descripción del grupo en la fuente de datos. Esta descripción debe ser actualizada en el objeto Grupo antes de llamar a este método. • <u><i>actualizarDesagruparPalabra(grupo : Grupo, palabra : Palabra) : void</i></u> Permite actualizar la eliminación de una palabra en el grupo pasado como argumento. • <u><i>actualizarAgruparPalabra(grupo : Grupo, palabra : Palabra) : void</i></u> Permite actualizar en la fuente de datos, que un grupo agrupe a una palabra.

5.1.4.3. Diagrama de clases – controladores.administracion

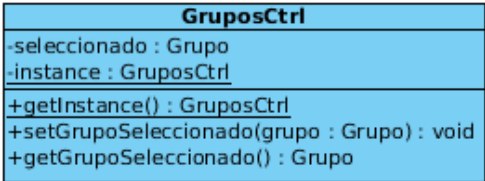
En el siguiente diagrama se muestran las clases *PalabrasCtrl*, *FiltroPalabra*, *GruposCtrl* y *IdiomasCtrl*. Este diagrama rompe un poco las normas del UML, puesto que hay clases que no están relacionadas, pero es que realmente, no lo están. Puse las clases por claridad de qué iba a haber en el programa. Las clases *PalabraCtrl*, *GruposCtrl* y *IdiomasCtrl*, son clases singleton.

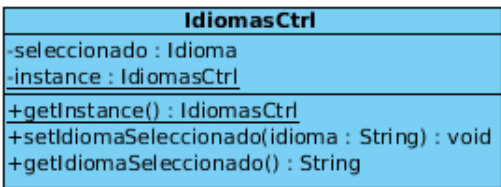
Esto lo hice porque las vistas seguramente serán independientes unas de ellas, así que es útil tener un solo controlador en todo el programa, accesible por todas las vistas. Esto ayuda a que cada vista tenga una restricción del estado del objeto controlador, para realizar su función. Por ejemplo, en la acción de modificar una palabra, tiene que tener el controlador previamente seleccionada la palabra en cuestión.



PalabrasCtrl	
Imagen	 <pre> classDiagram class PalabrasCtrl { -seleccionada : Palabra -filtro : FiltroPalabra -instance : PalabrasCtrl +getInstance() : PalabrasCtrl +getPalabraSeleccionada() : Palabra +setPalabraSeleccionada(palabra : Palabra) : void +getFiltro() : FiltroPalabra +setFiltro(filtro : FiltroPalabra) : void } </pre>
Descripción	Representa el controlador de las palabras, usado por las vistas. Esta clase ayuda a las vistas a llevar un control de por cual transición van.
Atributos	<ul style="list-style-type: none"> • <u><i>seleccionada : Palabra</i></u> Palabra que en está seleccionada, esta será visible por todas las vistas. • <u><i>filtro : FiltroPalabra</i></u> Filtro de las palabras que se están usando, esta será accesible para la vista interesada. • <u><i>instance : PalabrasCtrl</i></u> Instancia de la clase <i>PalabrasCtrl</i> para habilitar el modelo de clases <i>Singleton</i>.
Métodos	<ul style="list-style-type: none"> • <u><i>getInstance() : PalabrasCtrl</i></u> Permite acceder a la clase <i>PalabrasCtrl</i>. • <u><i>getPalabraSeleccionada() : Palabra</i></u> Permite tener acceso a la palabra seleccionada por otra vista. • <u><i>setPalabraSeleccionada(palabra : Palabra) : void</i></u> Permite establecer una nueva palabra seleccionada para las demás vistas. • <u><i>getFiltro() : FiltroPalabra</i></u> Permite obtener el filtro de palabras establecido por algún componente de una vista. • <u><i>setFiltro(filtro : FiltroPalabra) : void</i></u> Permite establecer un filtro de palabras para las demás vistas.
Relaciones	<ul style="list-style-type: none"> • <i>PalabrasCtrl</i> → <i>FiltroPalabra</i> El controlador de palabras también controla el filtro de estas. Por ello, el controlador tiene un filtro de palabras.

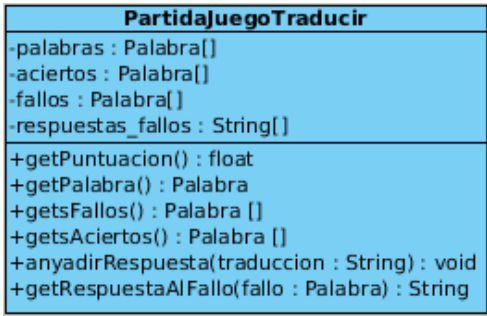
FiltroPalabra											
Imagen	<table><tr><th>FiltroPalabra</th></tr><tr><td>-nombre : String</td></tr><tr><td>-grupo : Grupo</td></tr><tr><td>-idioma : String</td></tr><tr><td>+getNombre() : String</td></tr><tr><td>+setNombre(nombre : String) : void</td></tr><tr><td>+getGrupo() : Grupo</td></tr><tr><td>+setGrupo(grupo : Grupo) : void</td></tr><tr><td>+getIdioma() : String</td></tr><tr><td>+setIdioma(idioma : String) : void</td></tr></table>	FiltroPalabra	-nombre : String	-grupo : Grupo	-idioma : String	+getNombre() : String	+setNombre(nombre : String) : void	+getGrupo() : Grupo	+setGrupo(grupo : Grupo) : void	+getIdioma() : String	+setIdioma(idioma : String) : void
FiltroPalabra											
-nombre : String											
-grupo : Grupo											
-idioma : String											
+getNombre() : String											
+setNombre(nombre : String) : void											
+getGrupo() : Grupo											
+setGrupo(grupo : Grupo) : void											
+getIdioma() : String											
+setIdioma(idioma : String) : void											
Descripción	Representa una agrupación de atributos de la clase <i>Palabra</i> para filtrarlas. Estos atributos son <i>nombre</i> , <i>grupo</i> y <i>idioma</i> . Estos pueden ser nulos, es decir, que si se quiere filtrar por nombre solamente, se establecerán el grupo y el idioma a <i>NULL</i> .										
Atributos	<ul style="list-style-type: none">• <u><i>nombre : String</i></u> Almacena el nombre por el cual se va a filtrar la palabra.• <u><i>grupo : Grupo</i></u> Almacena el grupo por el cual se va a filtrar la palabra.• <u><i>idioma : Idioma</i></u> Almacena el idioma por el cual se va a filtrar la palabra.										
Métodos	<ul style="list-style-type: none">• <u><i>getNombre() : String</i></u> Permite obtener el nombre de la palabra filtrada.• <u><i>setNombre(nombre : String) : void</i></u> Permite establecer un nuevo nombre al filtro de palabras o <i>NULL</i>.• <u><i>getGrupo() : Grupo</i></u> Permite obtener el grupo de la palabra filtrada.• <u><i>setGrupo(grupo : Grupo) : void</i></u> Permite tener acceso al grupo de la palabra filtrada o <i>NULL</i>.• <u><i>getIdioma() : Idioma</i></u> Permite obtener el idioma del filtro de palabras.• <u><i>setIdioma(idioma : Idioma) : void</i></u> Permite establecer un nuevo idioma al filtro de las palabras o <i>NULL</i>.										
Relaciones	<ul style="list-style-type: none">• <i>FiltroPalabra</i> → <i>PalabrasCtrl</i> El filtro de las palabras es controlador por un solo controlador de las palabras (<i>PalabrasCtrl</i>).										

GruposCtrl	
Imagen	 <pre> classDiagram class GruposCtrl { -seleccionado : Grupo -instance : GruposCtrl +getInstance() : GruposCtrl +setGrupoSeleccionado(grupo : Grupo) : void +getGrupoSeleccionado() : Grupo } </pre>
Descripción	Representa el controlador de los grupos, usado por las vistas. Esta clase ayuda a las vistas a llevar un control de por cual transición van.
Atributos	<ul style="list-style-type: none"> • <u>seleccionado : Grupo</u> Grupo que en está seleccionado, este será visible por todas las vistas. • <u>instance : GruposCtrl</u> Instancia de la clase <i>GruposCtrl</i> para habilitar el modelo de clases <i>Singleton</i>.
Métodos	<ul style="list-style-type: none"> • <u>getInstance() : PalabraCtrl</u> Permite acceder a la clase <i>PalabrasCtrl</i>. • <u>getGrupoSeleccionada() : Grupo</u> Permite tener acceso a al grupo seleccionado por otra vista. • <u>setGrupoSeleccionada(grupo : Grupo) : void</u> Permite establecer un nuevo grupo seleccionado para las demás vistas.

IdiomasCtrl	
Imagen	 <pre> classDiagram class IdiomasCtrl { -seleccionado : Idioma -instance : IdiomasCtrl +getInstance() : IdiomasCtrl +setIdiomaSeleccionado(idioma : String) : void +getIdiomaSeleccionado() : String } </pre>
Descripción	Representa el controlador de los idiomas, usado por las vistas. Esta clase ayuda a las vistas a llevar un control de por cual transición van.
Atributos	<ul style="list-style-type: none"> • <u>seleccionado : Idioma</u> Idioma que en está seleccionado, este será visible por todas las vistas. • <u>instance : IdiomasCtrl</u> Instancia de la clase <i>IdiomasCtrl</i> para habilitar el modelo de clases <i>Singleton</i>.
Métodos	<ul style="list-style-type: none"> • <u>getInstance() : IdiomasCtrl</u> Permite acceder a la clase <i>IdiomasCtrl</i>. • <u>getIdiomaSeleccionada() : Grupo</u> Permite tener acceso a al idioma seleccionado por otra vista. • <u>setIdiomaSeleccionada(idioma : String) : void</u> Permite establecer un nuevo idioma seleccionado para las demás vistas.

5.1.4.4. Diagrama de clases – cotroladores.juego

En el diagrama de controladores de juego, puesto que solamente hay un juego; hay solo una clase. Esta clase controla el juego de traducir palabras.

PartidaJuegoTraducir	
Imagen	 <pre> classDiagram class PartidaJuegoTraducir { -palabras : Palabra[] -aciertos : Palabra[] -fallos : Palabra[] -respuestas_fallos : String[] +getPuntuacion() : float +getPalabra() : Palabra +getsFallos() : Palabra [] +getsAciertos() : Palabra [] +anyadirRespuesta(traduccion : String) : void +getRespuestaAlFallo(fallo : Palabra) : String } </pre>
Descripción	<p>Representa el controlador de una partida de traducir palabras. Este controlador permite llevar el control de las palabras que se están jugando, cuántos aciertos lleva el usuario, cuántos fallos lleva, qué puso cuando falló el usuario y su puntuación.</p> <p>La partida termina cuando el método <i>getPalabra()</i>, devuelve <i>NULL</i>.</p>
Atributos	<ul style="list-style-type: none"> • <u><i>palabras : Palabra[]</i></u> Almacena las palabras que se juegan en la partida. • <u><i>aciertos : Palabra[]</i></u> Almacena las palabras acertadas en la partida. • <u><i>fallos : Palabra[]</i></u> Almacena las palabras falladas en la partida. • <u><i>respuestas_fallos : String[]</i></u> Almacena las cadenas que ha introducido el usuario cuando ha fallado.
Métodos	<ul style="list-style-type: none"> • <u><i>getPuntuacion() : float</i></u> Permite obtener la puntuación de la partida, esta es correcta cuando la partida termina. Si no, dará una puntuación incorrecta. • <u><i>getPalabra() : Palabra</i></u> Permite conocer la siguiente palabra para el usuario, o la primera. Si devuelve <i>NULL</i> significa que ya ha terminado la partida. Esta no cambiará hasta que no se añada una respuesta. • <u><i>getsFallos() : Palabra[]</i></u> Permite obtener todos los fallos que ha tenido el usuario.

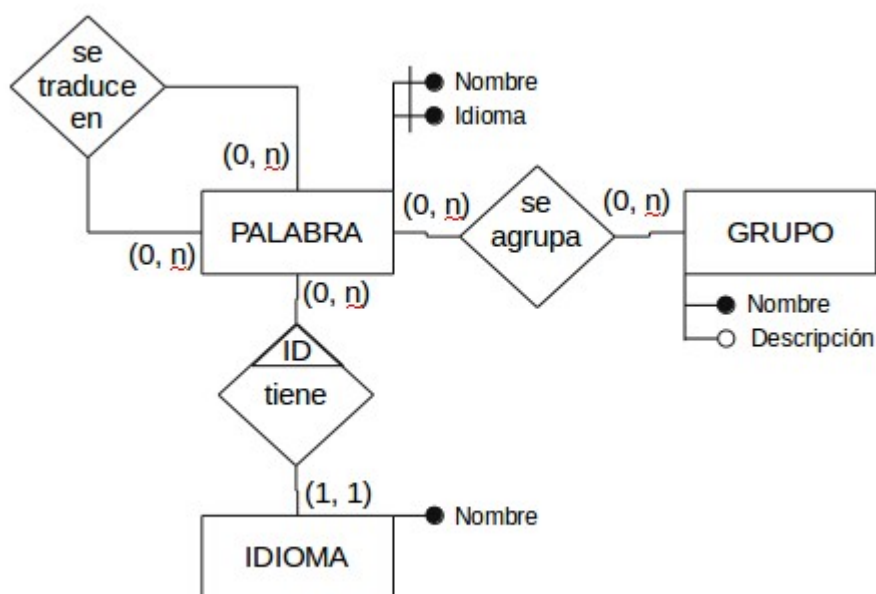
Métodos	<ul style="list-style-type: none">• <u><i>getsAciertos() : Palabra[]</i></u> Permite obtener todos los aciertos del usuario.• <u><i>anyadirRespuesta(traduccion : String) : void</i></u> Permite añadir la respuesta del usuario, para que sea evaluada por la clase. Cuando se llama a este método, la clase pasa a la siguiente palabra. Por lo que <i>getPalabra()</i> devolverá la siguiente, o <i>NULL</i>.• <u><i>getRespuestaAlFallo(fallo : Palabra) : String</i></u> Permite conocer la respuesta del usuario a un palabra que falló. Esto es útil para mostrar más información al usuario.
----------------	--

5.2. Diseño de la base de datos.

La base de datos de este programa es bastante pequeña, en respecto al número de tablas. Puesto que este diseño se basa en la información que queremos hacer persistente; en este caso, este diagrama se saca del diagrama de clases del paquete *modelo*. Así que encontraremos tres tablas relacionadas; *palabra*, *grupo* e *idioma*.

En este apartado, mostraré el diagrama de entidad-relación, el diagrama relacional, una tabla con las propiedades de las claves foráneas y una tabla del nombre de los triggers, y qué hacen.

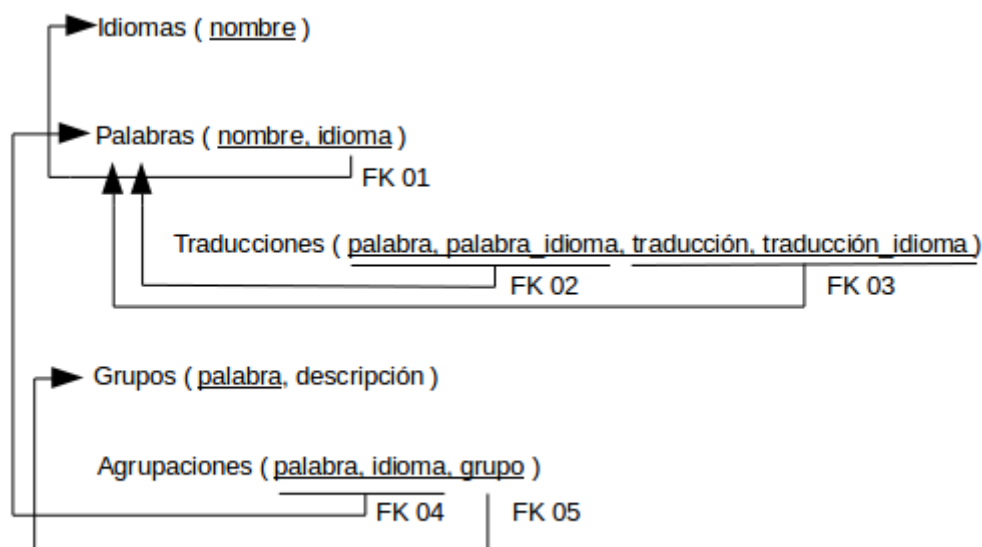
El diagrama de E/R es el siguiente:



La descripción del diagrama de E/R es el siguiente; un idioma puede tener alguna palabra o ninguna, puesto que puede haber un idioma creado sin palabras aun. Las palabras a su vez, solamente podrá tener un idioma, se podrá agrupar en más de un grupo, o en ninguno; y se podrá traducir en varias palabras de distinto idioma. Por último, un grupo podrá agrupar a varias palabras, o a ninguna.

Todos los atributos que aparecen son obligatorios, excepto la descripción del grupo. Esta no será necesaria, puesto que a veces, con un nombre del grupo es suficiente.

El diagrama relacional del E/R es el siguiente:



La tabla de las claves foráneas es la siguiente:

Clave foránea	Nullable	On update	On delete
FK 01	No	Cascade	Cascade
FK 02	No	Cascade	Cascade
FK 03	No	Cascade	Cascade
FK 04	No	Cascade	Cascade
FK 05	No	Cascade	Cascade

Hay varios *triggers* en la base de datos importantes, estos son:

- Un *trigger* para que cuando una palabra cambia de idioma, se borren sus traducciones. Puesto que no tiene sentido que halla.
- Un *trigger* para que rellene la traducción cruzada. Es decir, si se introduce que A se traduzca a B, se escribe un registro más en la que B se traduce a A.
- Un *trigger* para que se rellene la traducción transitiva. Es decir, si A se traduce a B y a C, si B se traduce a A, también se traduce a C. Siendo A, B y C de distinto idioma.
- Un *trigger* para controlar que no se introduzcan traducciones con el mismo idioma.

5.3. Codificación

En este apartado del documento explicaré qué lenguaje he utilizado en el proyecto, qué sistema gestor de base de datos he utilizado, estructura de ficheros y algunos algoritmos importantes del programa.

5.3.1. Lenguaje de programación y IDE utilizado

He elegido el lenguaje de programación Java. Ya que es el lenguaje que mayoritariamente he aprendido, y me permite hacer un programa multiplataforma. Aunque en opinión personal, este lenguaje consume muchos recursos, puesto que tiene que dar soporte multiplataforma; yo lo hubiera hecho en C++. Pero eso puede ser también una actualización del programa si el programa consumiera muchos recursos.

Las librerías utilizadas han sido:

- El JDK de java, versión 8.
- Las librerías de SWING, para el entorno gráfico.
- La librería de SQLite, versión 3.8.7.
- La librería JDBC de Java para tratar las bases de datos.

Para implementar el programa he utilizado el IDE Netbeans. Ya que considero que es muy versátil y permite realizar programas bastante rápido, debido a las herramientas que ofrece.

5.3.2. SGDB utilizado.

He elegido el SGDB embebido SQLite en sus versión 3.8.7.

¿Por qué SQLite? Bueno, me parece que es un SGBD muy ligero, permite hacer de forma fácil los *triggers*, tiene poco tiempo de acceso, aunque no sea concurrente y soporta todos los tipos de datos necesitados por el proyecto. Otro SGBD embebida que podría haber utilizado podría ser *Derby*. Pero, por experiencia, funciona un poco lento en Linux.

¿Por qué la versión 3.8.7?, por que en esta versión soporta las restricciones de claves foráneas, mediante la ejecución al conectarse '*PRAGMA foreign_key = ON*'. No hace falta que afirme, que esto es muy útil para el programador. Ya que de otro modo, el programador tendría que crear los *triggers* necesarios para dar soporte a esta tecnología.

Las sentencias SQL del DDL de la base de datos se encuentran en el fichero `/src/lanword/resources/sqlite_ddl.sql`. Picha aquí para ir al anexo con el código.

Las sentencias SQL del DML de la base de datos, se encuentran en las clases del paquete `lanword.interfaces.bd.sqlite.*`.

5.3.3. Estructura de ficheros del proyecto.

Todo el código fuente se encuentra en la carpeta /src y contiene lo siguiente:

- gui

Contiene todo el código que tiene relación con la interfaz gráfica y el *main* de la interfaz gráfica.

➔ Administración

Todo el código que contiene la ventana de administración.

➔ Componentes

Todas las clases comunes por varias vistas, como *combobox* que muestran grupos, idiomas, etc.

➔ images

Contiene las Imágenes de la interfaz gráfica.

➔ Juegos

Contiene todo el código de la ventana de juegos.

➔ Util

Clases útiles para la interfaz, como el gestor de paneles.

- Lanword

Núcleo del programa, contiene el código explicado en los apartados anteriores. Ya que contiene las clases que forman el dominio del problema.

➔ Controladores

Contiene dos paquetes, cada uno diseñado para guardar información entre controladores de más alto nivel, como los *Listener* de una interfaz gráfica.

- Administración

Contiene las clases que guardan información sobre los grupos, idiomas y palabras.

- Juegos

Contiene la clase que guarda el progreso del juego de traducir palabras.

➔ Interfaces/bd

Contiene las interfaces y la clase abstracta que deberán definirse para cada SGDB de Lanword.

- Sqlite

Contiene las clases derivadas de la interfaz para tratar la conexión con un SGDB de Sqlite.

➔ Modelo

Contiene las clases más importantes del programa. Ya que todo el programa gira en torno a las palabras, grupos e idiomas. Además de la clase *ClasePersistente*, que hace de caché en el programa.

➔ Resources

Ficheros necesarios para el modelo, como las sentencias DDL de la base de datos.

5.3.4. Algoritmos utilizados.

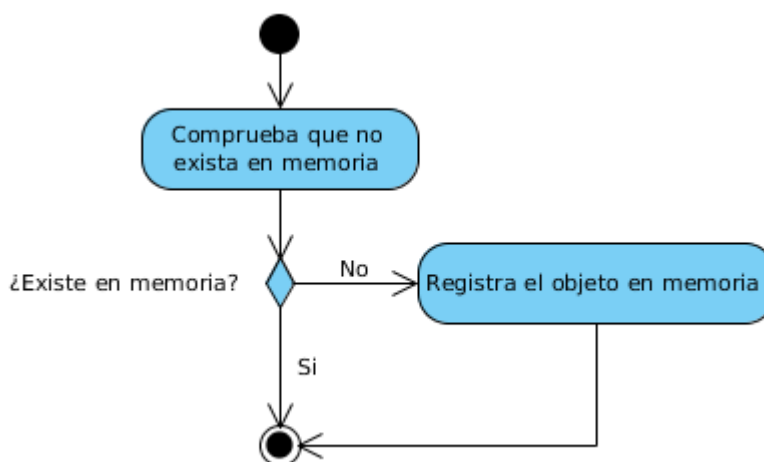
Los algoritmos utilizados en este programa son muy convencionales, excepto uno que maneja la caché del programa. Este algoritmo o metodología de tratar los objetos persistentes del programa, la he implementado por primera vez en este programa, y debe de tratarse como procedimiento obligatorio en la consulta, creación y eliminación de objetos que derivan de la clase *ObjetoPersistente*.

Este algoritmo se aplica directamente a los objetos de las clases *Idioma*, *Palabra* y *Grupo*, y usan los métodos de la clase *ObjetoPersistente* la interfaz de la base de datos. Hay dos tipos de persistencia en la clase *ObjetoPersistente*, la que está en memoria y la que está escrita en el fuente de datos. Para los dos casos, hay tres algoritmos, uno de búsqueda, eliminación y creación. En total son seis algoritmos, que sus procedimientos son:

- **Creación de objetos en memoria.**

En este caso el programa tiene que buscar en memoria si existe el objeto, si no existe; se registra en memoria. Si existe, no se hace nada.

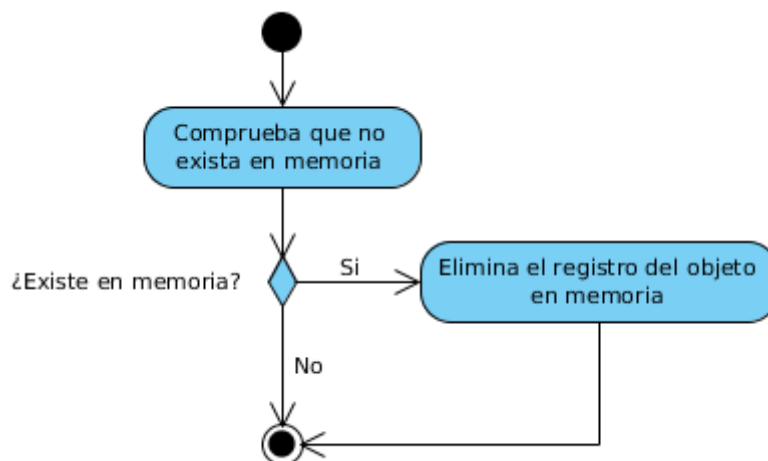
Este procedimiento se puede ver en el siguiente diagrama:



- **Borrados de objetos en memoria.**

En este caso el programa tiene que eliminar registro de la memoria si existe el objeto, si no existe; no hace nada. Si existe, lo elimina del registro.

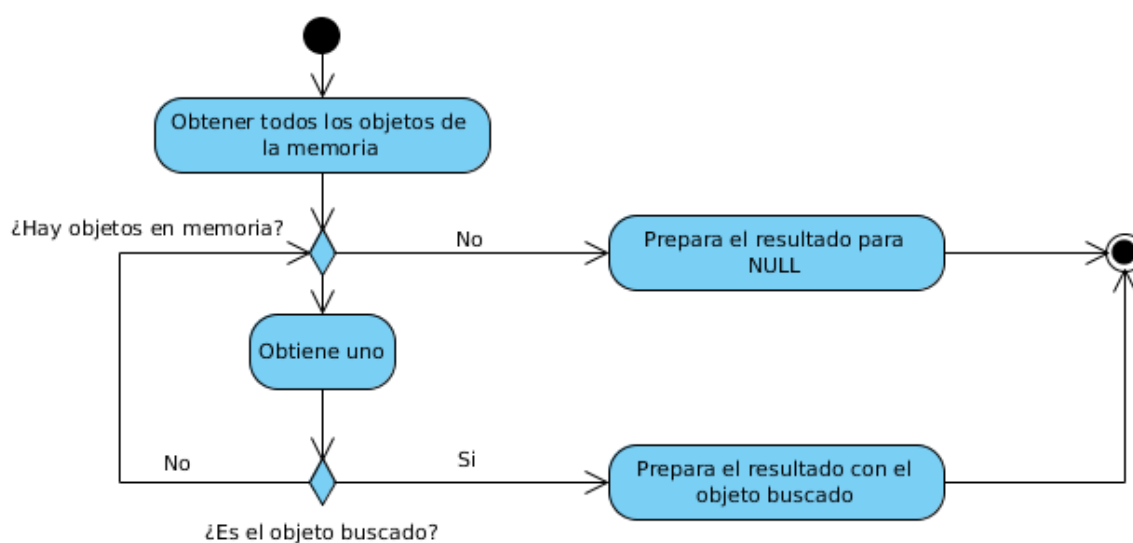
Este procedimiento se puede ver en el siguiente diagrama:



- **Búsqueda de objetos en memoria.**

Busca entre todos los objetos en memoria, uno. Para ello, accede a todos y los recorre iterativamente hasta encontrarlo, o no.

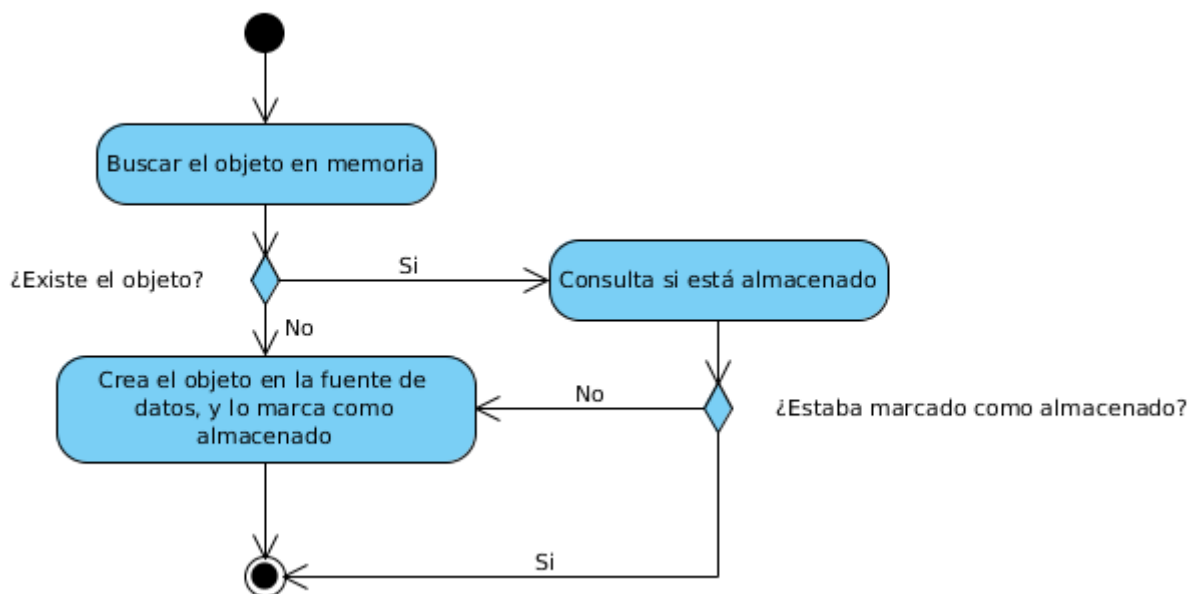
Este procedimiento se puede ver en el siguiente diagrama:



- **Creación de objetos en la fuente de datos.**

En este caso el programa tiene que buscar en la memoria el objeto, tiene que asegurarse de que el objeto no está marcado como almacenado, y si no está marcado como almacenado lo crea en la fuente de datos.

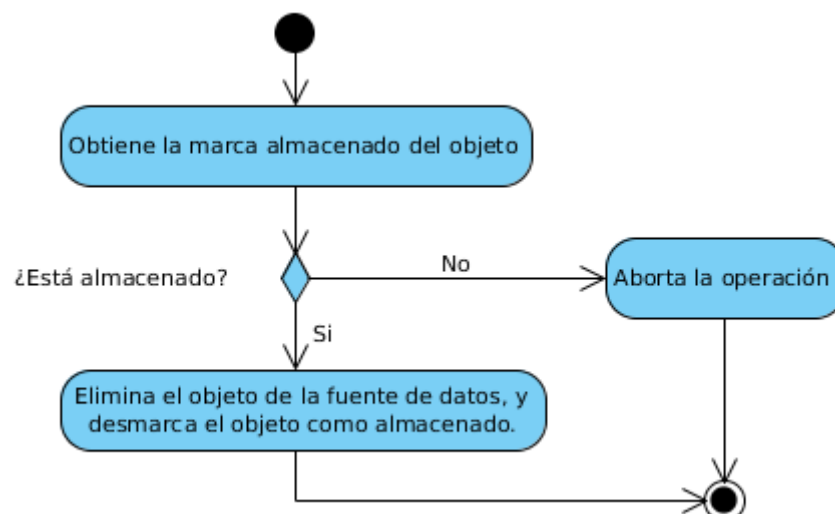
Este procedimiento se puede ver en el siguiente diagrama:



- **Borrados de objetos en la fuente de datos.**

En este caso, mira si está almacenado el objeto y si sí esta, se borra, si no, se aborta la operación.

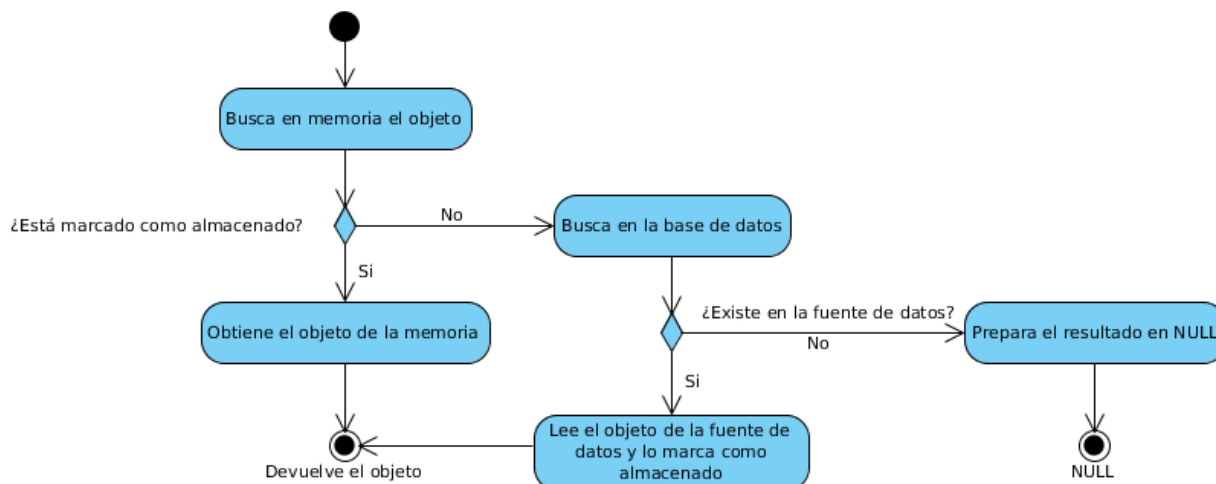
Este procedimiento se puede ver en el siguiente diagrama:



- **Búsqueda de objetos en al fuente de datos.**

En la operación de búsqueda, se tiene que mirar primero en la memoria, por si existiera el objeto. Si existe, hay que ver si está marcado como almacenado. Si esta marcado, no busca en la base de datos, puesto que ya existe en memoria; si no lo esta, lo busca. Si lo encuentra, lo crea y lo marca como almacenado, y si no, prepara el resultado como nulo.

Este procedimiento se puede ver en el siguiente diagrama:



Con estos procedimientos de búsqueda, creación y eliminación; se solventa el problema de tener dos objetos en memoria iguales para la base de datos. Como se puede observar en los diagramas, cada vez que se intenta crear un objeto de exista ya, no se registra en memoria. Por lo que si un tercero quiere encontrarlo, se le dará la estancia del registro y no la del duplicado. El duplicado al no tener instancia guarda en otro objeto, sera desechado por el *GarbageCollector* de Java. Esta misma filosofía se usa a la hora de borrar un objeto de memoria.

Estos procedimientos también ayudan a la base de datos a consultar menos, ya que si existe un objeto en memoria, no consulta la base de datos.

Nota: Cuando en las explicaciones se habla de “marcar como almacenado”. Se refiere al atributo de la clase *ClasePersistente*, *stored*. Es booleano y guarda el valor de marca.

Nota: El registro de memoria, es un *array* estático en la clase *ClasePersistente*. Tiene el método estático *buscar()*, que es utilizado para tal fin.

6. Anexos.

6.1. Sentencias SQL del DDL.

```
PRAGMA foreign_keys = ON;

CREATE TABLE idiomas (
    nombre TEXT PRIMARY KEY
);

CREATE TABLE palabras (
    nombre TEXT,
    idioma TEXT,
    PRIMARY KEY(nombre, idioma),
    FOREIGN KEY(idioma) REFERENCES idiomas(nombre)
        ON DELETE CASCADE
        ON UPDATE CASCADE
);

CREATE TABLE traducciones (
    palabra TEXT,
    idioma_palabra TEXT,
    traduccion TEXT,
    idioma_traduccion TEXT,
```

```

PRIMARY KEY (palabra, idioma_palabra,
             idioma_traduccion, traduccion),

FOREIGN KEY(palabra, idioma_palabra)

REFERENCES palabras(nombre, idioma)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (traduccion, idioma_traduccion)

REFERENCES palabras(nombre, idioma)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE grupos (

nombre TEXT,

descripcion TEXT,

PRIMARY KEY(nombre)

);

CREATE TABLE agrupaciones (

palabra TEXT,

idioma TEXT,

grupo TEXT,

PRIMARY KEY(palabra, idioma, grupo),

FOREIGN KEY(palabra, idioma) REFERENCES palabras(nombre, idioma)

ON DELETE CASCADE ON UPDATE CASCADE,

```

```
FOREIGN KEY(grupo) REFERENCES grupos(nombre)

    ON DELETE CASCADE

    ON UPDATE CASCADE

);

-- Trigger para comprobar que no halla dos traducciones con el mismo
-- idioma

CREATE TRIGGER check_traduccion

BEFORE INSERT ON traducciones

FOR EACH ROW BEGIN

    SELECT RAISE(ABORT, 'No se puede traducir una palabra al mismo

                    idioma.')

    WHERE NEW.idioma_palabra = NEW.idioma_traduccion;

END;
```



```
-- Trigger para rellenar las traducciones cruzadas y las transitivas.
CREATE TRIGGER rellenar_traduccion
AFTER INSERT ON traducciones
FOR EACH ROW BEGIN

    INSERT INTO traducciones VALUES (

        NEW.traduccion, NEW.idioma_traduccion, NEW.palabra,

        NEW.idioma_palabra

    );

    INSERT INTO traducciones

    SELECT NEW.traduccion, NEW.idioma_traduccion, traduccion,

        idioma_traduccion

    FROM traducciones

    WHERE palabra = NEW.palabra AND traduccion <> NEW.traduccion

    UNION

    SELECT traduccion, idioma_traduccion, NEW.traduccion,

        NEW.idioma_traduccion

    FROM traducciones

    WHERE palabra = NEW.palabra AND traduccion <> NEW.traduccion;

END;
```

```
-- Trigger para quitar las traducciones cruzadas.

CREATE TRIGGER quitar_traduccion_cruzada
AFTER DELETE ON traducciones
FOR EACH ROW BEGIN

    DELETE FROM traducciones

    WHERE palabra = OLD.traduccion AND traduccion = OLD.palabra AND

        idioma_palabra = OLD.idioma_traduccion AND

        idioma_traduccion = OLD.idioma_palabra;

END;

-- Trigger que borra las traducciones cuando se actualiza un idioma.

CREATE TRIGGER update_idioma_remove_traducciones
AFTER UPDATE OF idioma ON palabras
FOR EACH ROW WHEN OLD.idioma <> NEW.idioma
BEGIN

    DELETE FROM traducciones

    WHERE palabra = OLD.nombre AND idioma_palabra = OLD.idioma;

END;
```