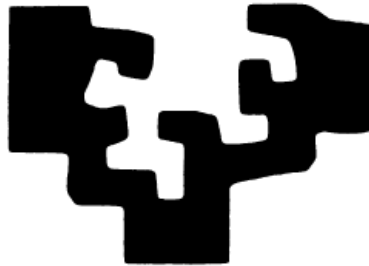


eman ta zabal zazu



universidad
del país vasco

euskal herriko
unibertsitatea

Facultad de Informática / Informatika Fakultatea

Android app para Multilingual Central Repository

Alumno/a: D./Dña. Aritza Ledesma Ruiz

Director/a: D./Dña. Germán Rigau Claramunt

Proyecto Fin de Carrera, junio 2014

Índice

1	Introducción	1
1.1	Presentación	1
1.2	Motivación	1
2	Dop	5
2.1	Objetivos	5
2.2	Método de trabajo.....	5
2.3	Alcance.....	7
2.3.1	Recursos humanos.....	7
2.3.2	Recursos materiales.....	7
2.3.3	Recursos económicos	7
2.4	EDT	8
2.5	Procesos.....	10
2.5.1	Procesos tácticos	10
2.5.2	Procesos formativos	10
2.5.3	Procesos operativos.....	12
2.5.4	Análisis de factibilidad	13
2.6	Planificación temporal: Diagrama de Gantt	13
2.7	Plan de contingencia.....	15
2.7.1	Riesgos Tácticos.....	15
2.7.2	Riesgos Operativos	16
2.7.3	Riesgos Formativos.....	17
3	Antecedentes.....	19
3.1	WordNet	19
3.2	EuroWordNet.....	23
3.3	Open Multilingual Wordnet.....	23
3.4	Multilingual Central Repository (MCR).....	25
3.5	Multilingual Central Repository 3.0	26
3.6	Web EuroWordNet Interface.....	26
3.7	Aplicaciones alternativas	28
3.7.1	ColorDict Diccionario	28

3.7.2	Free Dictionary Org.....	29
4	Elección de la tecnología	31
4.1	Android	31
4.1.1	Análisis de mercado.....	31
4.1.2	Versión de Android.....	32
4.2	Resto de tecnología	33
4.2.1	Sistema operativo.....	33
4.2.2	Entorno de desarrollo.....	33
4.2.3	PhpMyAdmin	34
4.2.4	SQLitestudio.....	34
4.2.5	PHP.....	34
4.2.6	JSON.....	35
4.2.7	Documentación	35
5	Captura de requisitos	37
5.1	Diseño de pantallas.....	38
5.2	Diagrama de casos de uso MCRA 3.0.....	40
5.3	Casos de uso reales.....	41
5.4	Diagrama de casos de uso MCRBD 3.0	47
5.4.1	Caso de uso reales	47
5.5	Modelo de datos.....	49
5.5.1	Base de datos MCR 3.0	49
5.5.2	Nueva base de datos	53
6	Análisis.....	57
6.1	Cotratos.....	59
6.1.1	Obtener significados.....	59
6.1.2	Obtener traducción	60
6.1.3	Obtener sinónimos	61
6.1.4	Obtener antónimos	62
6.1.5	Obtener hipónimos.....	63
6.1.6	Obtener hiperónimo.....	64
6.1.7	Instalar base de datos.....	65
7	Diseño.....	67

7.1	Diagramas de secuencia	67
7.1.1	Obtener significado	68
7.1.2	Obtener traducción	69
7.1.3	Obtener sinónimos, antónimos, hiperónimos o hipónimos	70
7.2	Base de datos online	71
7.2.1	Comunicación con la base de datos externa	71
7.3	Base de datos offline.....	72
7.3.1	Obtener significado	72
7.3.2	Obtener traducción	72
7.3.3	Obtener sinónimos/antónimos/hiperónimos/hipónimos	73
7.4	Comprobar conexión	73
7.5	Crear base de datos	74
8	Implementación.....	75
8.1	Android SDK	75
8.2	Interfaz.....	76
8.3	Activity	77
8.4	Estructura de un proyecto Android	79
8.4.1	Carpeta src.....	80
8.4.2	Carpeta res.....	80
8.4.3	Carpeta bin	81
8.4.4	Carpeta assets.....	81
8.4.5	Carpeta libs.....	81
8.4.6	Archivo AndroidManifest.xml.....	81
8.5	Estructura MCRA 3.0.....	83
8.5.1	ActivityMain.xml.....	83
8.5.2	MainActivity.java	83
8.5.3	ActivityTwo.xml	84
8.5.4	ActivityTwo.java	84
8.5.5	Palabra.java	84
8.5.6	Custom_bg_2.xml.....	84
8.5.7	DataBaseOnline.java.....	84
8.5.8	DataBaseOffline.java	85

8.5.9	String.xml.....	85
8.5.10	Pantalla	85
8.6	Estructura MCRBD 3.0	87
8.6.1	Interfaz Main Activity.xml	87
8.6.2	Main Activity.java	88
8.6.3	String.xml.....	88
8.6.4	Pantalla	88
9	Pruebas.....	91
9.1	Pruebas Unitarias.....	91
9.1.1	Comprobar conexión	91
9.1.2	Creación del menú de opciones y cambio de idiomas	92
9.1.3	Obtener traducciones offline	92
9.1.4	Obtener significados offline	93
9.1.5	Obtener sinónimos/antónimos/hipónimos/hiperónimos offline	94
9.1.6	Obtener traducciones online.....	94
9.1.7	Obtener significados online.....	95
9.1.8	Obtener sinónimos/antónimos/hipónimos/hiperónimos online	95
9.2	Pruebas de integración y sistema	96
9.2.1	Consultar significado de una palabra offline.....	96
9.2.2	Consultar traducción de una palabra offline.....	97
9.2.3	Consultar sinónimos/antónimos/hipónimos/hiperónimos offline	97
9.2.4	Consultar significado de una palabra online	98
9.2.5	Consultar traducción de una palabra online	98
9.2.6	Consultar sinónimos/antónimos/hipónimos/hiperónimos online	99
10	Implantación.....	101
11	Gestión del proyecto	103
11.1	Comparativa entre esfuerzo planificado y real.....	103
11.1.1	Procesos tácticos	103
11.1.2	Procesos formativos	104
11.1.3	Procesos operativos.....	105
11.2	Comparativa de las horas totales planificadas frente a las reales.....	108
11.3	Justificación de las desviaciones	109

11.4	Incidencias.....	109
12	Conclusiones.....	111
12.1	Objetivos cumplidos.....	111
12.2	Trabajos futuros	112
12.2.1	Alojamiento en internet	112
12.2.2	Actualizaciones	112
12.2.3	Otros sistemas operativos	112
12.2.4	Ampliar base de datos	112
13	Bibliografía.....	113

1 Introducción

1.1 Presentación

Este documento es la memoria de un Proyecto de Fin de Carrera de Ingeniería en Informática, en la Facultad de Informática de San Sebastián, que se encuadra en el desarrollo de una aplicación Android usando una base de conocimiento multilingüe. Está realizado por Aritza Ledesma Ruiz y dirigido por Germán Rigau Claramunt, profesor asociado en la Facultad de Informática de San Sebastián, Universidad Pública Vasca (UPV/EHU).

1.2 Motivación

Este proyecto surge de la necesidad de llevar la base de conocimiento multilingüe *Multilingual Central Repository 3.0* (MCR 3.0)¹, a dispositivos móviles para una mayor accesibilidad. El MCR 3.0 está compuesto por diferentes bases de datos léxicas denominadas *wordnet*. Cada *wordnet* contiene información codificada manualmente sobre nombres, verbos, adjetivos y adverbios, las cuales están organizadas entorno a la noción de *synset*. Un *synset* es un conjunto de palabras de la misma categoría morfosintáctica que pueden ser intercambiados en un contexto dado y que están enlazados entre ellos mediante relaciones léxicas y semántico-conceptuales. El MCR 3.0 dispone de cinco *wordnet* para los siguientes idiomas: inglés, español, euskera, catalán y gallego, que nos permiten obtener todas las relaciones de las palabras, sus significados, su tipo y sus ejemplos. Actualmente disponemos de una interfaz web, *Web EuroWordNet Interface* (WEI)², que nos permite consultar la información contenida en el MCR 3.0.

El objetivo de nuestra aplicación es mostrar, en la medida de lo posible, la misma información que obtenemos con el MCR 3.0. Toda la información que mostrará nuestra aplicación estará disponible en cinco idiomas diferentes, al igual que la versión online. Más concretamente, nos centramos en las siguientes opciones:

¹ <http://adimen.si.ehu.es/web/MCR>

² <http://adimen.si.ehu.es/cgi-bin/wei/public/wei.consult.perl>

- Obtener significados.
- Obtener traducciones.
- Obtener sinónimos, antónimos, hipónimos³ o hiperónimos⁴.

Antes de seleccionar alguna de las opciones, será necesario seleccionar el idioma de la palabra que queremos consultar y el idioma en el que queremos que nos muestra la información. Obtener significados, nos mostrará los diferentes significados, ejemplos, tipo, (indicando si dicha respuesta está asociada a una palabra de tipo nombre, verbo, adjetivo o adverbio) y palabras asociados a la palabra que queremos consultar. Obtener traducciones nos mostrará el tipo y las diferentes palabras asociadas a la palabra introducida en el idioma seleccionado. Obtener sinónimos, antónimos, hipónimos o hiperónimos nos mostrara el tipo y las palabras asociadas a dicha opción. Para tener una noción básica del funcionamiento, expondremos una breve explicación:

El usuario indicará el idioma de origen de la palabra que está introduciendo, el idioma en que queremos que nos muestre el resultado y la operación que queremos realizar (traducciones o significados). Tras elegir la operación, el usuario pulsará el botón de enviar y se mostrará por pantalla el resultado de su operación. Además, si el usuario pulsa uno de los significados o traducciones obtenidas, se le desplegará un menú en el que se mostrará las operaciones de obtener: sinónimos, antónimos, hipónimos e hiperónimos. Si elegimos alguna de las operaciones, aparecerá una nueva pantalla con todos los posibles resultados asociados a esa palabra. Para ilustrar un poco más esta explicación, presentamos un pequeño ejemplo de una solución:

Por ejemplo, el usuario quiere conocer el significado de la palabra “beautiful”, para ello seleccionará el inglés como idioma de origen de la palabra, e inglés como destino; y, por último, selecciona la opción de obtener significados. En la figura 1, podemos observar gráficamente el resultado de dichas consultas.

³ **Hiponimia:** es el término específico usado para designar el miembro de una clase, X es un hipónimo de Y, si X es una clase de Y. En el caso de los verbos se denomina Troponimia.

⁴ **Hiperonimia:** es el término genérico usado para designar a una clase de instancias específicas. Y es un hiperónimo de X, si X es una clase de Y.

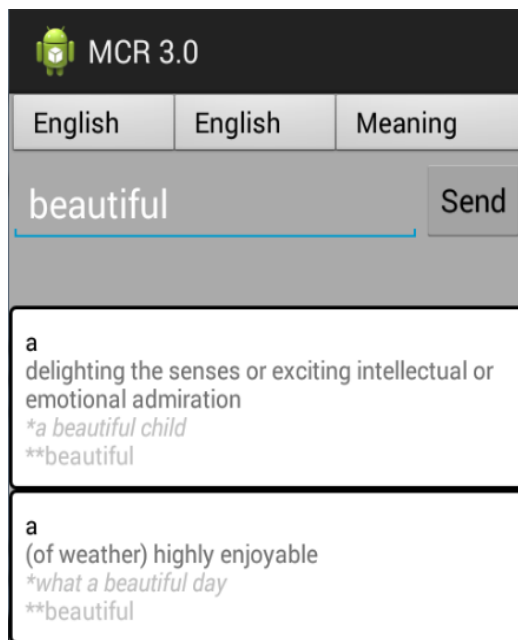


Figura 1: Resultado de la búsqueda "beautiful".

Si ahora queremos conocer los sinónimos de esa palabra, seleccionamos la primera respuesta y, en el menú que nos aparece, la opción de sinónimos. Tras realizar dicho proceso, nos aparecerán los resultados de la figura 2:

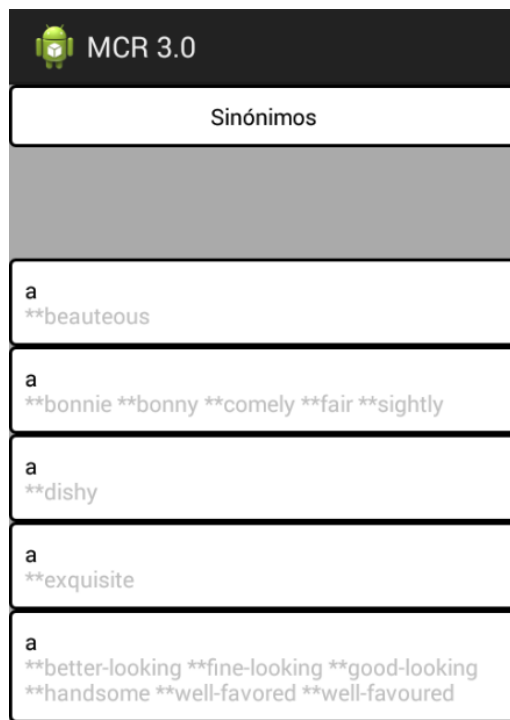


Figura 2: Sinónimos de la palabra "beautiful".

Este es el funcionamiento que tendrá nuestra aplicación, siendo muy similar a la que obtenemos en la versión online.

Este documento se centra en explicar el proceso seguido para el desarrollo de la aplicación, la tecnología seleccionada y el funcionamiento de la aplicación. Más concretamente, hemos dividido el documento de la siguiente manera:

En el capítulo dos, explicaremos el documento de objetivos del proyecto (DOP) en el que expondremos el método utilizado para el desarrollo de la aplicación, la división de las tareas en el tiempo, el plan de contingencia (por los posibles problemas que puedan aparecer) y el análisis de factibilidad. En el capítulo tres, explicaremos los antecedentes del proyecto, así como aplicaciones similares que actualmente se encuentran en el mercado. En el capítulo cuatro, explicaremos la elección de la tecnología y las razones que no han llevado a seleccionarla. En el capítulo cinco, realizaremos la captura de requisitos, en la que explicaremos los diagramas de casos de uso (junto a los diseños de pantalla) y el modelo de datos. En el capítulo seis, explicaremos el análisis, en el que expondremos los diferentes contratos de la aplicación. En el capítulo siete, explicaremos el diseño realizado, en el que encontraremos los diferentes diagramas de secuencia, así como una explicación de las funcionalidades más destacables de la aplicación. En el capítulo ocho, expondremos la implementación del proyecto, en la que explicaremos el funcionamiento de un proyecto Android y las estructuras de las dos aplicaciones realizadas. En el capítulo nueve, comprobaremos todas las pruebas realizadas en el proyecto, con el fin de detectar fallos y comprobar el buen funcionamiento de la aplicación. En el capítulo diez, explicaremos todo lo necesario para la implantación de la aplicación. En el capítulo once, comprobaremos las horas dedicadas con las horas reales y si ha habido una gran desviación o ha ido todo correctamente. Por último, en el capítulo doce, encontraremos las conclusiones del proyecto.

2 Dop

2.1 Objetivos

El objetivo del proyecto es crear una aplicación Android que nos proporcione el acceso a la información contenida en el MCR 3.0. En particular, dada una palabra, la aplicación nos proporcionará su significado, la traducción en diferentes idiomas, los sinónimos, antónimos, hiperónimos o hipónimos. También tendremos la opción de elegir el idioma en el que queremos que nos muestre el resultado de nuestra consulta. Dicha aplicación funcionará tanto online, como offline, para que esté siempre disponible.

2.2 Método de trabajo

Para desarrollar el proyecto, se ha elegido el Proceso Unificado de Desarrollo de Software (PUD), siendo uno de los métodos recomendados para el desarrollo del sistema de software.

El PUD se caracteriza por estar dirigido por casos de uso, centrado en la arquitectura y ser iterativo e incremental. Ahora pasaremos a explicar con más claridad, las características principales del PUD.

- Dirigido por los casos de uso

En el Proceso Unificado de Desarrollo, los casos de uso se utilizan para capturar los requisitos funcionales de la aplicación. Además, en cada iteración, se toma un conjunto de casos de uso o escenarios, para desarrollar cada una de las distintas disciplinas: diseño, implementación, prueba, etc.

- Centrado en la arquitectura

El Proceso Unificado de Desarrollo asume que, no existe un componente único que cubra todos los aspectos del sistema. Por dicho motivo, existen múltiples componentes que definen la arquitectura de software de un sistema. La analogía con la construcción es clara, cuando construyes un edificio, existen diversos componentes que facilitan los distintos servicios del mismo: electricidad, fontanería, etc.

- Iterativo e Incremental

El Proceso Unificado de Desarrollo contempla un marco de desarrollo iterativo e incremental, compuesto de cuatro fases denominadas: inicio, elaboración, construcción y transición. Cada una de estas fases está, a su vez, dividida en una serie de iteraciones. Estas iteraciones añaden o mejoran las funcionalidades del sistema en desarrollo.

Cada una de estas iteraciones se divide, a su vez, en una serie de disciplinas que recuerdan a las definidas en el ciclo de vida clásico o en cascada: análisis de requisitos, diseño, implementación y prueba. Aunque todas las iteraciones suelen incluir algún desarrollo en casi todas las disciplinas, el grado de esfuerzo, dentro de cada una de ellas, varía a lo largo del proyecto.

- Guiado por los riesgos

El Proceso Unificado de Desarrollo requiere que el equipo del proyecto se centre en identificar los riesgos críticos en una etapa temprana del ciclo de vida. Los resultados de cada iteración, en especial los de la fase de Elaboración, deben ser seleccionados en un orden que asegure que, los riesgos principales, son considerados en primer lugar.

Cada vez que se finaliza una fase, se concreta un día para realizar una reunión, comentar la fase terminada y concretar nuevos objetivos de la próxima iteración. Estas reuniones se realizaban tras finalizar los objetivos, puesto que no se iban a introducir nuevos si no se había podido finalizar los anteriores. En caso de duda, el alumno disponía de medios suficientes para intentar solventarla mediante: correo, internet, documentación, etc. Si por esos medios no se podía llegar a una solución, se acordaba una reunión con el director para poder solucionarlo cuanto antes y seguir avanzando. También se muestra al director, el progreso del proyecto y el funcionamiento del mismo. Por ese motivo, se muestra la ejecución del mismo.

- Orientado a objetos

El Proceso Unificado de Desarrollo se centra en un enfoque que modela un sistema como un grupo de objetos que interactúan entre sí. Este enfoque representa un dominio absoluto en términos de conceptos, compuestos por verbos y sustantivos, clasificados de acuerdo a su dependencia funcional.

2.3 Alcance

2.3.1 Recursos humanos

Los recursos humanos, para este proyecto, han sido el director y el alumno. El director iba marcando los objetos a cumplir, siendo responsabilidad del alumno completarlos en el tiempo establecido para ello.

2.3.2 Recursos materiales

El alumno disponía de un ordenador portátil con Windows 7, así como material de oficina (impresora, escáner....). También disponía de varias memorias USB en las que se podía hacer copias de seguridad, por si el ordenador portátil fallaba. Si el ordenador portátil fallaba, también se contaba con un ordenador de sobremesa para poder seguir con el proyecto.

El director contaba con su propio ordenador y su material de oficina para poder explicar al alumno los diferentes objetivos.

El lugar elegido para el desarrollo ha sido la casa del alumno en la que se dispone de conexión Wi-Fi y de material de oficina necesario. El lugar elegido para las reuniones ha sido el despacho del director.

2.3.3 Recursos económicos

Al ser un proyecto de fin de carrera, el coste de este proyecto es cero, por lo que no hay recursos económicos. Todo el material utilizado en él ya estaba disponible antes de empezar a realizarlo (ordenador portátil, memorias usb, impresora...), por lo que no se tuvo que hacer ninguna inversión. El único material nuevo fue la parte software, pero se decidió hacerlo con software libre. Por tanto, no se tuvo que pagar ninguna licencia.

2.4 EDT

El EDT es una descomposición jerárquica del trabajo a ser ejecutado. Cada nivel del EDT detalla con más precisión las distintas tareas del proyecto. El EDT es una herramienta fundamental en la gestión de proyectos. En la figura 3 podemos observar el EDT de este proyecto.

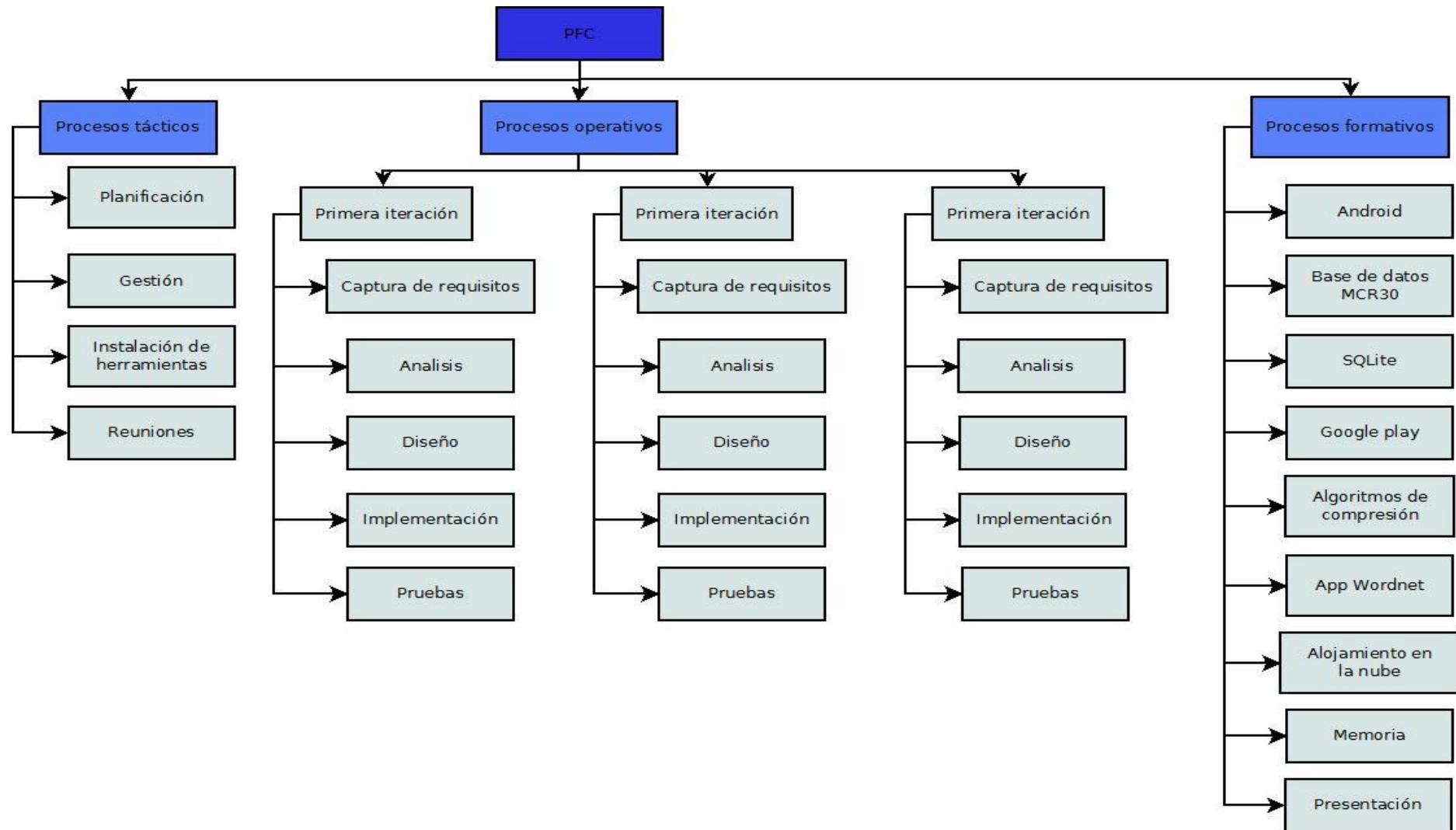


Figura 3: EDT del proyecto.

2.5 Procesos

Dividimos los procesos en las siguientes categorías:

- **Tácticos:** procesos de gestión del proyecto.
- **Formativos:** procesos orientados a la formación del alumno.
- **Operativos:** procesos orientados al producto.

A continuación, se describirá las diferentes partes que conforman dichos procesos, medidas en horas.

2.5.1 Procesos tácticos

En la tabla 1 podemos ver las actividades que conforman los procesos tácticos del proyecto, así como el tiempo planificado para cada una de ellas.

Actividad	Tiempo planificado
Planificación	10
Gestión	5
Instalación	15
Reuniones	10
Total	40

Tabla 1: Tiempo planificado de los procesos tácticos.

2.5.2 Procesos formativos

A continuación, explicaremos las actividades que conforman los procesos formativos del proyecto, contabilizando sus horas en la tabla 2.

- **Android**
 - Aprendizaje de programación en Android para la realización del proyecto, debido a que el alumno desconoce dicha tecnología.
- **Base de datos MCR 3.0**
 - Aprendizaje de la base de datos proporcionada, análisis de las tablas y sus relaciones.
- **SQLite**
 - Aprendizaje del funcionamiento de las bases de datos de tipo SQLite.

- **Google play**
 - Funcionamiento de dicha plataforma para alojar nuestra aplicación.
- **Algoritmos de compresión**
 - Algoritmos para la compresión de la base de datos en nuestra aplicación.
- **App Wordnet**
 - Funcionamiento de las aplicaciones de Wordnet que se encuentra alojada en Google play.
- **Alojamiento en la nube**
 - Funcionamiento de la nube para almacenar distintas funcionalidades de nuestra aplicación.
- **Memoria**
 - Realización de la memoria basada en el proyecto final de carrera.
- **Presentación**
 - Realización de la presentación del proyecto final de carrera.

Actividad	Tiempo estimado
Android	60
BD MCR 3.0	3
SQLite	2
Google Play	1
Algoritmos de compresión	6
App Wordnet	1
Alojamiento en la nube	3
Memoria	60
Presentación	6
Total	142

Tabla 2: Tiempo planificado de los procesos formativos.

2.5.3 Procesos operativos

En este apartado explicaremos las diferentes iteraciones que se han seguido para realizar los procesos operativos, así como las horas planificadas para cada una de ellas (que podremos apreciar en la tabla correspondiente).

- Primera iteración

Descripción: En esta iteración realizaremos el primer prototipo de la aplicación con una base de datos online y offline. En dicho prototipo, implementaremos el caso de uso de obtener significado de la palabra en un único idioma.

Actividad	Tiempo estimado
Captura de requisitos	7
Análisis	5
Diseño	7
Implementación	60
Pruebas	3
Total	82

Tabla 3: Tiempo planificado de la primera iteración.

- Segunda iteración

Descripción: Añadir el resto de casos de uso a la aplicación, así como el soporte para realizar búsquedas en otros idiomas.

Actividad	Tiempo planificado
Captura de requisitos	6
Análisis	4
Diseño	6
Implementación	70
Pruebas	5
Total	91

Tabla 4: Tiempo planificado de la segunda iteración.

- Tercera iteración

Descripción: Creación de una interfaz más visual y funcional de cara al usuario, así como mejoras de la aplicación.

Actividad	Tiempo planificado
Captura de requisitos	8
Análisis	6
Diseño	6
Implementación	50
Pruebas	4
Total	74

Tabla 5: Tiempo planificado de la tercera iteración.

2.5.4 Análisis de factibilidad

Suma total de las horas planificadas para la realización del proyecto

Actividad	Tiempo planificado
Procesos tácticos	40
Procesos formativos	142
Procesos operativos	247
Total	430

Tabla 6: Horas totales planificadas.

Tal y como podemos apreciar en la tabla 6, el tiempo total planificado es de 430 horas, que sobrepasa las horas propuestas para un Proyecto Fin de Carrera de Ingeniería Técnica en Informática de Sistemas, siendo de 400 horas. Aun así, ya se intuye que serán necesarias más horas para la realización del mismo.

Tras analizar las diferentes tareas, el tiempo estimado para su realización y los diferentes riesgos que puede haber, teniendo en cuenta las soluciones propuestas, vemos factible realizar el proyecto en el tiempo establecido.

2.6 Planificación temporal: Diagrama de Gantt

El diagrama de Gantt es una herramienta gráfica cuyo objetivo es exponer el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado. En la figura 4 podemos observar el diagrama de Gantt de nuestro proyecto.

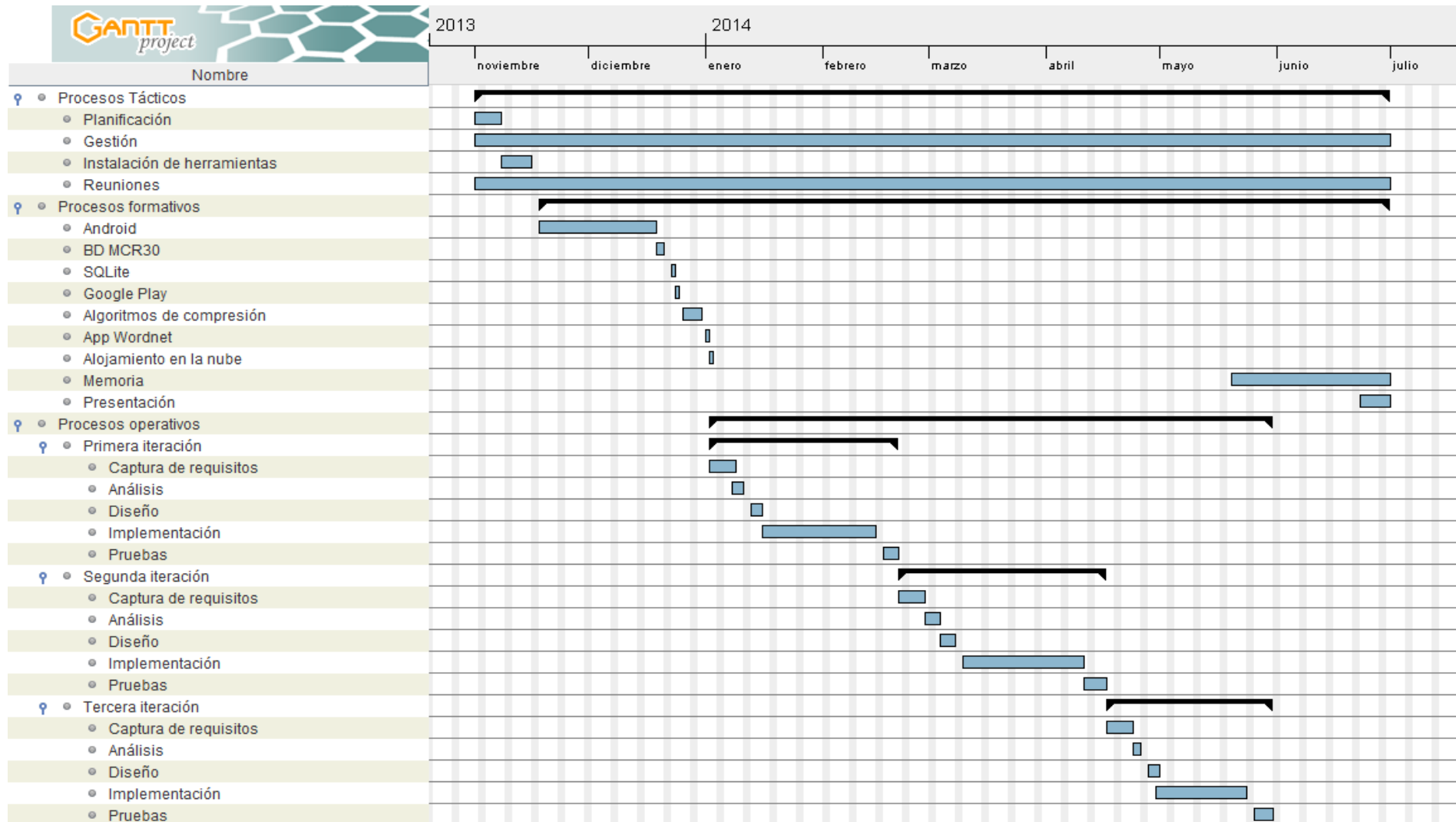


Figura 4: Diagrama de Gantt del proyecto.

2.7 Plan de contingencia

A continuación, se exponen los diferentes riesgos que pueden darse a lo largo del proyecto. Para poder actuar de una manera rápida y eficaz, es importante tenerlos identificados, debido a que puede significar el éxito o el fracaso del proyecto. Ahora, nos dispondremos a explicar una serie de riesgos, junto a sus soluciones:

- Pérdida de documentación

Descripción: Pérdida de la documentación, ya sea física o por no poder tener acceso a ella.

Probabilidad: Baja.

Gravedad: Alta.

Solución: La documentación física se podía conseguir imprimiendo otra vez lo mismo. Si son explicaciones sobre papel, hechas por el director, se le comunicaría para que pueda repetirlas. El resto de la documentación, proporcionada por el director, se obtiene del correo del alumno, siendo éste el medio por el que se mandaba. Toda la documentación obtenida por el alumno se guardaba en una copia de seguridad.

2.7.1 Riesgos Tácticos

- Imposibilidad para reunirse

Descripción: La imposibilidad por parte del alumno o del directo para reunirse.

Probabilidad: Baja.

Gravedad: Media.

Solución: Utilizar el correo o la videoconferencia (skype), con el cual, el alumno y el director, puedan seguir comunicándose para avanzar con el proyecto.

- Retraso en la tarea

Descripción: Retraso en alcanzar los objetivos marcados por el director.

Probabilidad: Media.

Gravedad: Media.

Solución: Resolver los problemas que han originado el retraso y replanificar.

2.7.2 Riesgos Operativos

- Pérdida del proyecto

Descripción: Pérdida de todos los datos relacionados con el proyecto.

Probabilidad: Baja.

Gravedad: Alta.

Solución: Usar la copia de seguridad más reciente guardado en una memoria USB.

- Ordenador estropeado

Descripción: El ordenador actual de trabajo deja de funcionar.

Probabilidad: Media.

Gravedad: Media.

Solución: Se carga la copia de seguridad más reciente en un segundo ordenador, que estaba disponible desde el principio del proyecto.

- Problemas a la hora de instalar el software

Descripción: Problemas que puedan surgir a la hora de instalar el software necesario para la elaboración del proyecto.

Probabilidad: Media.

Gravedad: Media.

Solución: Reunirse con el director para intentar solucionarlos lo antes posible. En caso de que no haya sido posible solucionarlos, buscar información por internet, manuales, etc.

- Copia de seguridad estropeada

Descripción: La unidad USB en la que se guardaba la copia de seguridad deja de funcionar.

Probabilidad: Baja.

Gravedad: Alta.

Solución: Tener siempre dos memorias USB en las que guardar una copia de seguridad, porque al no ser un proyecto de mucha envergadura, es posible realizarlas en un pequeño lapso de tiempo. También sustituir la memoria USB estropeada por otra, y realizar la copia de seguridad.

2.7.3 Riesgos Formativos

- Dificultad a la hora de manejar el software

Descripción: Problemas surgidos a la hora de manejar el software del proyecto por desconocimiento del alumno.

Probabilidad: Media.

Gravedad: Media.

Solución: Buscar información en internet, manuales, etc., para intentar solucionarlo lo antes posible. Si el director tiene conocimiento sobre el mismo, preguntarle directamente.

3 Antecedentes

En los siguientes apartados, expondremos el marco de trabajo de nuestro proyecto.

Explicaremos las bases de conocimiento léxicas que forman parte de nuestro proyecto. Más concretamente, presentaremos WordNet⁵, base de conocimiento léxico en inglés más utilizado en el área del procesamiento del lenguaje natural, EuroWorNet⁶, arquitectura para el desarrollo de una base de conocimiento multilingüe que incluye wordnets de idiomas europeos, y *Multilingual Central Repository (MCR)*⁷ (así como el su versión más actual, MCR 3.0 y su interfaz, *Web Euro WordNet Interface*⁸) para el mantenimiento de la compatibilidad entre wordnets de diferentes idiomas. Todos los wordnet comentados, cuya licencia nos permita su distribución, podemos encontrarlos en la página *Open Multilingual Wordnet*⁹, que también explicaremos más adelante.

Además, analizaremos diferentes propuestas que podemos encontrar en el mercado que usan la base de conocimiento léxica WordNet, así como sus diferentes funcionalidades.

3.1 WordNet

Desarrollada en la universidad de Princeton, WordNet (Miller et al., 1990) es una base de datos léxica de dominio general para el inglés, que actualmente constituye uno de los recursos léxicos más utilizados en el área del Procesamiento del Lenguaje Natural (PLN). WordNet (WN) ha sido creado para intentar organizar la información léxica por significados. A diferencia de los diccionarios convencionales, donde esta información está organizada por la forma de las unidades léxicas.

WN está estructurada como una red semántica cuyos nodos, denominados synsets (synonym sets, o conjuntos de sinónimos) constituyen su unidad básica de significado. Cada uno de ellos se compone de un conjunto de las lexicalizaciones que representan un sentido y se identifica mediante un “offset” (byte) y su correspondiente PoS (Part of Speech); que puede ser (n) para nombres, (v) para verbos, (a) para adjetivos y ® para adverbios. Por ejemplo:

- 02152053#n fish#1
- 01926311#v run#1
- 02545023#a funny#4

⁵ <http://wordnet.princeton.edu>

⁶ <http://www.illc.uva.nl/EuroWordNet>

⁷ <http://adimen.si.ehu.es/web/MCR>

⁸ <http://adimen.si.ehu.es/cgi-bin/wei/public/wei.consult.perl>

⁹ <http://compling.hss.ntu.edu.sg/omw/>

- 00005567#r automatically#1

El número que aparece después de cada palabra indica el número de sentido que representa el synset. Una palabra puede ser polisémica, esto es, tener varios significados. Por ejemplo:

- 02152053#n fish#1; representa la palabra “fish” con el sentido de pez como animal vertebrado acuático.
- 07775375#n fish#2; representa la palabra “fish” con el sentido de pescado como alimento.

También puede ser que varias palabras tengan el mismo significado, esto es, que sean sinónimas. En Wordnet están representados en el mismo synset:

- 02383458#n car#1, auto#1, automobile#1, machine#4, motorcar#1; representa el vehículo de cuatro ruedas.

Todos los synsets incluyen una glosa a modo de definición similar a la del diccionario tradicional, que describe el significado del concepto de forma explícita. Por ejemplo:

- 02383458#n car#1; 4-wheeled motor vehicle; usually propelled by an internal combustion engine: he needs a car to get to work;

Como ya se expuso anteriormente, WordNet está estructurada como una red semántica. Además de la información que pueden proporcionar los sinónimos (componentes del synset) y la glosa, tenemos que tener en cuenta los arcos de la red semántica. Éstos establecen diferentes relaciones entre los synsets, por ejemplo:

Hiperonimia: Es el término genérico usado para designar a una clase de instancias específicas. Y es un hiperónimo de X, si X es una clase de Y.

- tree#n#1 HYPERONYM oak#n#2

Hiponimia: Es el término específico usado para designar el miembro de una clase, X es un hipónimo de Y, si X es una clase de Y. En el caso de los verbos se denomina Troponimia.

- oak#n#2 HYPONYM tree#n#1

Antonimia: Es la relación que enlaza dos sentidos con significados opuestos.

- active#a#1 ANTONYM_OF inactive#a#2
- inactive#a#2 ANTONYM_OF active#a#2

Meronomia: Es la relación que se define como componente de, substancia de, o miembro de algo, X es merónimo de Y si X es parte de Y.

- car##1 HAS_PART window##2
- milk##1 HAS_SUBSTANCE protein##1
- family##1 HAS_MEMBER child##2

Holonomia: Es la relación contraria a la meronomia, Y es holónimo de X si X es una parte de Y.

- window##2 PART_OF car##1
- protein##1 SUBSTANCE_OF milk##1
- child##2 MEMBER_OF family##2

El conocimiento de Wordnet va aumentando en cada una de sus versiones según se van incluyendo nuevos synsets. En las tablas 7 y 8 podemos comprobar el estado en el que encuentra WordNet en su versión 3.0.

Pos	Únicos Strings	Synsets	Pares de sentido-palabra
Nombre	117798	82115	146312
Verbo	11529	13767	25047
Adjetivo	21479	18156	30002
Adverbio	4481	3621	5580
Total	155287	117659	206941

Tabla 7: Número de palabras, synsets y sentidos.

Pos	Únicos Strings	Synsets	Pares de sentido-palabra
Nombre	101863	15935	44449
Verbo	6277	5252	18770
Adjetivo	16503	4976	14399
Adverbio	3748	733	1832
Total	128391	26896	79450

Tabla 8: Información Polisémica que contiene.

WordNet también dispone de una interfaz on-line ¹⁰ para poder consultar su base de datos. Nos permite buscar una palabra en inglés, junto con las opciones que queremos que nos muestre de dicha palabra. En la figura 5 podemos apreciar el funcionamiento de dicha interfaz.

¹⁰ <http://wordnetweb.princeton.edu/perl/webwn>

WordNet Search - 3.1

- [WordNet home page](#) - [Glossary](#) - [Help](#)

Word to search for:

Display Options:

Key: "S:" = Show Synset (semantic) relations, "W:" = Show Word (lexical) relations

Display options for sense: (gloss) "an example sentence"

Noun

- **S: (n) market, marketplace, market place** (the world of commercial activity where goods and services are bought and sold) *"without competition there would be no market"; "they were driven from the marketplace"*
- **S: (n) market** (the customers for a particular product or service) *"before they publish any book they try to determine the size of the market for it"*
- **S: (n) grocery store, grocery, food market, market** (a marketplace where groceries are sold) *"the grocery store included a meat market"*
- **S: (n) market, securities industry** (the securities markets in the aggregate) *"the market always frustrates the small investor"*
- **S: (n) marketplace, market place, mart, market** (an area in a town where a public mercantile establishment is set up)

Verb

- **S: (v) market** (engage in the commercial promotion, sale, or distribution of) *"The company is marketing its new line of beauty products"*
- **S: (v) market** (buy household supplies) *"We go marketing every Saturday"*
- **S: (v) market** (deal in a market)
- **S: (v) commercialize, commercialise, market** (make commercial) *"Some Amish people have commercialized their way of life"*

Figura 5: Ejemplo WordNet.

Cada uno de los resultados obtenidos por la interfaz, es agrupado por su tipo de palabra: nombre, verbo, adjetivo o adverbio. Esto también está reflejado delante de cada una de las respuestas obtenidas, en este caso, *n* o *v*. Cada respuesta representa un sentido diferente de la palabra por lo que cada una tendrá sus palabras, significados y ejemplos propios, tal y como podemos apreciar en la figura 5. Además, la interfaz nos deja movernos por las diferentes relaciones conectadas con alguno de los sentidos de la palabra que hemos buscado, solamente tendríamos que seleccionar las palabra que se encuentran en azul. Se podría decir que es un gráfico dinámico que nos permite acceder a los distintos nodos.

3.2 EuroWordNet

El proyecto EuroWordNet (Vossen, 1998) diseñó una arquitectura completa para el desarrollo de una base de conocimiento multilingüe que incluyera varios wordnets de idiomas europeos (entre ellos, holandés, italiano, castellano, alemán, francés, checo y estonio). En EuroWordNet, cada WordNet representa un único sistema interno de lexicalizaciones siguiendo la estructura del wordnet inglés. Los wordnets de los distintos idiomas están ligados mediante el Inter-Lingual Index (abreviado como ILI). Estas conexiones permiten acceder a palabras similares en cualquiera de los idiomas integrados en la arquitectura EuroWordNet. Además, el ILI da acceso a una ontología lingüística compuesta por 63 relaciones semánticas distintas. Esta ontología proporciona una categorización común para todos los idiomas, mientras que las distinciones específicas de cada idioma están en cada uno de los wordnets locales. Aunque el proyecto EuroWordNet se concluyó en el verano de 1999, muchos de sus principios siguen aún vigentes. Por ejemplo, el diseño de la arquitectura multilingüe, los Base Concepts, las relaciones, la ontología, etc., se han seguido buscando por grupos de investigación, que están desarrollando wordnets en otros idiomas (como por ejemplo, el castellano, el euskera, el catalán y el gallego) usando buena parte de la especificación de EuroWordNet. Si los wordnets son compatibles con la especificación, pueden ser añadidos a una base de datos común, y mediante el ILI, ser conectados con otros wordnets, permitiendo el uso aplicaciones multilingües de lenguaje natural.

3.3 Open Multilingual Wordnet

Open Multilingual Wordnet¹¹ nos proporciona acceso a los wordnets de una gran variedad de lenguajes, todos ellos enlazados al WordNet de Princeton. Cada uno de los wordnets ha sido desarrollado en diferentes proyectos, por lo que tienen un tamaño y precisión diferentes. Esta página se ha encargado de realizar las siguientes operaciones sobre los diferentes wordnets:

- Extraer y normalizar la información.
- Enlazarlo al WordNet 3.0 de Princeton.
- Colocarlo en un único lugar.

Sólo se incluyen aquellos cuya licencia nos permita su distribución. Si queremos comprobar todos los wordnets existentes, independientemente de su licencia, podemos consultar la lista en *Global Wordnet Association's, Wordnets in the World*¹².

¹¹ <http://compling.hss.ntu.edu.sg/omw/>

¹² <http://globalwordnet.org/wordnets-in-the-world/>

Cada uno de los wordnet de la página contienen: el nombre del wordnet, el código del idioma, el número de synset, palabras y sentidos, cantidad, la licencia que utiliza, la información y la citación. Entre todos los wordnet, podemos encontrar el MCR con sus cuatro idiomas: español, euskera, catalán y gallego. En la figura 6 podemos ver un ejemplo de la estructura de los wordnet:

Wordnet	Lang	Synsets	Words	Senses	Core	Licence	Data	Citation
Albanet	als	4,676	5,990	9,602	31%	CC BY 3.0	als.zip (+xml)	cite:als; (.bib)
Arabic WordNet (AWN)	arb	10,165	14,595	21,751	48%	CC BY SA 3.0	arb.zip (+xml)	cite:arb; (.bib)
Chinese Open Wordnet	cmn	42,312	61,533	79,809	100%	wordnet	cmn.zip (+xml)	cite:cmn; (.bib)
Chinese Wordnet (Taiwan)	qcn	4,913	3,206	8,069	28%	wordnet	qcn.zip (+xml)	cite:qcn; (.bib)
DanNet	dan	4,476	4,468	5,859	81%	wordnet	dan.zip (+xml)	cite:dan; (.bib)
Greek Wordnet	ell	18,049	18,227	24,106	57%	Apache 2.0	ell.zip (+xml)	cite:ell; (.bib)
Princeton WordNet	eng	117,659	148,730	206,978	100%	wordnet	eng.zip (+xml)	cite:eng; (.bib)
Persian Wordnet	fas	17,759	17,560	30,461	41%	Free to use	fas.zip (+xml)	cite:fas; (.bib)
FinnWordNet	fin	116,763	129,839	189,227	100%	CC BY 3.0	fin.zip (+xml)	cite:fin; (.bib)
WOLF (Wordnet Libre du Français)	fra	59,091	55,373	102,671	92%	CeCILL-C	fra.zip (+xml)	cite:fra; (.bib)
Hebrew Wordnet	heb	5,448	5,325	6,872	27%	wordnet	heb.zip (+xml)	cite:heb; (.bib)
MultiWordNet	ita	34,728	40,343	61,558	83%	CC BY 3.0	ita.zip (+xml)	cite:ita; (.bib)
Japanese Wordnet	jpn	57,184	91,964	158,069	95%	wordnet	jpn.zip (+xml)	cite:jpn; (.bib)
Multilingual Central Repository	cat	45,826	46,531	70,622	81%	CC BY 3.0	cat.zip (+xml)	cite:cat; (.bib)
Multilingual Central Repository	eus	29,413	26,240	48,934	71%	CC BY-NC-SA 3.0	eus.zip (+xml)	cite:eus; (.bib)
Multilingual Central Repository	glg	19,312	23,124	27,138	36%	CC BY 3.0	glg.zip (+xml)	cite:glg; (.bib)
Multilingual Central Repository	spa	38,512	36,681	57,764	76%	CC BY 3.0	spa.zip (+xml)	cite:spa; (.bib)

Figura 6: Wordnets disponibles.

3.4 Multilingual Central Repository (MCR)

Uno de los principales resultados del proyecto MEANING¹³ fue el desarrollo de la primera versión del Multilingual Central Repository, para mantener la compatibilidad entre wordnets de distintos idiomas y versiones, tanto nuevos como anteriores, así como el nuevo conocimiento que se fuera adquiriendo.

Todo el diseño del MCR sigue la arquitectura propuesta por EuroWordNet. Esta arquitectura hace posible desarrollar wordnets locales de forma relativamente independiente, garantizando al mismo tiempo un alto nivel de compatibilidad. Esta estructura multilingüe permite transportar el conocimiento de un wordnet al resto de wordnets a través del ILI (Inter-Lingual Index), manteniendo la compatibilidad entre todos ellos. De esta forma, la estructura del ILI (incluyendo la Top, Wordnet Domains y la ontología SUMO) actúa como la columna vertebral que permite transferir el conocimiento adquirido de cada uno de los wordnets locales al resto. Del mismo modo, los diferentes recursos (ej. las diferentes ontologías) están relacionadas mediante el ILI, y en consecuencia también pueden ser validados entre ellos.

Al término del proyecto MEANING, el MCR ha continuado su desarrollo y mejora en los proyectos nacionales KNOW¹⁴ y KNOW2¹⁵, así como varias acciones complementarias, con especial énfasis en los idiomas inglés, castellano, catalán, euskera y gallego.

La versión actual del MCR integra, siguiendo la arquitectura EuroWordNet, wordnets de cinco idiomas diferentes: inglés, castellano, catalán, euskera y gallego. El *Inter-Lingual-Index* (ILI) permite la conectividad entre las palabras en un idioma con las traducciones equivalentes en cualquiera de las otras lenguas, gracias a los enlaces generados automáticamente. El ILI actual corresponde a la versión 3.0 de WordNet.

Por ello, el MCR constituye un recurso multilingüe, de amplia cobertura, que puede ser de gran utilidad para un gran número de procesos semánticos que requieren de conocimientos lingüístico-semánticos ricos y complejos (por ejemplo, ontologías para la web semántica). Así, el MCR está siendo utilizado en múltiples proyectos y desarrollos. Por ejemplo, los proyectos europeos KYOTO¹⁶, PATHS¹⁷, OpeNER¹⁸ y NewsReader, y el proyecto nacional SKaTer.

¹³ <http://nlp.lsi.upc.edu/projectes/meaning>

¹⁴ <http://ixa2.si.ehu.es/know>

¹⁵ <http://ixa2.si.ehu.es/know2>

¹⁶ <http://www.kyoto-project.eu>

¹⁷ <http://www.paths-project.eu>

¹⁸ <http://www.opener-project.org/>

3.5 Multilingual Central Repository 3.0

La versión actual del MCR usa un ILI basado en WordNet 3.0 integra, siguiendo el modelo propuesto por EuroWordNet y MEANING, wordnets de cinco idiomas distintos, incluyendo el inglés, castellano, catalán, euskera y gallego. Como en la versión anterior, los wordnets están conectados a través del *Inter-Lingual-Index* (ILI), permitiendo conectar palabras de un idioma, con las palabras equivalentes en los otros idiomas, también integrados en el MCR. Como la versión actual del ILI del MCR es la correspondiente a la versión 3.0 del WordNet en inglés, la mayoría del conocimiento ontológico ha sido transportado desde las versiones anteriores al nuevo MCR 3.0. Así, la mayoría de los recursos transportados han tenido que ser alineados a la nueva versión. La descripción completa del proceso empleado para llevar a cabo el transporte y actualización de todos los recursos se puede consultar.

Además, para poder interactuar con el MCR y actualizar su contenido, hemos actualizado el Web EuroWordNet Interface (WEI), una nueva interfaz web para navegar y editar el MCR 3.0.

3.6 Web EuroWordNet Interface

Web EuroWordNet Interface (WEI) permite consultar y editar la información contenida en el MCR. WEI usa tecnología CGI, lo que significa que todos los datos se procesan sólo en el servidor, y los usuarios trabajan con clientes ligeros con capacidad de navegación web HTML. Todos los datos se almacenan en una base de datos MySQL. La interfaz se ha ido actualizando desde su desarrollo inicial en el proyecto EuroWordNet.

WEI permite navegar, consultar y editar la información asociada a un ítem (que puede ser un synset, una palabra, un variant o un (ILI) de uno de los wordnets integrados en el MCR. La aplicación WEI consulta la información correspondiente a ese ítem y la información ontológica asociada a los ILIs correspondientes. La aplicación también permite consultar por los synsets relacionados del propio wordnet origen de la consulta (usando las relaciones codificadas en el MCR de hiperonimia, hiponimia, meronimia, etc.), o de algún otro wordnet integrado en el MCR (a través del ILI).

La aplicación consta de dos marcos. En el marco superior, introducimos los parámetros para la búsqueda, y en inferior nos muestra los resultados de la consulta. Los diferentes parámetros de búsqueda son los siguientes:

- **Ítem:** el ítem que pretendemos buscar, que puede ser una palabra, un synset, un variant o un ILI.
- **Tipo de ítem:** el tipo del ítem que pretendemos buscar (palabra, variant, synset o ILI).
- **PoS:** la categoría gramatical del ítem (nombres, verbos, adjetivos o adverbios).
- **Relación:** que se carga dinámicamente desde la base de datos (sinónimos, hipónimos, hiperónimos, etc.)
- **WordNet origen:** el wordnet desde donde realizamos la consulta.
- **WordNet navegación:** el wordnet al cual seguimos las relaciones.
- **Glossa:** si está seleccionado, se muestran las glosas de los synsets.
- **Score:** si está seleccionado, se muestran los valores de confianza.
- **Rels:** si está seleccionado, muestra información acerca de las relaciones que el synset tiene en todos los wordnets seleccionados.
- **Full:** si está seleccionado, realiza una búsqueda transitiva por todas las relaciones.
- **WordNets mostrados:** wordnets seleccionados.

En la figura 7, podemos ver una captura con el funcionamiento de la aplicación, así como todas las opciones anteriormente comentadas:

<input type="text" value="party"/>	<input type="button" value="Look up"/>	<input checked="" type="checkbox"/> Gloss	<input checked="" type="checkbox"/> English_3.0	<input type="checkbox"/> Catalan_3.0
Word <input type="text" value="Nouns"/>	English_3.0 <input type="text"/>	<input type="checkbox"/> Score	<input type="checkbox"/> Basque_3.0	
near_synonym <input type="text"/>	English_3.0 <input type="text"/>	<input checked="" type="checkbox"/> Rels	<input checked="" type="checkbox"/> Spanish_3.0	
		<input checked="" type="checkbox"/> Full	<input type="checkbox"/> Galician_3.0	

Multilingual Central Repository (ILI 3.0) - [WikiMCR](#)

ili-30-08256968-n

[anthropology](#)

[history](#)

[politics](#)

[sociology](#)


[group](#)

[PoliticalOrganization](#)


[Function](#)

[Group](#)

[Human](#)

eng-30-08256968-n  37

[party_1](#) [political_party_1](#)

spa-30-08256968-n  37

[partido_1](#) [partido_politico_1](#)

an organization to gain political power: in 1992 Perot tried to organize a third party at the national level;

una organización para obtener poder político: en 1992 Perot trató de organizar un tercer partido a nivel nacional;

[34](#) [has_hyponym](#) [1](#) [has_holo_member](#) [1](#) [gloss](#) [1](#) [has_hyperonym](#) [113](#) [rgloss](#)

[34](#) [has_hyponym](#) [1](#) [has_holo_member](#) [1](#) [gloss](#) [1](#) [has_hyperonym](#) [113](#) [rgloss](#)

Figura 7: Búsqueda de la palabra “party”.

WEI también permite editar el contenido del MCR. Su funcionamiento es exactamente igual a la consulta, pero en modo edición, tanto los synsets como los ILIs pueden seleccionarse y editarse. Al editar un synset, nos aparece una pantalla de donde podemos añadir, eliminar o modificar, los variants del synset, modificar su glosa y ejemplos, así como las relaciones que tiene con otros synsets. Al editar un ILI, nos

aparece una pantalla donde podremos añadir, eliminar o modificar la información ontológica asociada al ILI.

3.7 Aplicaciones alternativas

Actualmente existen diferentes aplicaciones que utilizan Wordnet para obtener los diferentes significados de las palabras, ejemplos y palabras, todas ellas agrupadas por el sentido. A continuación, haremos un repaso de las aplicaciones más destacadas, así como su funcionamiento.

3.7.1 ColorDict Diccionario

ColorDict¹⁹ es aplicación gratuita que nos permite buscar el significado de una palabra o frase, mediante WordNet, StarDict²⁰ o Wikipedia y, en este último caso, darnos su información (siempre que sea posible). Para poder utilizar Wordnet o StarDict, será necesario descargarlo con antelación. En el caso de StarDict, tenemos los siguientes diccionarios:

- Diccionario de sinónimos en Ingles (stardict)
- Diccionario Inglés-Japonés sin conexión (stardict)
- Diccionario Inglés-Chino(stardict)
- Diccionario Inglés-Alemán (stardict)
- Inglés-Español

Para instalar el diccionario de WordNet, será necesario descargar otra aplicación llamada *English Dictionary Wordnet*²¹, que únicamente servirá para instalarnos el diccionario en la tarjeta SD.

¹⁹ <https://play.google.com/store/apps/details?id=com.socialnmobile.colordict>

²⁰ <http://es.wikipedia.org/wiki/StarDict>

²¹

<https://play.google.com/store/apps/details?id=com.socialnmobile.dictdata.dictionary.english.wordnet3>

En la figura 8 podemos observar la interfaz de la aplicación, así como la forma en que representa los diferentes datos:



Figura 8: Interfaz ColorDict.

3.7.2 Free Dictionary Org

*Free Dictionary Org*²² es una aplicación gratuita que nos permite buscar el significado de una palabra en inglés mediante WordNet o Wikipedia, así como buscar en Google imágenes asociadas a la palabra o realizar pequeños tests. Como con la aplicación anterior, será necesario descargar el diccionario de WordNet mediante otra aplicación llamada *Wordnet-Free Dictionary Org*²³ para poder utilizarlo. En las siguientes figuras podemos observar la interfaz de la aplicación cuando buscamos una palabra en WordNet o en Wikipedia.

²² <https://play.google.com/store/apps/details?id=org.freedictionary>

²³ <https://play.google.com/store/apps/details?id=org.freedictionary.offline.wordnet.data&hl=es>

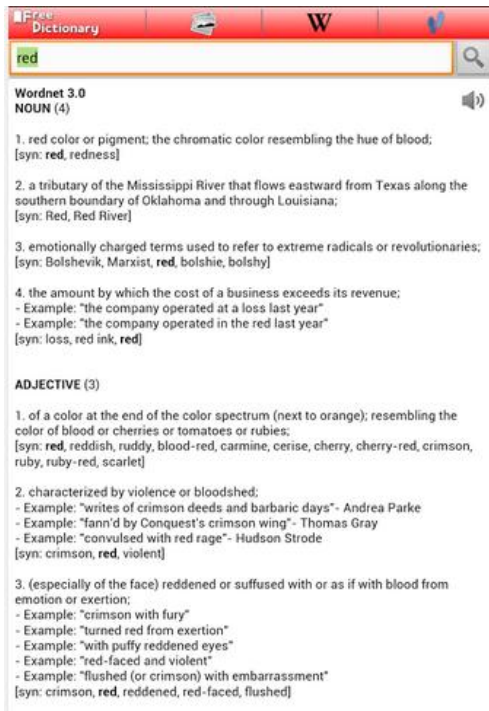


Figura 10: Búsqueda en WordNet.

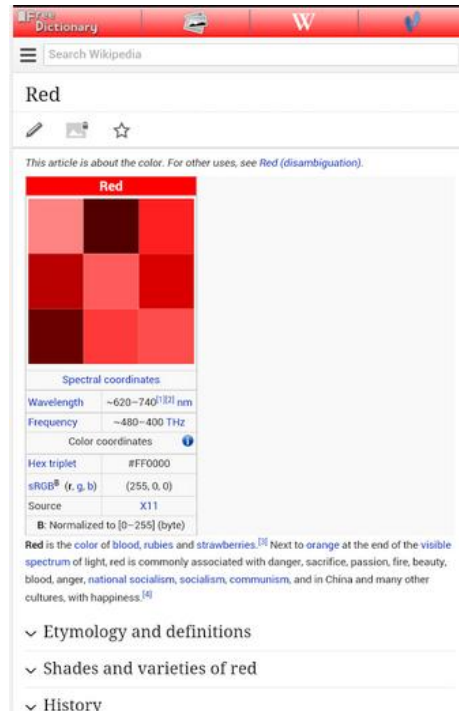


Figura 9: Búsqueda en Wikipedia.

4 Elección de la tecnología

Para la realización de este proyecto, ha sido necesario decidir en qué sistema móvil se iba a realizar la aplicación. El elegido ha sido el sistema operativo Android, el cual pasaremos a explicar a continuación, junto a las razones que nos han llevado a dicha decisión.

4.1 Android

Android es un sistema operativo basado en el kernel de Linux, diseñado, principalmente, para dispositivos móviles con pantalla táctil, como teléfonos inteligentes o tabletas, inicialmente desarrollado por Android, Inc. Google respaldó económicamente y más tarde compró esta empresa en 2005. Android fue presentado en 2007, junto a la fundación del *Open Handset Alliance*²⁴: un consorcio de compañías de hardware, software y telecomunicaciones, para avanzar en los estándares abiertos de los dispositivos móviles.

4.1.1 Análisis de mercado

El sistema operativo Android de Google se convirtió en el más utilizado en los teléfonos del mundo en el tercer trimestre del 2013, ampliando su ventaja sobre el iOS de Apple y el Windows Phone de Microsoft.

Como se puede ver en la tabla 9, Android lidera ampliamente como sistema operativo más utilizado, rozando el 80% de la cuota global de mercado en todo el mundo, mientras que iOS se encuentra en la segunda posición superando el 13%, y Windows Phone queda relegado a la tercera posición con un 3,7%.

Sistema Operativo	3Q13 Unidades	3Q13 Cuota mercado	3Q12 Unidades	3Q12 Cuota mercado	Cambio anual
Android	211.6	81.0%	139.9	74.9%	51.3%
iOS	33.8	12.9%	26.9	14.4%	25.6%
Windows Phone	9.5	3.6%	3.7	2.0%	156.0%
BlackBerry OS	4.5	1.7%	7.7	4.1%	-41.6%
Otros	1.7	0.6%	8.4	4.5%	-80.1%
Total	261.1 mill	100.0%	156.2	100.0%	39.9%

Tabla 9: Mercado Android.

²⁴ <http://www.openhandsetalliance.com/>

Tercero, comprobamos los informes realizados por Kantar²⁵. En cuanto a los sistemas operativos móviles vendidos, encontramos cifras tan interesantes como que en España, la cuota Android es el 91% en España, más del 70% en Italia, 77% en Alemania, 51% en EEUU y 62% en México.

Mercado de smartphones español: veamos las variaciones entre los trimestres de 2012 y 2013:

- **Android:** 91% (3Q'13) vs 82%(3Q'12)
- **iOS:** 5.6% (3Q'13) vs 2.8%(3Q'12)
- **Symbian:** 0.9%(3Q'13) vs 2.6%(3Q'12)
- **Blackberry:** 0.4% (3Q'13) vs 9.2%(3Q'12)
- **Windows Phone:** 1.4% (3Q'13) vs 2.2%(3Q'12)

Android e iOS representan en España el 96,4% de los sistemas operativos del mercado, mientras que vemos como Symbian (Nokia), RIM (Blackberry) y Windows Phone alcanzan justo el 3% del total entre los 3.

4.1.2 Versión de Android

Como cualquier otro sistema operativo, Android también dispone de diferentes versiones que se han ido creando con el paso del tiempo para mejorar dicho sistema. En la figura 11 podemos observar las diferentes versiones de Android, junto al API asociado a cada una de ellas, así como su popularidad.

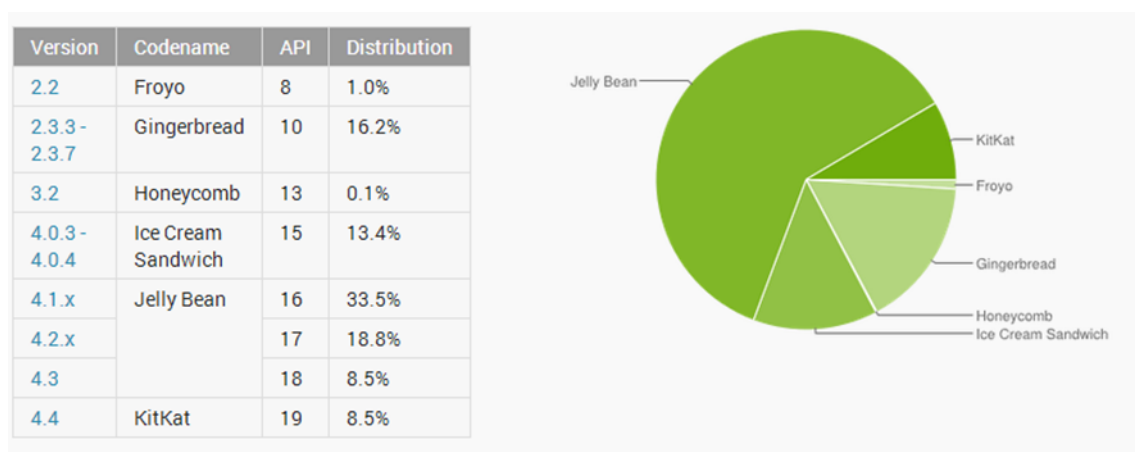


Figura 11: Versiones Android.

²⁵ <http://www.kantar.com/>

Tras analizar los datos, se optó por realizar la aplicación como sistema operativo más bajo el 2.3 y el más alto el 4.3. Se decidió realizarlo de esta manera debido a que Android nos exige que indiquemos cual será la más baja y la más alta a la hora de crear una aplicación.

4.2 Resto de tecnología

Tras decidir el sistema operativo para el que se iba a desarrollar, había que decidir en qué lenguaje de programación se iba a realizar, puesto que Android soporta varias alternativas. Se optó por el lenguaje de programación Java frente a C, debido a que era el más aconsejable para programar en Android, disponía de muchos manuales con los que trabajar y era el lenguaje con el que el alumno estaba más familiarizado. Para la base de datos online, se decidió usar MySQL, puesto que el profesor nos proporcionaba la base de datos en dicho formato, y era la manera más rápida y fácil de trabajar. Para la base de datos offline, se tuvo que usar SQLite, siendo la única con la que Android trabaja internamente.

A continuación explicaremos las herramientas elegidas para desarrollar cada parte de dicha aplicación.

4.2.1 Sistema operativo

El sistema operativo elegido ha sido Windows 7, puesto que es el sistema que tenía el alumno y con el que mejor se manejaba. En este caso no habría mucha diferencia entre trabajar en Windows o Linux, siendo el entorno de programación y el emulador los mismos en ambos sistemas operativos.

4.2.2 Entorno de desarrollo

El entorno de desarrollo elegido ha sido Eclipse²⁶. Se ha decidido usar este entorno, gracias a que Google nos proporcionaba una versión preparada para programar en Android, sin necesidad de andar instalando nada y ahorrándonos tiempo. También contamos con un emulador integrado en el que podemos llevar a cabo las pruebas pertinentes sobre nuestra aplicación. Si queremos instalar API's menores o mayores,

²⁶ <http://www.eclipse.org/>

también nos ofrece una utilidad con la que realizar dicha instalación sin mayor complicaciones.

4.2.3 PhpMyAdmin

Para la base de datos online, se decidió usar phpMyAdmin²⁷. Es una herramienta escrita en PHP, con la intención de manejar la administración de MySQL a través de páginas web, utilizando Internet. Actualmente, puede crear y eliminar Bases de Datos; crear, eliminar y alterar tablas; borrar, editar y añadir campos; ejecutar cualquier sentencia SQL, administrar claves en campos, administrar privilegios, exportar datos en varios formatos y está disponible en 62 idiomas. Se encuentra disponible bajo la licencia GPL.

4.2.4 SQLitestudio

Para la base de datos offline, se decidió usar SQLitestudio²⁸. Es una herramienta multiplataforma que nos permite administrar bases de datos en SQLite, además de que puede hacer las mismas operaciones que realizamos en phpMyAdmin.

4.2.5 PHP

Para la base de datos online, ha sido necesario utilizar el lenguaje PHP, puesto que Android no permitía una conexión directa con la base de datos. PHP es un lenguaje de programación de uso general de código del lado del servidor, originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML, en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante.

²⁷ http://www.phpmyadmin.net/home_page/index.php

²⁸ <http://sqlitestudio.pl/>

4.2.6 JSON

Para llevar los datos del archivo PHP a la aplicación, se decidió utilizar JSON, puesto que nos ofrece una manera fácil y cómoda de almacenar los datos. JSON, o *JavaScript Object Notation*, es un formato de estándar abierto que utiliza texto legible para transmitir estructuras de datos, que constan de un atributo y un valor. Se utiliza sobre todo para transmitir datos entre un servidor y aplicaciones web, como alternativa a XML.

4.2.7 Documentación

Es necesario decidir la tecnología utilizada para realizar la documentación del proyecto. Se ha elegido el paquete ofimático de Microsoft Office frente a la opción de Libre Office²⁹, gracias a que ofrece una interfaz más sencilla y manejable, perdiendo así menos tiempo realizando las tareas. Otra de las razones por las que se eligió este paquete, fue porque el alumno estaba más familiarizado con él y ya lo tenía instalado.

4.2.7.1 Microsoft Word

Microsoft Word es el procesador de texto del paquete ofimático de Microsoft Office. Su formato propietario DOC es considerado un estándar de facto, aunque en su versión Word 2007, utiliza un nuevo formato basado en XML llamado DOCX, pero también tiene la capacidad de guardar y abrir documentos en el formato DOC. También nos permite guardar el trabajo en PDF evitando tener que usar herramientas externas para realizar la conversión.

4.2.7.2 Microsoft PowerPoint

Microsoft PowerPoint es un popular programa para desarrollar y desplegar presentaciones visuales en entornos Windows y Mac que forma parte de del paquete ofimático de Microsoft office. Es usado para crear diapositivas multimediales, es decir, compuesta por texto, imágenes, sonido, animaciones y vídeos.

²⁹ <https://es.libreoffice.org/>

5 Captura de requisitos

Tal y como hemos comentado anteriormente, nuestro objetivo es crear una versión móvil del diccionario online MCR 3.0, que llamaremos MCRA 3.0. Dicha aplicación debe estar disponible en cualquier lugar, tengamos o no conexión a internet. Para cumplir con el requisito de poder acceder a los datos sin tener conexión, se ha creado una base de datos interna y otra externa, que explicaremos con más detalle en los siguientes apartados.

Uno de los grandes objetivos es crear una aplicación intuitiva, fácil de usar y que sea agradable para el usuario. Para ello, decidimos que no era necesario mostrar toda la información que mostraba la versión online, ya que podría saturar al usuario con tantos datos. La información que se mostrará al usuario, siempre y cuando esté disponible, con cada palabra que busque será la siguiente:

- 1) **Significado:** Se mostrará los diferentes significados de dicha palabra, junto a los ejemplos disponibles y las palabras asociadas a dichos significados (palabras asociadas al mismo sentido) indicando si se refiere a nombres, verbos, adjetivos o adverbios.
- 2) **Traducción:** Se decidió añadir esta funcionalidad que la aplicación online no tenía por sí misma, para darle al usuario la oportunidad de utilizar la aplicación como un traductor, sin la necesidad de mostrar todo el significado de la palabra.
- 3) **Obtener sinónimos, antónimos, hipónimos e hiperónimos:** Una vez mostrada la información de alguna de las anteriores funciones, el usuario tiene la posibilidad de obtener los sinónimos, antónimos, hipónimos e hiperónimos haciendo click encima de la información y seleccionando la opción que quiera del menú que aparece.

Toda la información mostrada por pantalla, estará disponible en cinco idiomas diferentes: inglés, español, euskera, catalán y gallego. En la opción de obtener los sinónimos, antónimos, hipónimos o hiperónimos, se mostrarán en el mismo idioma que el seleccionado para obtener los significados o las traducciones.

En el caso de la base de datos offline, se decidió no incluirla en la aplicación directamente, debido a que puede que haya personas que no les interese descargársela. Para ellos, se creó otra aplicación más sencilla que se encargase de instalar la base de datos en la tarjeta SD del dispositivo móvil, que llamaremos MCRBD 3.0. Esta decisión se tomó tras analizar el funcionamiento de otras aplicaciones que actualmente se encontraban en el mercado.

5.1 Diseño de pantallas

Para llevar a cabo las operaciones anteriormente descritas, se diseñaron una serie de pantallas que reflejarían de una manera, más o menos real, el estado final de las interfaces de la aplicación. A la hora de diseñarlas, se pensó en crear una interfaz sencilla, intuitiva y clara, que ayudase al usuario a realizar las operaciones que considerase oportunas sin perder demasiado tiempo aprendiendo el funcionamiento de la aplicación.

La aplicación principal, encargada de mostrar la información referente a las palabras, tendrá dos interfaces:

- **Principal:** contará con un botón desplegable para seleccionar el idioma de origen (idiomaO), el idioma de destino (idiomaD) y la funcionalidad (funcio) que queremos utilizar. Para escribir la palabra, tendremos un cuadro de texto junto a un botón (Enviar), que será el encargado de indicar que empiece la búsqueda. En la parte inferior se mostrará toda la información referente a esa palabra dentro de diferentes cuadros de texto. En la figura 12 podemos observar el lugar en el que estarán colocados los elementos dentro de la pantalla.

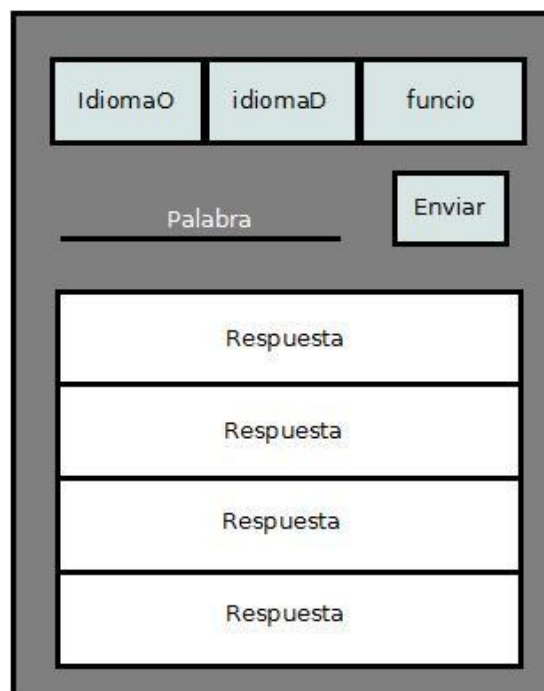


Figura 12: Diseño de la interfaz principal.

- **Secundaria:** en ella mostrares los resultados asociados a la opción seleccionada, ya sean sinónimos, antónimos, hipónimos o hiperónimos. Constará de dos cuadros de texto, uno mostrará la opción seleccionada y, el otro, el resultado asociado a dicha opción. En la figura 13 podemos observar el lugar en el que estarán colocados los elementos dentro de la pantalla.

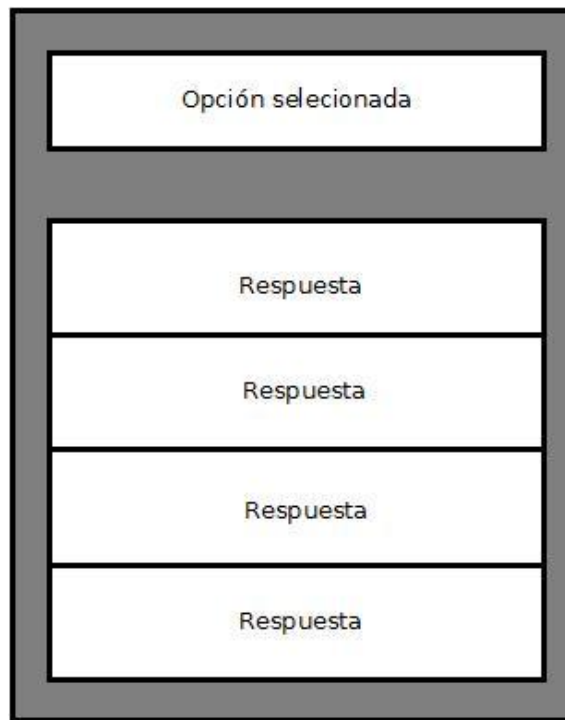


Figura 13: Diseño de la interfaz secundaria.

La aplicación encargada de instalar la base de datos solamente dispone de una interfaz:

- **Principal:** en ella se mostrará al usuario un botón para comenzar la instalación, un cuadro de texto para mostrar información y una barra de progreso para indicar el estado de instalación. En la figura 13 podemos observar el lugar en el que estarán colocados los elementos dentro de la pantalla.

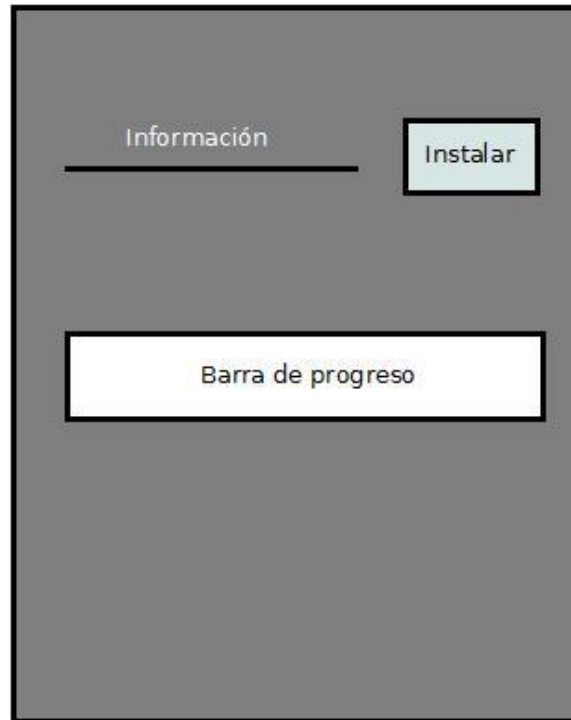


Figura 14: Diseño de la interfaz principal de la instalación de la base de datos.

5.2 Diagrama de casos de uso MCRA 3.0

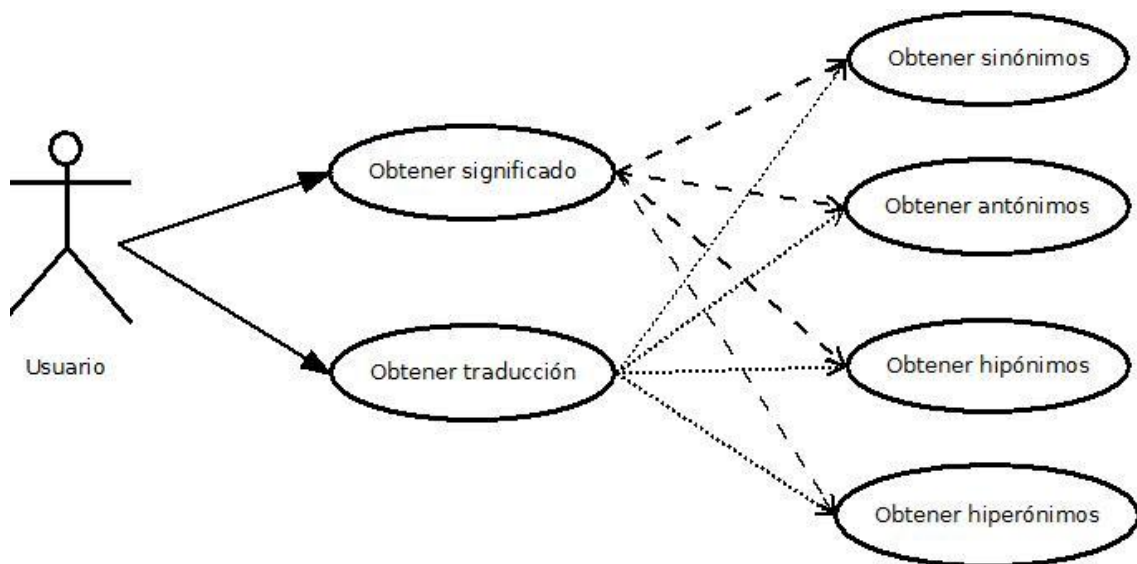


Figura 15: Diagrama de casos de uso de la aplicación MCRA 3.0.

5.3 Casos de uso reales

En este apartado, expondremos los casos de uso reales de la aplicación MCRA 3.0 junto a su correspondiente diseño de pantalla, que podemos apreciar en las diferentes figuras.

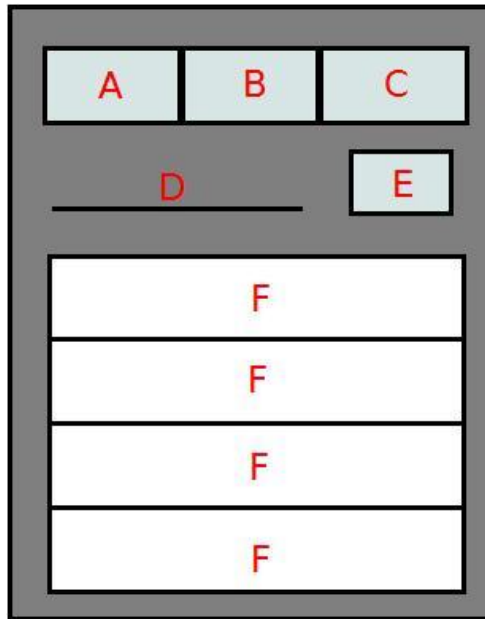


Figura 16: Pantalla principal de la aplicación MCRA 3.0.

Obtener significado

Actores

- Usuario.

Descripción

- El usuario escribe la palabra que quiere buscar en el cuadro de texto *D*, pulsa el botón *A* para seleccionar el idioma de la palabra que ha introducido, pulsa el botón *B* para seleccionar el idioma en el que quiere que aparezca el resultado y, finalmente, pulsa el botón *C* para seleccionar la operación de obtener el significado de la palabra. Tras seleccionar todas las opciones, pulsa el botón *E* para que la aplicación muestre por pantalla, y agrupadas por los sentidos, el significado de la palabra, las palabras, ejemplos de la misma y si esa palabra era un verbo, nombre, adjetivo o adverbio.

Precondición

- El dispositivo móvil debe tener conexión a internet o una base de datos guardada en el dispositivo móvil para poder realizar las consultas.

Post-condiciones**Escenario principal**

- 1) Usuario: Escribe la palabra que va a buscar, selecciona el idioma de la palabra, en qué idioma se va a presentar el resultado y la operación de que nos muestre el significado de la palabra.
- 2) Sistema: Muestra el significado de la palabra en el idioma en el que se ha indicado.

Escenario alternativo:

- a) No tenemos información de esa palabra, por lo que se mostrará un mensaje indicando dicha información
- b) No se ha recibido ninguna información, ya sea porque se ha perdido la conexión o a habido un error, por lo que se le indicara al usuario que no ha podido mostrarse ninguna información al respecto.

Obtener traducción**Actores**

- Usuario.

Descripción

- El usuario escribe la palabra que quiere buscar en el cuadro de texto *D*, pulsa el botón *A* para seleccionar el idioma de la palabra que ha introducido, pulsa el botón *B* para seleccionar el idioma en el que quiere que aparezca el resultado y, finalmente, pulsa el botón *C* para seleccionar la operación de obtener las traducciones de la palabra. Tras seleccionar todas las opciones, pulsa el botón *E* para que la aplicación muestre por pantalla, y agrupadas por los sentidos, las palabras traducidas y si esa palabra era un verbo, nombre, adjetivo o adverbio.

Precondición

- El dispositivo móvil debe tener conexión a internet o una base de datos guardada en el dispositivo móvil para poder realizar las consultas.

Post-condiciones**Escenario principal**

- 1) Usuario: Escribe la palabra que va a buscar, selecciona el idioma de la palabra, en que idioma se va a presentar el resultado y la operación de que nos muestre la traducción de la palabra.
- 2) Sistema: Muestra la traducción de la palabra en el idioma en que se ha indicado.

Escenario alternativo:

- a) No se ha encontrado ninguna traducción para la palabra introducida, por lo que mostraremos dicha información por pantalla.
- b) No se ha recibido ninguna información, ya sea porque se ha perdido la conexión o a habido un error, por lo que se le indicará al usuario que no ha podido mostrarse ninguna información al respecto.

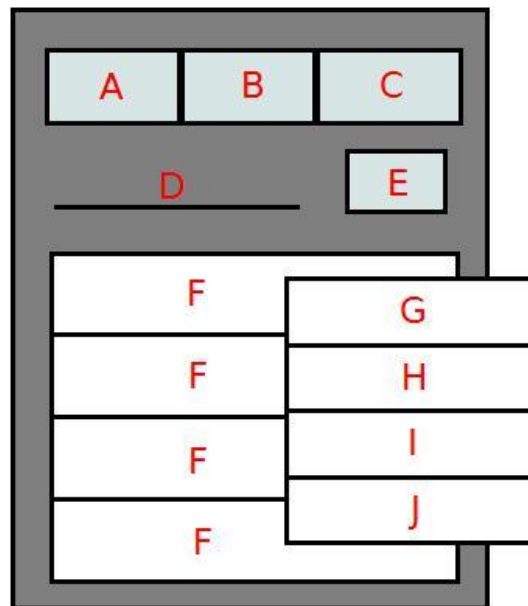


Figura 17: Menú con las opciones disponibles.

Obtener sinónimos**Actores**

- Usuario.

Descripción

El usuario selecciona uno de los resultados obtenidos (F) por el caso de uso obtener traducción u obtener significado y aparecerá un menú en el que seleccionará la operación G, encargada de mostrar los sinónimos. Tras seleccionarla, se mostrará por

pantalla los diferentes sinónimos asociados a dicha palabra, agrupadas por los sentidos.

Precondición

- El dispositivo móvil debe tener conexión a internet o una base de datos guardada en el dispositivo móvil para poder realizar las consultas.

Post-condiciones**Escenario principal**

- 1) Usuario: Selecciona la palabra o el significado y elige la opción de sinónimos del menú que se le muestra.
- 2) Sistema: Muestra los sinónimos de la palabra o del significado seleccionado.

Escenario alternativo:

- a) No se ha encontrado ningún sinónimo para la palabra introducida por lo que mostraremos dicha información por pantalla.
- b) No se ha recibido ninguna información, ya sea porque se ha perdido la conexión o a habido un error, por lo que se le indicara al usuario que no ha podido mostrarse ninguna información al respecto.

Obtener antónimos**Actores**

- Usuario.

Descripción

- El usuario selecciona uno de los resultados obtenidos (*F*) por el caso de uso obtener traducción u obtener significado y aparecerá un menú en el que seleccionará la operación *H*, encargada de mostrar los antónimos. Tras seleccionarla, se mostrará por pantalla los diferentes antónimos asociados a dicha palabra, agrupadas por los sentidos.

Precondición

- El dispositivo móvil debe tener conexión a internet o una base de datos guardada en el dispositivo móvil para poder realizar las consultas.

Post-condiciones**Escenario principal**

- 1) Usuario: Selecciona la palabra o el significado y elige la opción de antónimos del menú que se le muestra.
- 2) Sistema: Muestra los antónimos de la palabra o del significado seleccionado.

Escenario alternativo:

- a) No se ha encontrado ningún antónimo para la palabra introducida, por lo que mostraremos dicha información por pantalla.
- b) No se ha recibido ninguna información, ya sea porque se ha perdido la conexión o a habido un error, por lo que se le indicara al usuario que no ha podido mostrarse ninguna información al respecto.

Obtener hiperónimo**Actores**

- Usuario.

Descripción

- El usuario selecciona uno de los resultados obtenidos (*F*) por el caso de uso obtener traducción u obtener significado y aparecerá un menú en el que seleccionará la operación *I*, encargada de mostrar los hiperónimos. Tras seleccionarla, se mostrará por pantalla los diferentes hiperónimos asociados a dicha palabra, agrupadas por los sentidos.

Precondición

- El dispositivo móvil debe tener conexión a internet o una base de datos guardada en el dispositivo móvil para poder realizar las consultas.

Post-condiciones**Escenario principal**

- 1) Usuario: Selecciona la palabra o el significado y elige la opción de hiperónimo del menú que se le muestra.
- 2) Sistema: Muestra los hiperónimo de la palabra o del significado seleccionado.

Escenario alternativo:

- a) No se ha encontrado ningún hiperónimo para la palabra introducida por lo que mostraremos dicha información por pantalla.
- b) No se ha recibido ninguna información, ya sea porque se ha perdido la conexión o a habido un error, por lo que se le indicara al usuario que no ha podido mostrarse ninguna información al respecto.

Obtener hipónimo

Actores

- Usuario.

Descripción

- El usuario selecciona uno de los resultados obtenidos (*F*) por el caso de uso obtener traducción u obtener significado y aparecerá un menú en el que seleccionará la operación *J*, encargada de mostrar los hipónimos. Tras seleccionarla, se mostrará por pantalla los diferentes hipónimos asociados a dicha palabra, agrupadas por los sentidos.

Precondición

- El dispositivo móvil debe tener conexión a internet o una base de datos guardada en el dispositivo móvil para poder realizar las consultas.

Post-condiciones

Escenario principal

- 1) Usuario: Selecciona la palabra o el significado y elige la opción de hipónimo del menú que se le muestra.
- 2) Sistema: Muestra los hipónimos de la palabra o del significado seleccionado.

Escenario alternativo:

- a) No se ha encontrado ningún hipónimo para la palabra introducida, por lo que mostraremos dicha información por pantalla.
- b) No se ha recibido ninguna información, ya sea porque se ha perdido la conexión o a habido un error, por lo que se le indicara al usuario que no ha podido mostrarse ninguna información al respecto.

5.4 Diagrama de casos de uso MCRBD 3.0

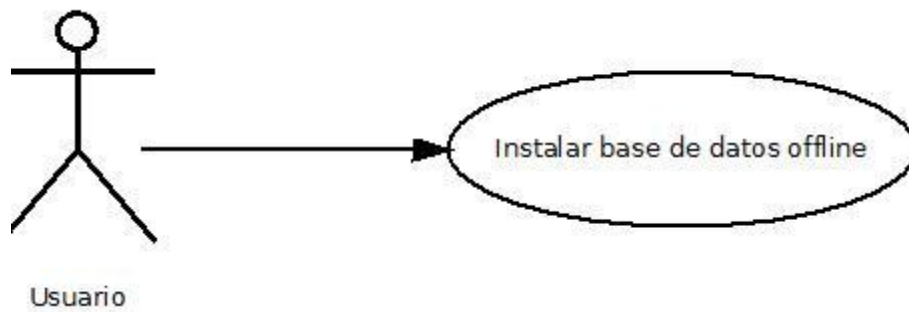


Figura 18: Diagrama de casos de uso de la aplicación MCRBD 3.0.

5.4.1 Caso de uso reales

En este apartado, expondremos los casos de uso reales de la aplicación MCRBD 3.0 junto a su correspondiente diseño de pantalla, que podemos apreciar en las diferentes figuras.

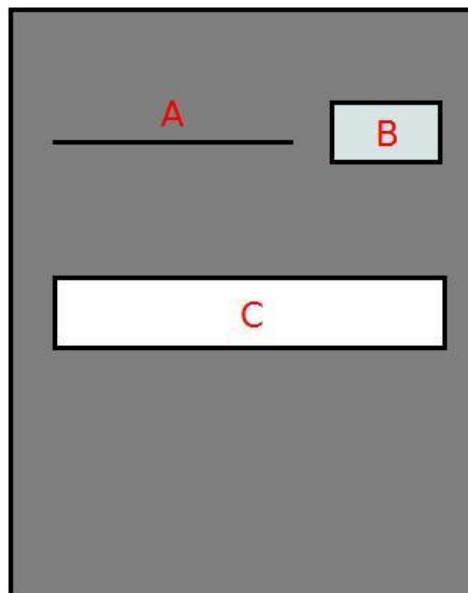


Figura 19: Pantalla principal de la aplicación MCRBD 3.0.

Instalar base de datos offline

Actores

- Usuario.

Descripción

- Pulsamos el botón *B* para comenzar la instalación de la base de datos en la tarjeta SD. Si no disponemos de espacio suficiente, se mostrará un mensaje en el cuadro de texto *A*, indicando dicho problema o, si hay espacio, indicando que la instalación ha comenzado. La barra de progreso *C* indicará el progreso de la copia.

Precondición

- Tener una tarjeta SD en el dispositivo móvil.

Post-condiciones

Escenario principal

- 1) Usuario: Selecciona el lugar en el que quiere almacenar la base de datos.
- 2) Sistema: Crea la base de datos en el lugar especificado.

Escenario alternativo

- a) Si no hay espacio suficiente, se le indicará al usuario que no se ha podido instalar la base de datos.

5.5 Modelo de datos

Actualmente el MCR 3.0 dispone de una gran base de datos, de la que explicaremos las tablas principales. El xxx asociada a cada tabla se refiere a las tablas de los diferentes idiomas: wei eng-30 to ili (inglés), wei spa-30 to ili (español), etc. Además, en cada tabla explicaremos los atributos más relevantes de cada uno de ellas.

5.5.1 Base de datos MCR 3.0

5.5.1.1 wei xxx-30 to ili

Esta tabla conecta el ILI, en formato *ili-30-xxxxxx-y* (las *x* indican el offset del *synset*, y las *y* representan la categoría gramatical o *Part-Of-Speech* (PoS)), con su correspondiente número de *synset* (offset). En la figura 20, podemos ver la tabla junto a sus diferentes atributos.



Figura 20: Tabla wei_xxx_30_to_ili.

A continuación, podemos observar un ejemplo de los datos que encontramos en dicha tabla:

- **iliOffset:** ili-30-00001740-a
- **pos:** a
- **offset:** cat-30-00001740-a
- **iliWnld:** eng-30
- **cscs:** 1

5.5.1.2 *wei xxx-30 relation:*

Esta tabla contiene todas las relaciones del wordnet. Cada registro (que es la instancia de una relación), guarda un código que indica el tipo de relación (que se almacena de la tabla *wei relations*), la dirección de la relación (*synset origen* y *synset destino*), el valor de confianza y el wordnet del que proviene. En la figura 21, podemos ver la tabla, junto a sus diferentes atributos.

mcr30.wei_cat-30_relation	
relation	: smallint(6)
sourceSynset	: char(17)
sourcePos	: char(1)
targetSynset	: char(17)
targetPos	: char(1)
csc	: float
method	: char(2)
version	: char(1)
wnSource	: char(4)

Figura 21: Tabla *wei_cat-30_relation*.

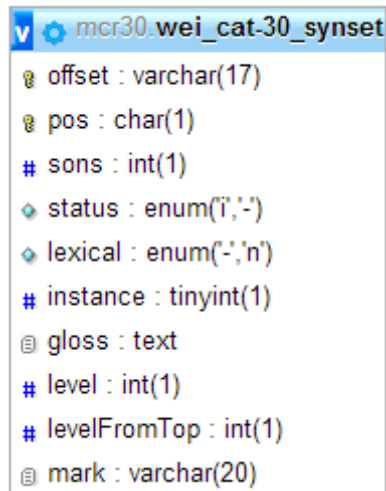
A continuación, podemos observar un ejemplo de los datos que encontramos en dicha tabla:

- **relation:** 64
- **sourceSynset:** cat-30-02772310-v
- **sourcePos:** v
- **targetSynset:** cat-30-13450417-n
- **targetPos:** n
- **csc:** 99
- **method:** mn
- **versión:** 1
- **wnSource:** en30

5.5.1.3 *wei xxx-30 synset:*

Aquí se almacena la información acerca del *synset*: el identificador, el número de descendientes, la glosa, el número máximo de niveles en su jerarquía, el nivel en el que se encuentra (contando desde arriba), y, finalmente, una marca (opcional) y un

comentario del synset (opcional). En la figura 22, podemos ver la tabla, junto a sus diferentes atributos.



Attribute	Data Type
offset	varchar(17)
pos	char(1)
sons	int(1)
status	enum('i','-')
lexical	enum('-', 'n')
instance	tinyint(1)
gloss	text
level	int(1)
levelFromTop	int(1)
mark	varchar(20)

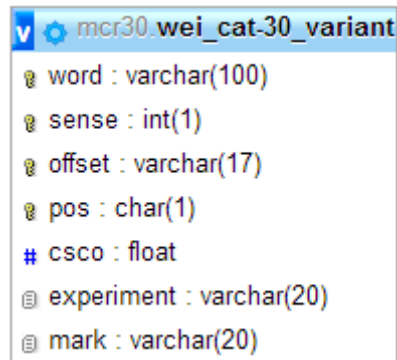
Figura 22: Tabla wei_cat-30_synset.

A continuación, podemos observar un ejemplo de los datos que encontramos en dicha tabla:

- **offset:** cat-30-00001740-n
- **pos:** n
- **sons:** 82319
- **status:** -
- **lexical:** -
- **instance:** 0
- **gloss:** Realitat considerada per abstracció com a unitat (amb o sense vida)
- **level:** 19
- **levelFromTop:** 0
- **mark:** -----

5.5.1.4 wei xxx-30 variant

Aquí se almacenan todos los variant. Cada registro representa un único variant y contiene la siguiente información: el variant, el sentido de la palabra, el identificador de synset, un valor de confianza, el experimento del que proviene (opcional), y finalmente la marca (opcional) y el comentario del variant (opcional). En la figura 23, podemos ver la tabla, junto a sus diferentes atributos.



The screenshot shows a table schema for 'mcr30.wei_cat-30_variant'. The columns and their data types are:

- word : varchar(100)
- sense : int(1)
- offset : varchar(17)
- pos : char(1)
- # cscs : float
- experiment : varchar(20)
- mark : varchar(20)

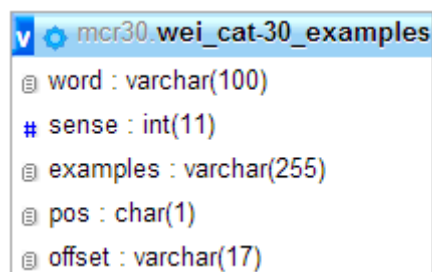
Figura 23: Tabla wei_cat-30_variant.

A continuación, podemos observar un ejemplo de los datos que encontramos en dicha tabla:

- **word:** apte
- **sense:** 1
- **offset:** cat-30-00001740-a
- **pos:** a
- **cscs:** 99
- **experiment:** NULL
- **mark:** -----

5.5.1.5 wei xxx-30 examples

En esta tabla se listan todos los ejemplos del wordnet. Cada ejemplo está identificado por el número de synset, la palabra y el sentido. En la figura 24, podemos observar la tabla, junto a sus diferentes atributos.



The screenshot shows a table schema for 'mcr30.wei_cat-30_examples'. The columns and their data types are:

- word : varchar(100)
- # sense : int(11)
- examples : varchar(255)
- pos : char(1)
- offset : varchar(17)

Figura 24: Tabla wei_cat-30_example.

A continuación, podemos observar un ejemplo de los datos que encontramos en dicha tabla:

- **word:** medicina
- **sense:** 1

- **examples:** L'objectiu és l'exercici de la medicina en el sistema sanitari públic i privat un cop acabada la formació especialitzada corresponent.
- **pos:** n
- **offset:** cat-30-06043075-n

5.5.2 Nueva base de datos

Al principio, se pensó en utilizar la misma base de datos, pero tras varias pruebas, se vio que el espacio que ocupaban dichas tablas era excesivo para una aplicación móvil, por lo que se decidió reducir la cantidad de tablas y compactar más la información de las tablas anteriores, así como eliminar cierta información, dando lugar al siguiente resultado:

5.5.2.1 Wei_xxx_30

Esta tabla es la versión reducida de la tabla `wei_xxx_30_variant`, debido a que no nos interesaban todos los atributos que contenía y, además, podíamos reducir el espacio que ocupaban al eliminarlos. Para ahorrarnos consultas y agilizar la obtención de información, se decidió añadir los ejemplos de la tabla `wei_xxx_30_example` a la tabla `wei_xxx_30`.

La tabla está compuesta por diferentes palabras, el código del synset asociado a ella (offset), qué tipo de palabra es (nombre, verbo, adjetivo o adverbio) y el ejemplo asociado a dicha palabra. Para ahorrarnos la tabla de traducción de synset (`wei_xxx_30_to_ili`), de un idioma al ILI para después traducirla al idioma destino, se decidió que todos los idiomas tendrían el valor del ILI por defecto en todos sus synset (el valor del offset pasaría de `xxx-30-00001740-a` a `ili-30-00001740-a`). Esto también nos beneficiaba a la hora de realizar las consultas, puesto que así podíamos acceder directamente de la tabla de un idioma a otra, sin tener que pasar por la tabla anteriormente comentada. En la figura 25, podemos ver la tabla, junto a sus diferentes atributos.

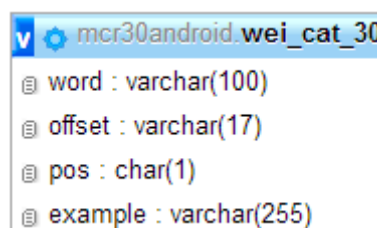


Figura 25: Tabla `wei_cat_30`.

A continuación, podemos observar un ejemplo de los datos que encontramos en dicha tabla:

- **word:** entitat
- **offset:** ili-30-00001740-n
- **pos:** n
- **example:** L'ontologia té com a objectiu descriure o posar les categories i relacions bàsiques entre les diferents entitats.

5.5.2.2 Wei_xxx_30_synset

Esta tabla es la versión reducida de la tabla wei_xxx_30_synset, debido a que la mayoría de los atributos no nos interesaban y solo era necesario dejar los dos que son útiles para la aplicación, además de ahorrar espacio. Contiene el código del synset (offset) y el significado asociado a ese synset (gloss). Las únicas diferencias con la tabla anterior es que, en ésta, eliminamos las columnas que no necesitábamos, para así reducir el espacio que ocupaban y que los offsets pasarían a ser de tipo ILI para ahorrarnos consultas y tablas, como hemos explicado anteriormente. En la figura 26, podemos ver la tabla, junto a sus diferentes atributos.

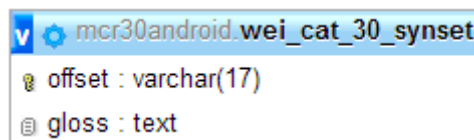


Figura 26: Tabla wei_cat_30_synset.

A continuación, podemos observar un ejemplo de los datos que encontramos en dicha tabla:

- **offset:** ili-30-00001740-n
- **gloss:** Realitat considerada per abstracció com a unitat (amb o sense vida).

5.5.2.3 Wei_30_relation

Esta tabla es idéntica a la que teníamos anteriormente, por lo que todos los atributos son los mismos. La única diferencia es que en la base de datos anterior, cada idioma tenía su propia tabla de relaciones, que era idéntica para todos los idiomas, salvo por el código del synset. Para no tener una tabla de relaciones para cada idioma (puesto que era una de las que más ocupaba), se modificó los synset para que todos fuesen de

tipo ILLI y así servir a todas las tablas por igual, tal y como habíamos comentado anteriormente. En la figura 27, podemos ver la tabla, junto a sus diferentes atributos.



Attribute	Type
relation	varchar(17)
sourceSynset	varchar(17)
sourcePos	varchar(1)
targetSynset	varchar(17)
targetPos	varchar(1)
wnSource	varchar(4)

Figura 27: Tabla wei_30_relation.

A continuación, podemos observar un ejemplo de los datos que encontramos en dicha tabla:

- **relation:** 1
- **sourceSynset:** ili-30-00001740-a
- **sourcePos:** a
- **targetSynset:** ili-30-05200169-n
- **targetPos:** n
- **wnSource:** en30

En la figura 28 podemos observar el estado final de la base de datos, junto a las relaciones de cada una de ellas.

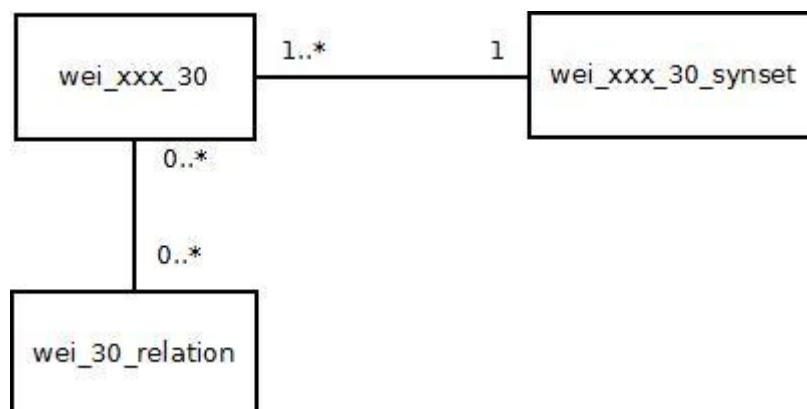


Figura 28: Relaciones de la base de datos.

Wei_xxx_30_synset contiene el significado de todos los synset que podemos encontrar en la tabla wei_xxx_30. Como cada synset es único, solo tendrá un significado asociado a él. Por el contrario, la tabla wei_xxx_30 contiene palabras que tienen el mismo

synset, por lo que todas compartirán el mismo significado (siempre y cuando se encuentre almacenado en la base de datos). Esta mecánica podemos apreciarla con el siguiente ejemplo en el que buscamos el significado de la palabra “casa”:

- **offset:** ili-30-03195485-n
- **gloss:** aposentos temporales
- **word:** alojamiento, casa, domiciliación, pensión y trincheras
- **offset:** ili-30-03259505-n
- **gloss:** vivienda en la que alguien está viviendo
- **word:** casa, domicilio, habitáculo, morada, piso y residencia

Tal y como vemos en el ejemplo, un synset solo puede tener un significado y las palabras asociadas a ese synset compartirán ese significado.

Wei_30_relation contiene todas las relaciones que pueden tener los synset de la tabla wei_xxx_30. En este caso, no todos los synset tienen una relación con otros synset, mientras que otros pueden tener diferentes relaciones entre ellos. Este funcionamiento podemos apreciarlo con el siguiente ejemplo en el que buscamos los sinónimos de uno de los sentidos de la palabra “ugly”:

- **offset:** ili-30-00221627-a
- **word:** grotesque, monstrous
- **offset:** ili-30-00221934-a
- **word:** hideous, repulsive Ahora seleccionamos los sinónimos:
- **offset:** ili-30-00217728-a
- **word:** beautiful

Tal y como vemos en el ejemplo, un synset puede tener más de una relación, dando lugar a una gran variedad de opciones. También puede darse el caso en el que un synset no tenga ningún tipo de relación con una opción concreta, en este caso, sucedería si obtenemos los hipónimos de dicho synset, por lo que no todos los synset tienen las mismas relaciones.

6 Análisis

Para realizar las funciones, sólo es necesario indicarle qué operación queremos realizar, el idioma en el que se encuentra la palabra y el idioma destino en el que mostraremos la respuesta. Los resultados obtenidos están compuestos por:

- **Offset:** atributo que indica el sentido de los significados, ejemplos y palabras que vamos a mostrar. En nuestro caso, lo utilizamos para agrupar cada uno de los atributos y mostrarlos todos dentro del marco, por lo que no se le mostrará este código al usuario.
- **Pos:** atributo que nos indica de qué tipo es el sentido que estamos mostrando: verbo, nombre, adverbio o adjetivo. Es el atributo que se mostrara primero y estará marcado en negrita.
- **Gloss:** atributo que contiene los significados asociados a los sentidos. Es el segundo atributo que se mostrará.
- **Ejemplo:** atributo que contiene los ejemplos asociados a los sentidos y palabras. Es el tercer atributo que se mostrará (siempre y cuando dispongamos de ejemplos), estará en cursiva, con un tono grisáceo y precedido por un “*”.
- **Word:** atributo que contendrá todas las palabras asociadas a los sentidos. Es el cuarto y último atributo que se mostrará. Estará precedido por dos “**” y con un tono grisáceo.

Se decidió agrupar los atributos, anteriormente descritos, mediante el offset, para que la información quedase más compacta y que todo lo que tenga el mismo sentido se muestre a la vez. En el caso de no agruparla, nos mostraba cada palabra (juntos a sus atributos) en un cuadro de texto diferente, dando la impresión de ser una aplicación menos funcional, debido a que contenía mucha información repetida. A continuación comprobaremos, de una forma más visual, la diferencia entre ambas opciones, usando uno de los sentidos de la palabra “house” como ejemplo:

Sin agrupar mediante el sentido:

- **n**
 - a building where theatrical performances or motion-picture shows can be presented
 - **house
- **n**
 - a building where theatrical performances or motion-picture shows can be presented
 - * the house was full
 - **theater

- n
 - a building where theatrical performances or motion-picture shows can be presented

****theatre**

Agrupandolos mediante el sentido:

- n
 - a building where theatrical performances or motion-picture shows can be presented
 - * the house was full
 - ****house, **theater, **theatre**

Tal y como podemos apreciar en el ejemplo, sin agrupar todo por su sentido, tenemos información repetida que estorbaría al usuario, mientras que al agruparla, quedaría mucho más limpio y entendible para el usuario.

En el caso de obtener los sinónimos, antónimos, hipónimos e hiperónimos, se decidió no mostrar los significados, ni los ejemplos asociados a ese sentido, ya que con mostrar la palabra era suficiente. Si el usuario quiere obtener el significado o los ejemplos, solamente tendría que acceder a la pantalla anterior y realizar otra búsqueda. Al igual que en el caso anterior, las palabras resultantes, así como su tipo, son agrupadas mediante su offset, debido a que si no pasaría lo mismo que hemos descrito arriba.

6.1 Contratos

En este apartado, analizaremos los diferentes contratos de la aplicación, junto a sus diagramas de secuencia del sistema, representados en las diferentes figuras.

6.1.1 Obtener significados

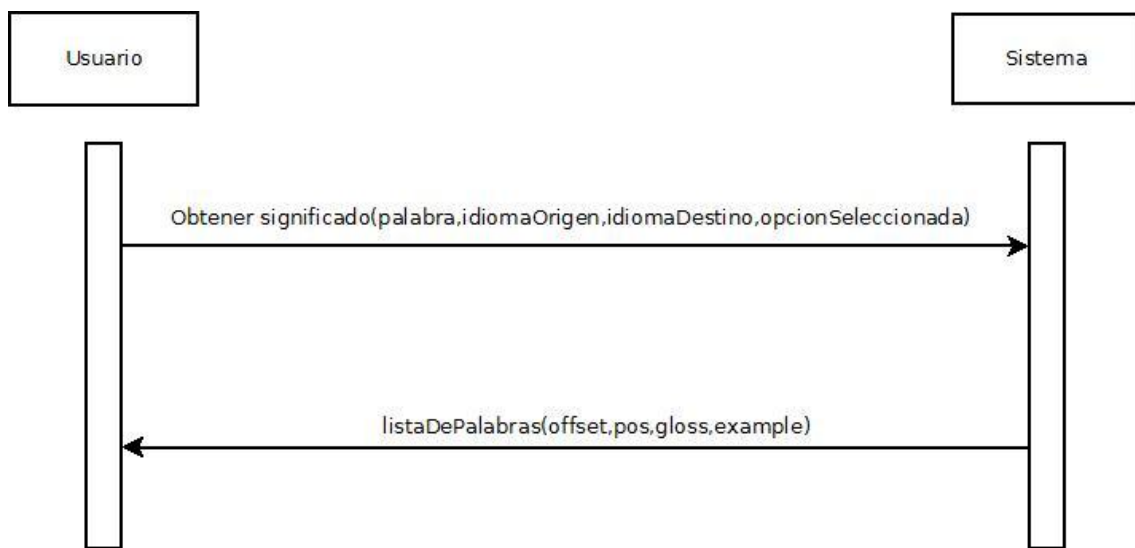


Figura 29: Obtener significados.

Nombre:

- Obtener significado.

Responsabilidades:

- Si la consulta es offline, que se disponga de una base de datos instalada en el teléfono móvil; y si la consulta es online, que se disponga de conexión a internet para realizar la consulta.

Pre-condición:

Post-condición:

Salida:

- Visualización de los diferentes significados de la palabra (en el idioma seleccionado), los ejemplos disponibles y las palabras asociadas ha dicho significado, junto al tipo de palabra.

6.1.2 Obtener traducción

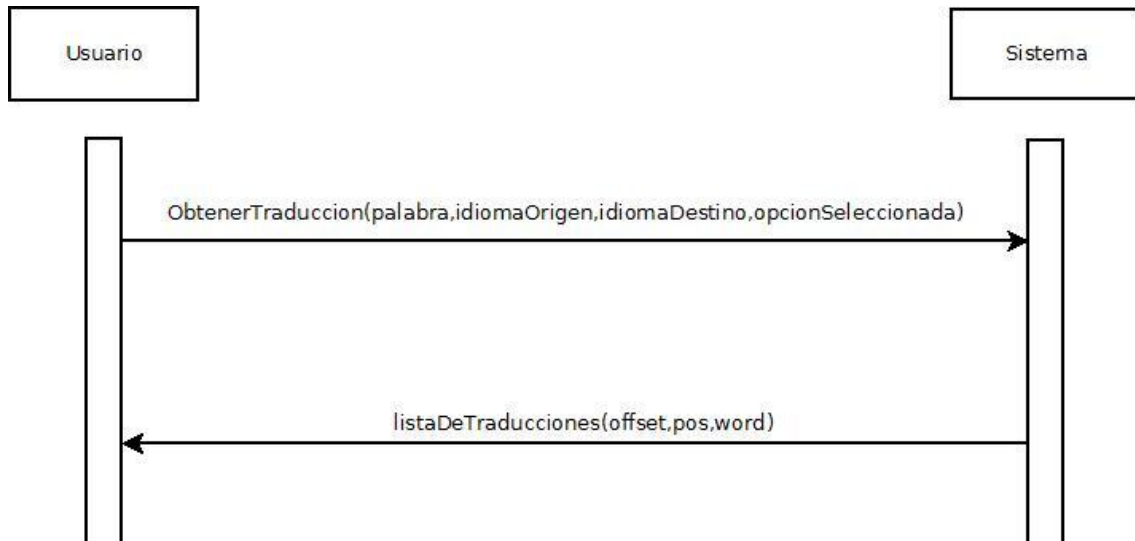


Figura 30: Obtener traducción.

Nombre:

- Obtener traducción.

Responsabilidades:

- Si la consulta es offline, que se disponga de una base de datos instalada en el teléfono móvil; y si la consulta es online, que se disponga de conexión a internet para realizar la consulta.

Pre-condición:

Post-condición

Salida:

- Visualización de las palabras traducidas en el idioma seleccionado junto a su tipo.

6.1.3 Obtener sinónimos



Figura 31: Obtener sinónimos.

Nombre:

- Obtener sinónimos.

Responsabilidades:

- Si la consulta es offline, que se disponga de una base de datos instalada en el teléfono móvil; y si la consulta es online, que se disponga de conexión a internet para realizar la consulta.

Pre-condición:

- Seleccionar uno de los significados (obtenidos de la funcionalidad “Obtener significados”) o palabras traducidas (obtenidas de la funcionalidad “Obtener traducción”) de los contratos anteriores.

Post-condición

Salida:

- Visualización de los sinónimos asociados a dicha palabra junto a su tipo.

6.1.4 Obtener antónimos

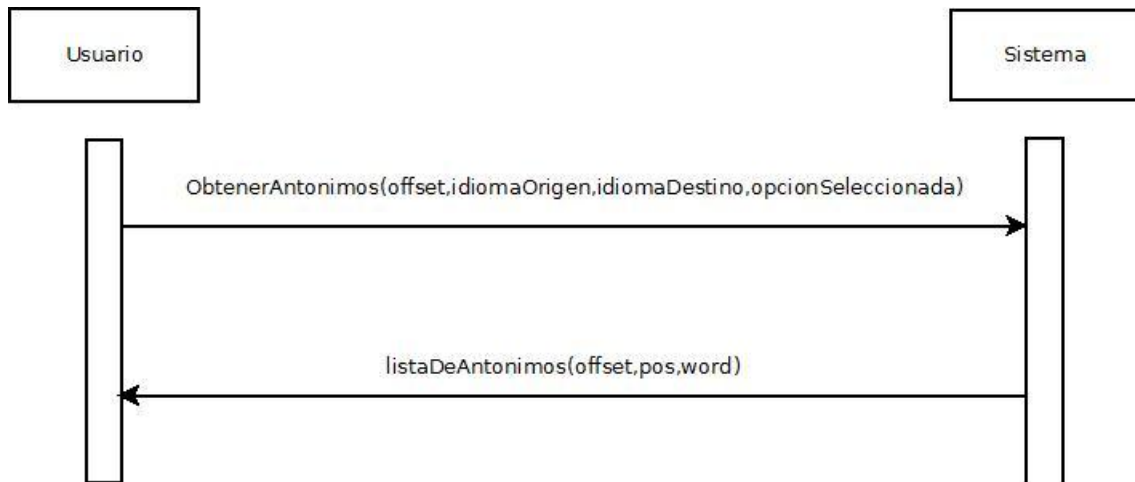


Figura 32: Obtener antónimos.

Nombre:

- Obtener antónimos.

Responsabilidades:

- Si la consulta es offline, que se disponga de una base de datos instalada en el teléfono móvil; y si la consulta es online, que se disponga de conexión a internet para realizar la consulta.

Pre-condición:

- Seleccionar uno de los significados (obtenidos de la funcionalidad "Obtener significados") o palabras traducidas (obtenidas de la funcionalidad "Obtener traducción") de los contratos anteriores.

Post-condición**Salida:**

- Visualización de los antónimos asociados a dicha palabra junto a su tipo.

6.1.5 Obtener hipónimos



Figura 33: Obtener hipónimos.

Nombre:

- Obtener hipónimos.

Responsabilidades:

- Si la consulta es offline, que se disponga de una base de datos instalada en el teléfono móvil; y si la consulta es online, que se disponga de conexión a internet para realizar la consulta.

Pre-condición:

- Seleccionar uno de los significados (obtenidos de la funcionalidad “Obtener significados”) o palabras traducidas (obtenidas de la funcionalidad “Obtener traducción”) de los contratos anteriores.

Post-condición

Salida:

- Visualización de los hipónimos asociados a dicha palabra junto a su tipo.

6.1.6 Obtener hiperónimo

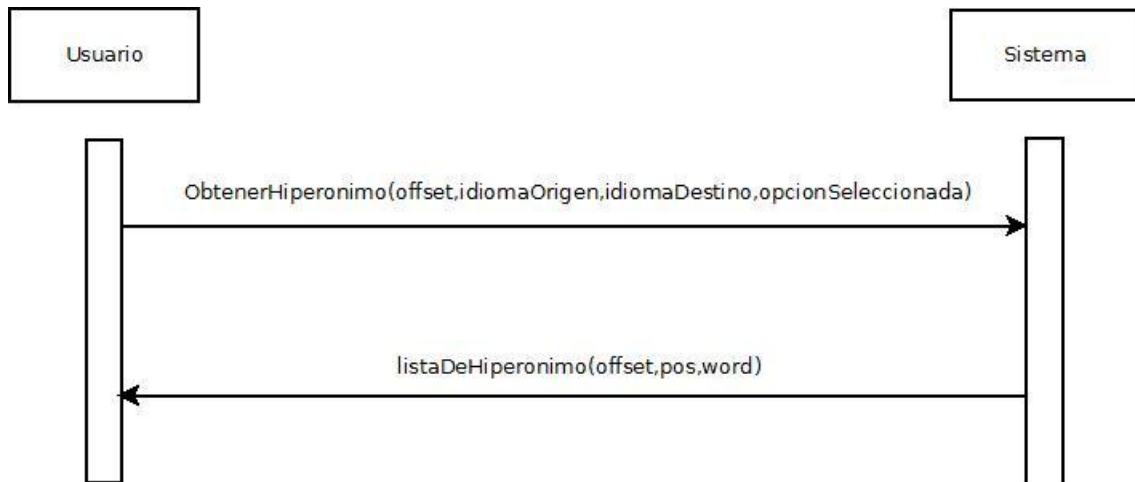


Figura 34: Obtener hiperónimos.

Nombre:

- Obtener hiperónimo.

Responsabilidades:

- Si la consulta es offline, que se disponga de una base de datos instalada en el teléfono móvil; y si la consulta es online, que se disponga de conexión a internet para realizar la consulta.

Pre-condición:

- Seleccionar uno de los significados (obtenidos de la funcionalidad "Obtener significados") o palabras traducidas (obtenidas de la funcionalidad "Obtener traducción") de los contratos anteriores.

Post-condición**Salida:**

- Visualización de los hiperónimos asociados a dicha palabra junto a su tipo.

6.1.7 Instalar base de datos

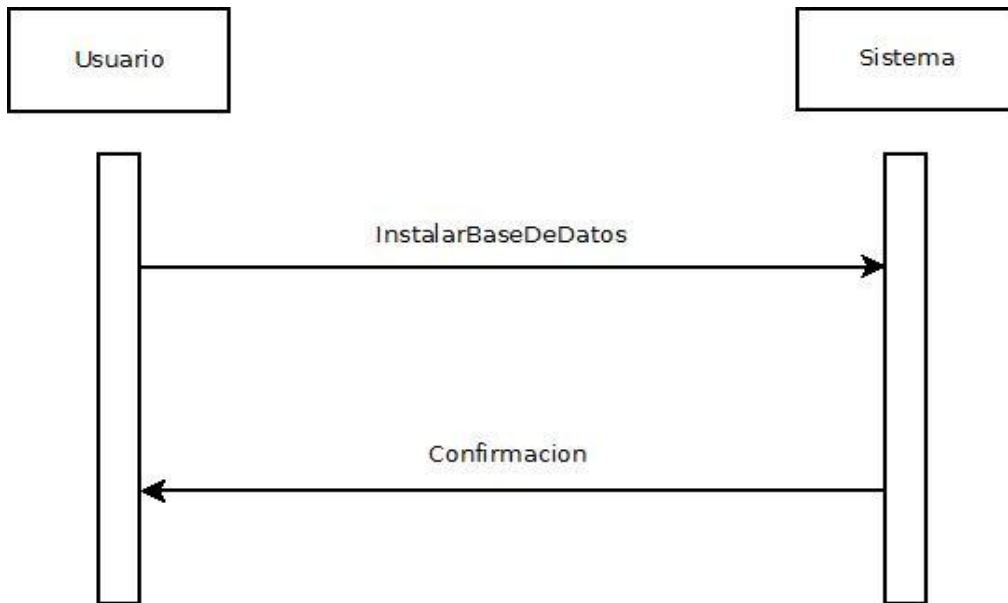


Figura 35: Instalar Base de datos.

Nombre:

- Instalar base de datos.

Responsabilidades:

- Tener una tarjeta SD en la que instalar la base de datos.

Pre-condición:

Post-condición

Salida:

- Obtener un mensaje de confirmación de que la base de datos se ha instalado correctamente.

7 Diseño

En este apartado, mostraremos los diagramas de secuencia de las operaciones más relevantes, así como el diseño realizado para ellas y otras operaciones no reflejadas en los diagramas (al no ser tan complejas como para tener que realizarlos).

7.1 Diagramas de secuencia

En las siguientes figuras, podemos comprobar los diagramas de secuencia para las operaciones:

- Obtener significado.
- Obtener traducción
- Obtener sinónimos, antónimos, hipónimos o hiperónimos. Estas opciones comparten el mismo diagrama, debido a que la forma de obtenerlos es idéntica en todos ellos, cambiando únicamente el valor de algunos parámetros.

7.1.1 Obtener significado

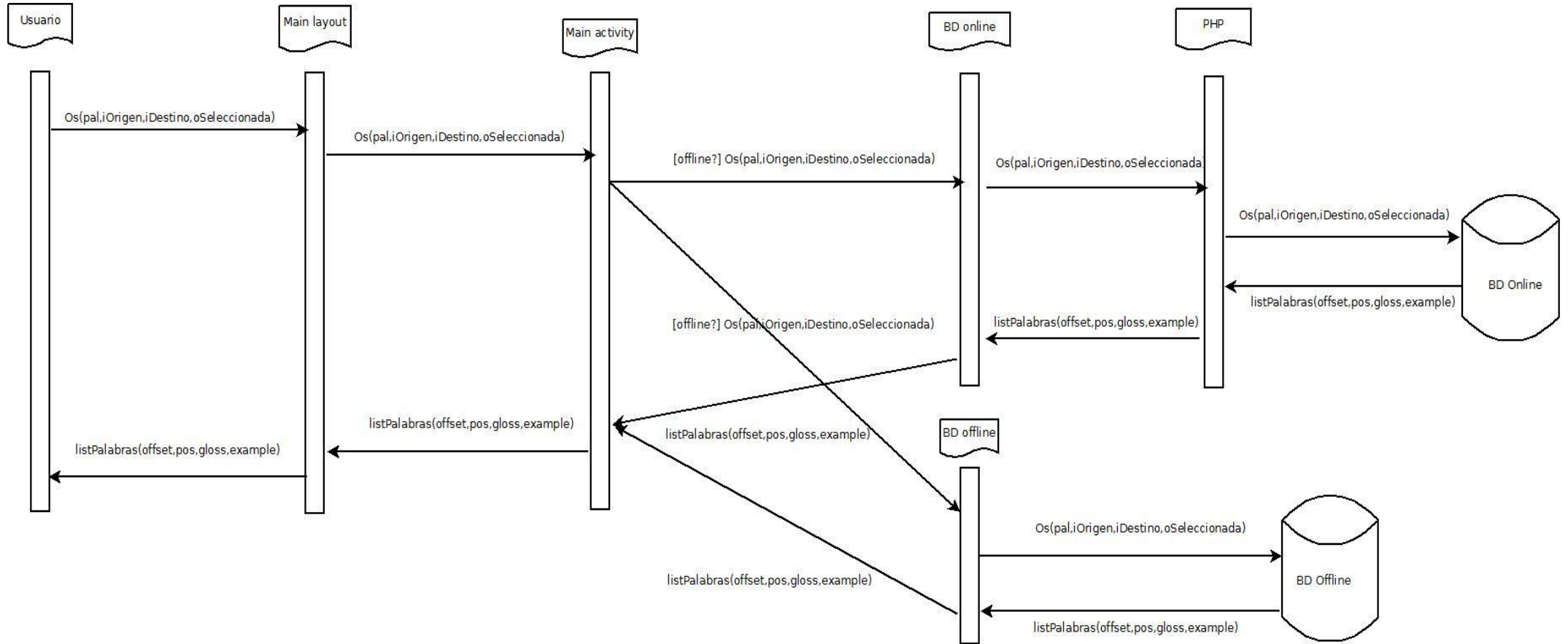


Figura 36: Diagrama de secuencia de obtener significados.

***Os(pal,iOrigen,iDestino,oSeleccionada)** = ObtenerSignificado(palabra,idiomaOrigen,idiomaDestino,operaciónSeleccionada)

***listPalabra** = ListaDePalabras(offset,pos,gloss,example)

7.1.2 Obtener traducción

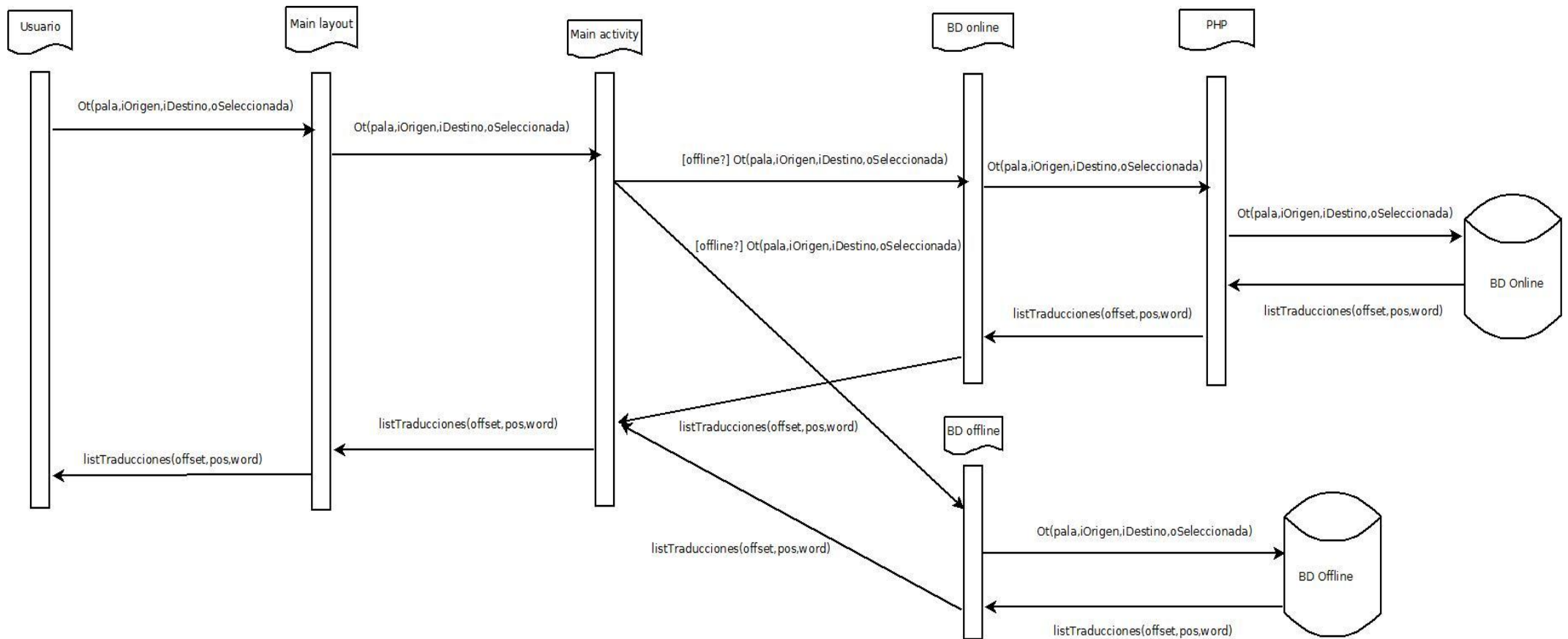


Figura 37: Diagrama de secuencia de obtener traducciones.

***Ot(pala,iOrigen,iDestino,oSeleccionada)** = ObtenerTraduccion(palabra,idiomaOrigen,idiomaDestino,operaciónSeleccionada)

***listTraducciones (offset,pos,word)** = listaDeTraducciones(offset,pos,word)

7.1.3 Obtener sinónimos, antónimos, hiperónimos o hipónimos

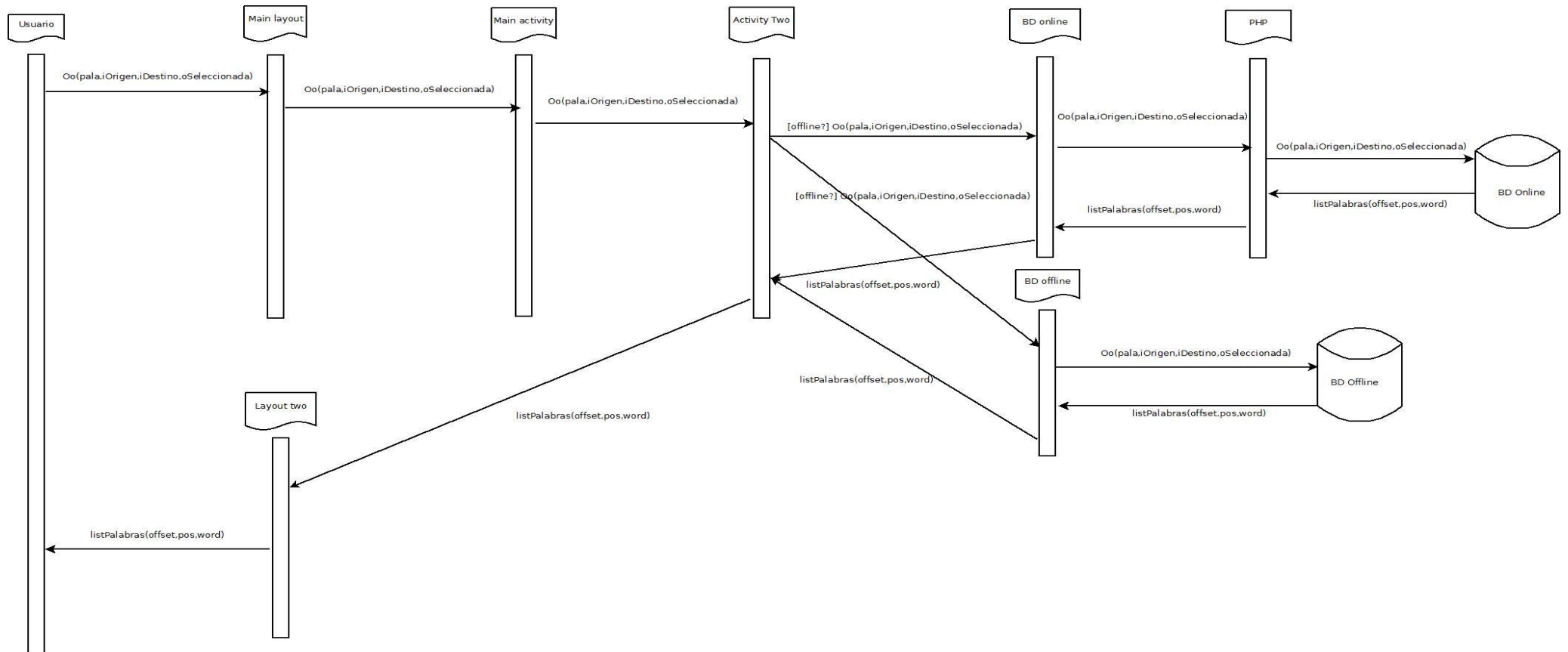


Figura 38: Diagrama de secuencia de obtener sinónimos, antónimos, hipónimos o hiperónimos.

***Oo(pala,iOrigen,iDestino,oSeleccionada)** = ObtenerOpción(palabra,idiomaOrigen,idiomaDestino,opciónSeleccionada)

***listaPalabras(offset,pos,word)**=listaDePalabras(offset,pos,Word)

7.2 Base de datos online

7.2.1 Comunicación con la base de datos externa

Para realizar las consultas a la base de datos externa, ha sido necesaria la creación de un módulo PHP, puesto que Android no permitía una conexión directa. Dicho PHP se comunica con la base de datos y devuelve la información en el formato JSON. Los datos obtenidos serán interpretados por la función correspondiente. Todos los datos necesarios para realizar las consultas son proporcionados por la clase encargada de la comunicación con la base de datos online.

1. Nos conectamos a la base de datos online.
2. Recibimos un parámetro indicando qué consulta se ejecutará (en función del caso de uso que estemos usando). Las opciones a ejecutar serán las siguientes:
 - a. Obtener el significado de la palabra que buscamos.
 - b. Obtener la traducción de la palabra que buscamos.
 - c. Obtener los sinónimos del sentido que buscamos.
 - d. Obtener los antónimos del sentido que buscamos.
 - e. Obtener los hipónimos del sentido que buscamos.
 - f. Obtener los hiperónimos del sentido que buscamos.
3. Los parámetros de la consulta también se recibirán por parámetro (los identificadores de las tablas de los idiomas a los que queremos acceder, la palabra a buscar y para los últimos casos, el tipo de relación que estamos buscando).
4. Transformamos la información a tipo de datos de JSON.
5. Devolvemos los datos.
6. Cerramos la conexión.

7.3 Base de datos offline

7.3.1 Obtener significado

Función que recibe el idioma de origen de la palabra, el idioma destino y la palabra para devolver las palabras, su tipo, ejemplos y definiciones en el idioma especificado (agrupadas por los sentidos).

1. Abrimos la base de datos offline.
2. Realizamos la consulta (a la que le pasamos el identificador de las tablas y la palabra por parámetro).
3. Vamos almacenando toda la información obtenida (significado, tipo de palabra, las palabras asociadas a ese significado y los ejemplos disponibles) agrupándola por su sentido.
4. Cerramos la base de datos.
5. Retornamos la información.

7.3.2 Obtener traducción

Función que recibe el idioma de origen de la palabra, el idioma destino y la palabra para devolver las traducciones en el idioma especificado (agrupadas por el sentido) junto a su respectivo tipo.

1. Abrimos la base de datos offline.
2. Comprobamos que el idioma de origen y destino sean diferentes (ya que sino no tendría sentido hacer ninguna traducción).
3. Realizamos la consulta (a la que le pasamos el identificador de las tablas y la palabra por parámetro).
4. Vamos almacenando toda la información obtenida (tipo de palabra, las palabras asociadas con el mismo código y los ejemplos disponibles) agrupándola por su sentido.
5. Cerramos la base de datos.
6. Retornamos la información.

7.3.3 Obtener sinónimos/antónimos/hiperónimos/hipónimos

Función que recibe el código del sentido (offset) de la palabra de la que vamos a obtener alguna de las opciones junto al idioma de origen y el de destino para devolver el conjunto de imágenes junto a su tipo (agrupadas por el sentido).

1. Abrimos la base de datos offline.
2. Comprobamos qué tipo de relación estamos buscando para elegir la consulta adecuada. Para saber que consulta ejecutar, nos fijamos en el código de la relación.
 - a. Si el código es 11 (identificador que nos indica el hiperónimos de la palabra):
 - i. Obtenemos los hiperónimos.
 - b. En el resto de los casos:
 - i. Obtenemos los antónimos, sinónimos o hipónimos de la palabra (esto sucede puesto que la forma de obtener los hiperónimos es diferentes del resto de consultas). Esto sucede debido a que la forma de obtener los hiperónimos es diferente del resto de relaciones.
3. Realizamos la consulta (a la que le pasamos las tablas, el tipo de relación y el código del que queremos obtener alguna de las opciones por parámetro).
4. Vamos almacenando toda la información obtenida (tipo de palabra y las palabras asociadas con el mismo código) agrupándola por su sentido.
5. Cerramos la base de datos.
6. Retornamos la información.

7.4 Comprobar conexión

Se encarga de comprobar si hay conexión a internet para realizar la consulta online.

1. Comprobamos si hay conexión a internet, ya sea por wifi o por datos.
2. Devolvemos true si hay alguna de las dos.

7.5 Crear base de datos

Se encarga de copiar la base de datos proporcionada en la ubicación que nos interesa.

1. Creamos un directorio en la tarjeta SD del dispositivo móvil.
2. Creamos un fichero vacío en dicha ubicación.
3. Copiamos la base de datos que tenemos en los recursos de la aplicación en el fichero que hemos creado mientras vamos actualizando el estado de la instalación en la barra de progresos.

8 Implementación

8.1 Android SDK

El SDK (Software Development Kit) de Android, incluye un conjunto de herramientas de desarrollo, entre las que encontramos, un depurador de código, biblioteca, un simulador de teléfono basado en QEMU, documentación, ejemplos de código y tutoriales. Las plataformas de desarrollo soportadas incluyen Linux (cualquier distribución moderna), Mac OS X 10.4.9 o posterior, y Windows XP o posterior. La plataforma integral de desarrollo (IDE, Integrated Development Environment), soportada oficialmente, es Eclipse, junto con el complemento ADT, Android Development Tools plugin, aunque también puede utilizarse un editor de texto para escribir ficheros Java y Xml y utilizar comandos en un terminal (se necesitan los paquetes JDK, Java Development Kit y Apache Ant) para crear y depurar aplicaciones. Además, pueden controlarse dispositivos Android que estén conectados (e.g. reiniciarlos, instalar aplicaciones en remoto).

Las Actualizaciones del SDK están coordinadas con el desarrollo general de Android. El SDK soporta también versiones antiguas de Android, por si los programadores necesitan instalar aplicaciones en dispositivos ya obsoletos o más antiguos. Las herramientas de desarrollo son componentes descargables, de modo que una vez instalada la última versión, pueden instalarse versiones anteriores y hacer pruebas de compatibilidad.

Una aplicación Android está compuesta por un conjunto de ficheros empaquetados en formato .apk y guardada en el directorio /data/app del sistema operativo Android (este directorio necesita permisos de superusuario, root, por razones de seguridad). Un paquete APK incluye ficheros .dex (ejecutables Dalvik, un código intermedio compilado), recursos, etc.

8.2 Interfaz

En Android las interfaces de usuario se realizan mediante el lenguaje XML, que permite crear la visualización de tu aplicación dentro del dispositivo.

Dichos archivos XML están formados por un árbol de elementos que especifican la manera en la que los elementos se acomodarán para definir la parte visual de un objeto View. Los atributos de cada elemento en el XML se llaman propiedades y describen cómo es que deben verse los elementos y cómo se deben comportar, tal y como puede apreciarse en la figura 39.

Los nodos, llamados ContainerViews o contenedores, son aquellos que pueden tener otros componentes como hijos y son derivados de la clase `android.view.ViewGroup`.

Un ViewGroup, normalmente, apenas realiza ningún dibujo y es responsable de situar a sus hijos en la pantalla, manteniéndolos en su sitio mientras que el dispositivo cambia de forma o de orientación.

En la carpeta "res", se almacenan todos los recursos del sistema. Dentro de estos recursos están nuestros archivos XML, que usaremos para crear las vistas de nuestra aplicación.

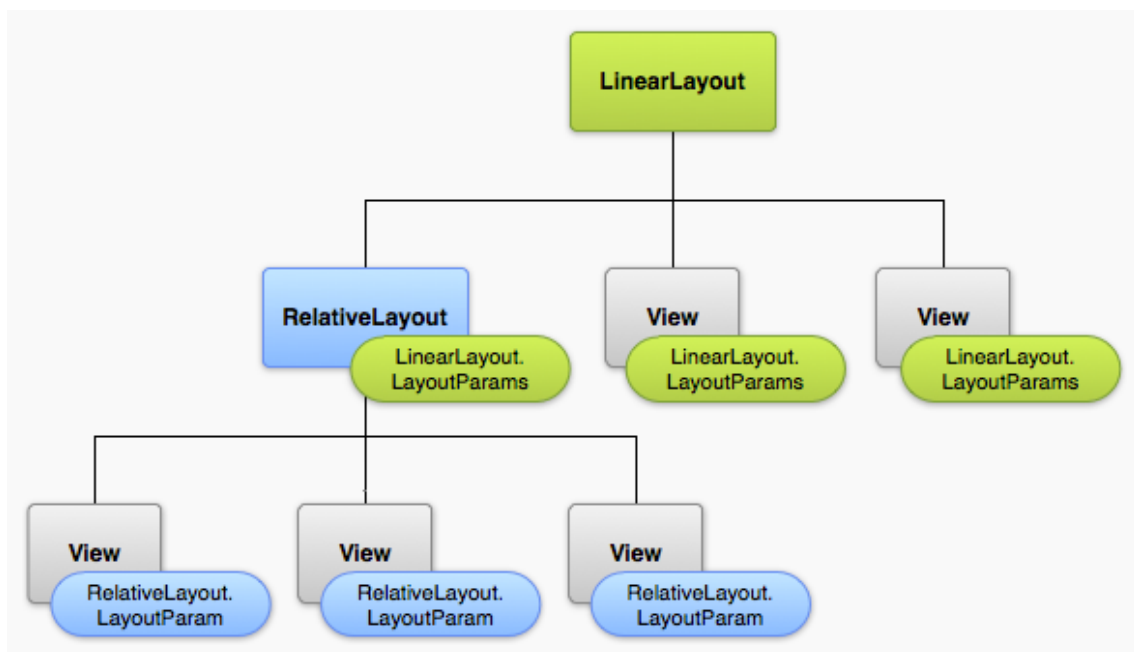


Figura 39: Herencia de objetos en la vista.

8.3 Activity

Una actividad o Activity es un componente de aplicación que proporciona una pantalla, con la que los usuarios pueden interactuar con el fin de hacer algo, como marcar un teléfono, tomar una foto, enviar un correo electrónico, o ver un mapa. Cada actividad genera una ventana en la que se muestra su interfaz de usuario. La ventana normalmente llena la pantalla, pero puede ser menor que la pantalla y flotan en la parte superior de otras ventanas.

Una aplicación, por lo general, se compone de múltiples actividades que están libremente delimitadas unas de las otras. Por lo general, una actividad en una aplicación se especifica como la actividad “principal”, que se presenta al usuario al iniciar la aplicación por primera vez. Cada actividad puede iniciar otra actividad, con el fin de realizar diferentes acciones. Cada vez que se inicia una nueva actividad, la actividad anterior se detiene, pero el sistema conserva la actividad en una pila (“backstack”). Cuando se inicia una nueva actividad, se inserta en la pila de nuevo y se le muestra al usuario (UI). El backstack tiene el funcionamiento básico de pila LIFO “last in, first out”, por lo que, cuando el usuario presiona el botón Atrás en la actividad actual, se extrae de la pila la actividad y se destruye, y la actividad anterior se visualiza.

Cuando una actividad se detiene porque se inicia otra actividad, este cambio en el estado se notificará a través de métodos de devolución de llamada (lifecyclecallback) de ciclo de vida de la actividad, tal y como podemos apreciar en la figura 40. Existen varios métodos de devolución de llamada que una actividad pueda recibir, debido a un cambio en su estado (esté creando el sistema, deteniéndolo, reanudándolo, o destruyéndolo) y cada devolución de llamada te ofrece la oportunidad de realizar un trabajo específico apropiado para el cambio de estado.

Para crear una actividad, debe crear una subclase de Activity (o una subclase existente de la misma). En la subclase, es necesario implementar métodos de devolución de llamada que el sistema llama cuando la actividad hace una transición entre los distintos estados de su ciclo de vida, como cuando se está creando la actividad, se detiene, se reanuda o se destruye.

A continuación explicaré, de una manera más concreta, cada uno de los estados por lo que la aplicación pasa:

- **Activa (Running):** La actividad está encima de la pila, lo que quiere decir que es visible y tiene el foco.
- **Visible (Paused):** La actividad es visible pero no tiene el foco. Se alcanza este estado cuando pasa a activa, otra actividad con alguna parte transparente o que no ocupa toda la pantalla. Cuando una actividad está tapada por completo, pasa a estar parada.
- **Parada (Stopped):** Cuando la actividad no es visible. El programador debe guardar el estado de la interfaz de usuario, preferencias, etc.
- **Destruída (Destroyed):** Cuando la actividad termina al invocarse el método finish(), o es matada por el sistema.

Cada vez que una actividad cambia de estado, se van a generar eventos que podrán ser capturados por ciertos métodos de la actividad. En la figura 40, se muestra un esquema que ilustra los métodos que capturan estos eventos.

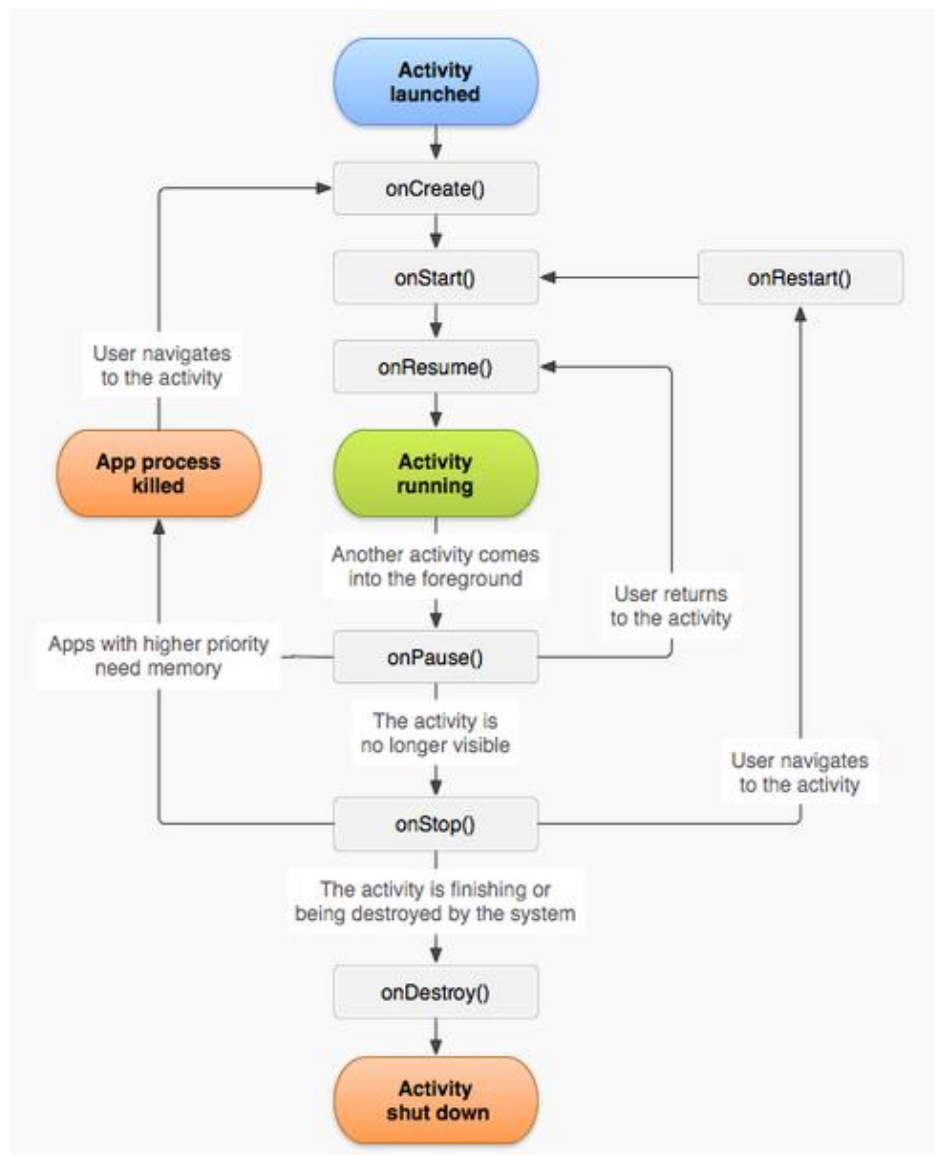


Figura 40: Ciclo de vida de una actividad.

- **onCreate(Bundle):** Se llama en la creación de la actividad. Se utiliza para realizar todo tipo de inicializaciones, como la creación de la interfaz de usuario o la inicialización de estructuras de datos. Puede recibir información de estado de la actividad (en una instancia de la clase Bundle), por si se reanuda desde una actividad que ha sido destruida y vuelta a crear.
- **onStart():** Nos indica que la actividad está a punto de ser mostrada al usuario.
- **onResume():** Se llama cuando la actividad va a comenzar a interactuar con el usuario. Es un buen lugar para lanzar las animaciones y la música.
- **onPause():** Indica que la actividad está a punto de ser lanzada a segundo plano, normalmente porque otra actividad es lanzada. Es el lugar adecuado para detener animaciones, música o almacenar los datos que estaban en edición.
- **onStop():** La actividad ya no va a ser visible para el usuario. Si hay muy poca memoria, es posible que la actividad se destruya sin llamar a este método.
- **onRestart():** Indica que la actividad va a volver a ser representada después de haber pasado por onStop().
- **onDestroy():** Se llama antes de que la actividad sea totalmente destruida. Por ejemplo, cuando el usuario pulsa el botón <volver> o cuando se llama al método finish().

8.4 Estructura de un proyecto Android

A continuación, explicaremos la estructura de un proyecto en Android y la funcionalidad de cada una de las carpetas que podemos apreciar en la figura 41.

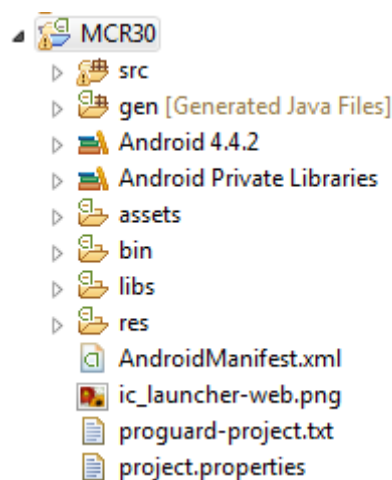


Figura 41: Estructura de un proyecto Android.

8.4.1 Carpeta src

Ésta es por defecto la carpeta donde se depositará el código fuente Java, y como en los proyectos para escritorio puede llamarse de otro modo si nos interesa o podemos disponer de distintas carpetas fuente, a veces puede ser un modo de enlazar con librerías.

Todo el código que pongamos en las carpetas fuente será compilado cuando se requiera. Y también, como en los proyectos tradicionales de java, el código se organizará en carpetas que resultarán en paquetes. Por norma general, la estructura de nuestro proyecto partirá de esta raíz, tal como se ve en la figura 42.

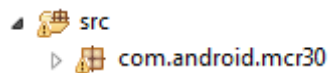


Figura 42: Estructura carpeta src.

8.4.2 Carpeta res

Es una carpeta más compleja, ya que se subdivide en múltiples subdirectorios (tal y como podemos apreciar en la figura 43), pero para resumir, diremos que va a contener lo que en Android llamaremos recursos. En estas carpetas podemos encontrar imágenes, textos y layouts; y pudiendo administrar como se van a gestionar los recursos para dispositivos con características y configuraciones distintas (densidad de píxel, localización, etc.).

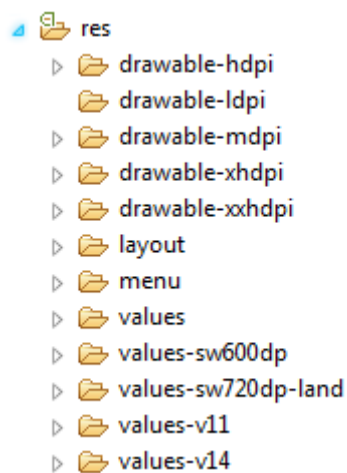


Figura 43: Estructura carpeta res.

Hay que tener en cuenta que, los archivos que vayamos a crear como recursos, deben seguir una nomenclatura determinada, ya que no se va a permitir que el nombre del archivo empiece por un valor numérico, por poner un ejemplo, o que contenga mayúsculas.

8.4.3 Carpeta bin

Esta carpeta, como la carpeta gen, se generan automáticamente, y las utiliza el compilador para preparar los archivos para el empaquetado final en forma de APK. Es un paso intermedio. Tanto esta carpeta, como la anterior, la podemos borrar si es necesario, debido a que se generaran de nuevo al compilar, y cabe considerar excluir estas carpetas si utilizáis un sistema de repositorio, puesto que al importar el proyecto de nuevo, sucederá lo mismo, generándose automáticamente.

8.4.4 Carpeta assets

En esta carpeta depositaremos todos los archivos que vayamos a querer que acompañen a la aplicación, pueden estar organizados en carpetas y serán empaquetados en el apk final. Podemos considerarlos como recursos, al igual que el contenido de la carpeta res, pero no estarán indexados ni compilados, de modo que el acceso a ellos es menos eficiente y algo más laborioso. Por otro lado, no tendremos que aplicar las reglas de los recursos en lo que respecta a nomenclaturas y organización.

8.4.5 Carpeta libs

Está pensada para contener las librerías que vamos a enlazar a nuestro proyecto, por lo general, se van a depositar aquí los archivos JAR empaquetados de librerías portables.

8.4.6 Archivo AndroidManifest.xml

Podemos decir que este es el archivo principal y que todas las aplicaciones o librerías deben contener la raíz del proyecto. En éste, se definen las características del proyecto

como el nombre, paquete o los permisos que va a requerir la aplicación. También es, en este archivo, donde se describen los componentes de la aplicación.

Como su nombre indica, es un archivo XMO, pero al tener instaladas el plugin de Android, podemos visualizarlo y editarlo a partir de un editor visual, además de sobre un editor de archivos XML. Si lo abrimos haciendo doble click sobre él, se mostrará en forma XML; pero podremos navegar por distintas pestañas que nos mostrarán el editor disgregado en secciones, tal y como se puede ver en la figura 44.



```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.mcroff"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="9"
        android:targetSdkVersion="18" />

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.android.mcr30.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity
            android:name="com.android.mcr30.ActivityTwo"
            android:label="@string/title_activity_activity_two" >
        </activity>
    </application>
```

Manifest Application Permissions Instrumentation AndroidManifest.xml

Figura 44: Ejemplo de un AndroidManifest.

8.5 Estructura MCRA 3.0

En este apartado explicaremos los diferentes archivos del proyecto MCRA 3.0, encargado de realizar las operaciones solicitadas con la palabra introducida, así como los elementos más significativos que podemos encontrar en cada uno de ellos.

8.5.1 ActivityMain.xml

Es el xml que contiene la interfaz que se cargará nada más lanzar la aplicación. Usa el *RelativeLayout* como *layout* principal para contener el resto de elementos. El *RelativeLayout* es un *layout* que nos permite posicionar los elementos de forma relativa a otros elementos (por ejemplo, a su izquierda, debajo de él) o a otro *layout* (por ejemplo, alineado a la parte de abajo o a la derecha). Dicha interfaz contiene los siguientes elementos: un menú desplegable para elegir el idioma de origen, el idioma de destino y la funcionalidad a utilizar, un cuadro de texto para introducir la palabra que nos interesa buscar, un botón para confirmar la palabra a buscar (para que comience el proceso) y un *LinearLayout* para mostrar los resultados que serán creados dinámicamente. El *LinearLayout* es un *layout* que nos permite alinear los elementos en una sola dirección, vertical u horizontal.

8.5.2 MainActivity.java

Es la clase principal de la aplicación y aquella que es llamada cuando la aplicación comienza a funcionar y usa el xml ActivityMain. Se encarga de: leer la palabra introducida para enviarla a la clase encargada de realizar la conexión de la base de datos, mostrar al usuario los resultados mediante la creación dinámica de los *TextView* (cuadros de texto que muestra la información), asignar los estilos a texto generado y al *layout* en el que se crean, comprobar si el dispositivo móvil dispone de conexión a internet y de crear el menú que aparece cuando pulsamos en uno de los resultados. En caso de que el usuario selecciona alguno de las opciones del menú, también se encarga de lanzar la segunda actividad.

8.5.3 ActivityTwo.xml

Es el xml que se cargará cuando lancemos la segunda actividad. Contiene un *RelativeLayout*, y dentro de éste, un *TextView* en el que mostraremos la opción que hemos seleccionado y un *LinearLayout*, en el que crearan dinámicamente *TextView* por cada uno de los resultados de la opción seleccionada.

8.5.4 ActivityTwo.java

Es la clase que se encarga de realizar la gestión de las operaciones necesarias para la realización de la opción seleccionada en la actividad anterior. Esta actividad utiliza el *ActivityTwo.xml* como interfaz. Igual que la actividad anterior, se encargará de gestionar las acciones necesarias para realizar las consultas con las que obtener los datos asociados a una de las opciones elegidas, mostrarlas dentro del *LinearLayout*, asignarles un estilo y comprobar si aún disponemos de conexión a internet o no.

8.5.5 Palabra.java

Clase auxiliar para almacenar de una manera sencilla y ágil los datos que recibimos de las consultas realizadas a la base de datos.

8.5.6 Custom_bg_2.xml

XML que contiene el estilo que utilizamos para mostrar los resultados de la aplicación. Dicho estilo agrega un cuadrado negro en el *TextView* que nosotros elijamos.

8.5.7 DataBaseOnline.java

Contiene las operaciones para conectarse a la base de datos online y de transformar la información que nos retorna la base de datos a una estructura más manejable, cómoda y agrupada por el mismo sentido. Esta clase se comunica con el php (Traducciones) que se encarga de realizar la consulta a la base de datos alojada en el servidor y de retornarnos los resultados en el formato JSON.

8.5.8 DataBaseOffline.java

Contiene las operaciones para realizar las diferentes consultas a la base de datos interna de la aplicación, e igual que la clase online, también transforma la información que nos retorna la base de datos offline a una estructura más manejable, cómoda y agrupada por el mismo sentido.

8.5.9 String.xml

Contiene todas las constantes de cadenas de caracteres que se necesitan en un programa, entre los que podemos encontrar los diferentes idiomas que se muestran en el menú desplegable, las opciones, etc. Además, podemos crear diferentes carpetas para todos los idiomas a los que queramos dar soporte, con el fichero string.xml incluido, y Android se cargará de cargar el archivo asociado a dicho idioma, facilitando así la opción de multilinguaje.

8.5.10 Pantalla

En este apartado comprobaremos el estado final de las interfaces durante la ejecución de la aplicación.

8.5.10.1 Interfaz Main activity

En la figura 45, podemos apreciar el primer estado de la interfaz, antes de haber introducida ninguna palabra.



Figura 45: Pantalla principal.

En la figura 46, podemos apreciar el estado de la interfaz tras haber seleccionado un idioma de origen, un idioma de destino y haber introducido una palabra a buscar.

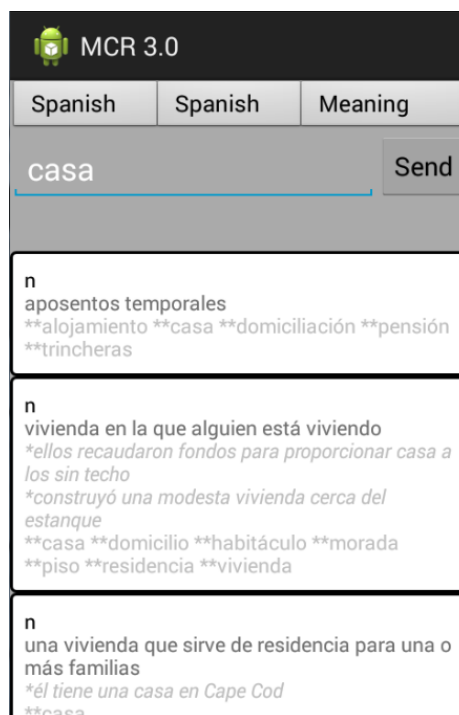


Figura 46: Pantalla principal tras realizar la búsqueda.

8.5.10.2 Interfaz ActivityTwo

En la figura 47, podemos apreciar el estado en el que vemos la interfaz tras haber seleccionado alguna de las opciones.

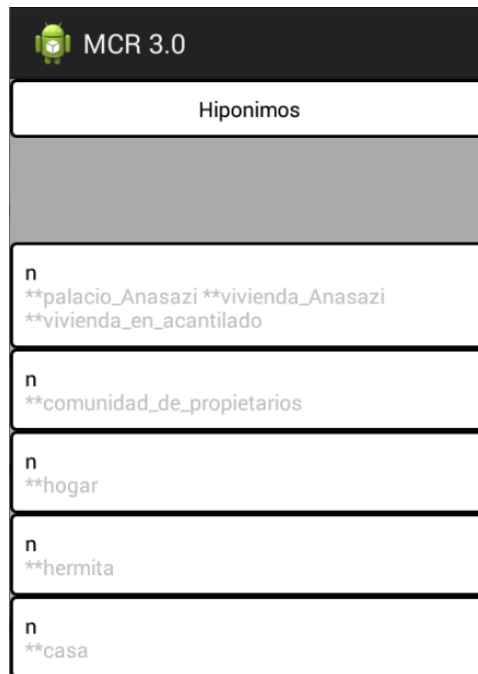


Figura 47: Obtención de hipónimos.

8.6 Estructura MCRBD 3.0

En este apartado explicaremos los diferentes archivos del proyecto MCRBD 3.0, encargado de instalar la base de datos en la tarjeta SD del dispositivo móvil, así como los elementos más significativos que podemos encontrar en cada uno de ellos.

8.6.1 Interfaz Main Activity.xml

Es el xml que contiene la interfaz que se cargará nada más lanzar la aplicación. Contiene un *RelativeLayout*, un cuadro de texto en el que mostrar la información, un botón que comenzará la instalación y una barra de progreso.

8.6.2 Main Activity.java

Es la clase principal de la aplicación que contiene las operaciones necesarias para copiar la base de datos en la tarjeta SD. Nada más ejecutar la aplicación, nos mostrará un mensaje indicándonos que podemos empezar la instalación. Tras pulsar el botón, la base de datos se empezará a copiar y se mostrará el estado de la instalación en la barra de progresos.

8.6.3 String.xml

Al igual que en el anterior archivo xml, se encarga de almacenar todas las constantes de cadenas de caracteres asociadas al proyecto y de ayudarnos a la hora de realizar una aplicación multilinguaje.

8.6.4 Pantalla

En este apartado podremos ver el estado final de la interfaz durante la ejecución de la aplicación.

8.6.4.1 Interfaz Main activity

En la figura 48, podemos ver el primer estado que observamos nada más ejecutar la aplicación.



Figura 48: Estado inicial.

En la figura 49, podemos apreciar el estado de la pantalla tras haber terminado la instalación



Figura 49: Base de datos instalada.

9 Pruebas

A lo largo del proyecto, hemos ido haciendo diferentes pruebas que nos han ayudado a depurar los errores que han ido apareciendo. Las pruebas son un elemento fundamental para garantizar la calidad del software, y representa una revisión final de las especificaciones. Las pruebas realizadas para asegurarnos del buen funcionamiento de la aplicación han sido las siguientes:

- Unitarias.
- Integración.
- Sistema.

Cada una de las pruebas referentes a las consultas de las palabras (ya sea significado, traducción, sinónimos, etc.) se realizó con un conjunto de 15 palabras, entre nombres, verbos, adjetivos y adverbios, comprobando todos sus resultados con la versión online MCR 3.0.

9.1 Pruebas Unitarias

9.1.1 Comprobar conexión

Se han realizado diferentes pruebas para analizar si el dispositivo comprobaba correctamente si estaba conectado a internet, tanto por wifi o por datos. Las pruebas realizadas fueron las siguientes:

- Poniendo y quitando manualmente el acceso a internet.
 - Con el internet conectado:
 - La respuesta esperada ha sido: true
 - La respuesta obtenida ha sido: true
 - Con el internet apagado:
 - La respuesta esperada ha sido: false
 - La respuesta obtenida ha sido: false
- Estando la opción del wifi conectada, pero teniendo sistema wifi apagado:
 - La respuesta esperada ha sido: false
 - La respuesta obtenida ha sido: false

Como podemos, apreciar la funcionalidad funciona de forma correcta.

9.1.2 Creación del menú de opciones y cambio de idiomas

Estas pruebas se han realizado para comprobar si el menú se muestra de forma correcta y si la aplicación cambia correctamente de idioma. Para ello, vamos cambiando el idioma del sistema operativo Android y vemos si la aplicación también lo cambia.

- Castellano:
 - El menú se muestra correctamente y se encuentra en el mismo idioma que el del sistema operativo.
- Inglés:
 - El menú se muestra correctamente y se encuentra en el mismo idioma que el del sistema operativo.
- Euskera:
 - El menú se muestra correctamente y se encuentra en el mismo idioma que el del sistema operativo.
- Catalán:
 - El menú se muestra correctamente y se encuentra en el mismo idioma que el del sistema operativo.
- Gallego:
 - El menú se muestra correctamente y se encuentra en el mismo idioma que el del sistema operativo.

Podemos apreciar que la aplicación cambia de idioma en función del idioma del sistema operativo y que el menú se crea correctamente en todos los casos.

9.1.3 Obtener traducciones offline

En esta prueba, se comprueba que se puede establecer conexión con la base de datos offline, que se ejecuta adecuadamente la consulta y nos devuelve una estructura de datos agrupados por los sentidos (con el tipo de palabras y las palabras). Las pruebas se realizaron con todos los idiomas en los siguientes casos:

- Palabras existentes:
 - Funcionamiento incorrecto de todos los idiomas.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:

- Funcionamiento correcto.

Nos dimos cuenta de que el primer elemento no estaba bien agrupado por su sentido en el caso de las palabras existentes (en el caso de las palabras no existentes y los caracteres aleatorios funciona bien, puesto que al estar la estructura vacía, no se agrupaba). Esto sucedía, debido a que la función que se encarga de realizarlo no funcionaba correctamente. Tras corregir eso, volvemos a realizar las pruebas a todos los idiomas.

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Tras solucionar este error, comprobamos que nos devuelve correctamente una estructura de datos agrupada por su sentido. Este cambio beneficia al resto de pruebas, debido a que nos ahorramos dicho fallo en todas las funciones que usarían dicha operación.

9.1.4 Obtener significados offline

En esta prueba, se comprueba que se puede establecer conexión con la base de datos offline, que se ejecuta adecuadamente la consulta y nos devuelve una estructura de datos agrupados por su sentido (con el tipo de palabras, significados, ejemplos y palabras). Las pruebas se realizaron con los siguientes casos:

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Tras comprobar los resultados, podemos asegurar que la operación funciona correctamente.

9.1.5 Obtener sinónimos/antónimos/hipónimos/hiperónimos offline

En esta prueba, se comprueba que se puede establecer conexión con la base de datos offline, que se ejecuta adecuadamente la consulta y nos devuelve una estructura de datos agrupados por su sentido (con el tipo de palabras, significados, ejemplos y palabras). Las pruebas se realizaron con los siguientes casos:

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Tras comprobar los resultados, podemos asegurar que la función encargada de obtener los sinónimos, antónimos, hipónimos e hiperónimos funciona correctamente.

9.1.6 Obtener traducciones online

En esta prueba, se comprueba que se puede establecer conexión con la base de datos online, que se ejecuta adecuadamente la consulta y nos devuelve una estructura de datos agrupados por el sentido (con el tipo de palabras y las palabras). Las pruebas se realizaron con los siguientes casos:

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Tras analizar los resultados, podemos decir que la operación funciona de forma correcta.

9.1.7 Obtener significados online

En esta prueba, se comprueba que se puede establecer conexión con la base de datos online, que se ejecuta adecuadamente la consulta y nos devuelve una estructura de datos agrupados por el sentido (con el tipo de palabras, significados, ejemplos y palabras). Las pruebas se realizaron con los siguientes casos:

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Tras comprobar los resultados podemos asegurar que la operación funciona correctamente.

9.1.8 Obtener sinónimos/antónimos/hipónimos/hiperónimos online

En esta prueba, se comprueba que se puede establecer conexión con la base de datos online, que se ejecuta adecuadamente la consulta y nos devuelve una estructura de datos agrupados por el sentido (con el tipo de palabras, significados, ejemplos y palabras). Las pruebas se realizaron con los siguientes casos:

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Tras comprobar los resultados, podemos asegurar que la función encargada de obtener los sinónimos, antónimos, hipónimos e hiperónimos funciona correctamente.

9.2 Pruebas de integración y sistema

Estas pruebas se encargan de comprobar si todas las funcionalidades, anteriormente descritas, funcionan en conjunto y si los datos que éstas devuelven son los que esperábamos. Se ha decidido realizar estas pruebas al mismo tiempo, debido a que al evaluar el funcionamiento, íbamos a obtener una respuesta que podía ser analizada para comprobar si lo que nos ha devuelto es lo que esperábamos o no, puesto que puede darse el caso de que funcionen de forma conjunta pero el resultado puede no ser el esperado. Todos los resultados se compararán con la aplicación MCR 3.0 para comprobar si son correctos o no.

9.2.1 Consultar significado de una palabra offline

En esta prueba, comprobaremos el funcionamiento de la aplicación con el caso de uso de obtener el significado cuando utilizamos la base de datos interna. Para ello, utilizaremos diferentes palabras y compararemos el resultado obtenido por nuestra aplicación con el que obtenemos del MCR 3.0 online.

- Palabras existentes:
 - Funcionamiento incorrecto en todos los idiomas.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Nos dimos cuenta que el diccionario no funcionaba correctamente, bien porque salían con extraños caracteres o bien porque faltaban ejemplos (con los caracteres aleatorios o palabras no existentes, no se daba el caso, puesto que no retornaba una estructura vacía). Tras un estudio de los motivos por los que no funcionaba correctamente, llegamos a la conclusión de que se debía a que la herramienta de SQLite no había interpretado correctamente los acentos ni las “ñ”, por lo que se había construido mal la base de datos. Se tuvo que volver a construir la base de datos con otro gestor de bases de datos que interpretase correctamente los acentos y las “ñ”. Para comprobar que todo funcionase bien, repetiremos las pruebas anteriores.

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.

- Caracteres aleatorios:
 - Funcionamiento correcto.

Podemos comprobar que los módulos funcionan correctamente de forma conjunta y que el resultado obtenido era el que esperábamos, por lo que podemos asegurar que el módulo funciona de forma correcta.

9.2.2 Consultar traducción de una palabra offline

En esta prueba, comprobaremos el funcionamiento del caso de uso obtener traducciones de una palabra, usando la base de datos interna.

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Podemos comprobar que los módulos funcionan correctamente de forma conjunta y que el resultado obtenido era el que esperábamos, por lo que podemos asegurar que el módulo funciona de forma correcta.

9.2.3 Consultar sinónimos/antónimos/hipónimos/hiperónimos offline

En esta prueba, comprobaremos el funcionamiento del caso de uso de obtener sinónimos, antónimos, hipónimos e hiperónimos.

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Podemos comprobar que los módulos funcionan correctamente de forma conjunta y que el resultado obtenido era el que esperábamos, por lo que podemos asegurar que este módulo funciona de forma correcta.

9.2.4 Consultar significado de una palabra online

En esta prueba, comprobaremos el funcionamiento de la aplicación con el caso de uso de obtener el significado cuando usamos la base de datos online. Para ello, utilizaremos diferentes palabras y compararemos el resultado obtenido por nuestra aplicación con el que obtenemos del MCR 3.0 online.

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Podemos comprobar que los módulos funcionan correctamente de forma conjunta y que el resultado obtenido era el que esperábamos, por lo que podemos asegurar que el módulo funciona de forma correcta.

9.2.5 Consultar traducción de una palabra online

En esta prueba, comprobaremos el funcionamiento del caso de uso obtener traducciones de una palabra usando la base de datos online.

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Podemos comprobar que los módulos funcionan correctamente de forma conjunta y que el resultado obtenido era el que esperábamos, por lo que podemos asegurar que el módulo funciona de forma correcta.

9.2.6 Consultar sinónimos/antónimos/hipónimos/hiperónimos online

En esta prueba, comprobaremos el funcionamiento del caso de uso de obtener sinónimos, antónimos, hipónimos e hiperónimos con una base de datos online.

- Palabras existentes:
 - Funcionamiento correcto.
- Palabras no existentes:
 - Funcionamiento correcto.
- Caracteres aleatorios:
 - Funcionamiento correcto.

Podemos comprobar que los módulos funcionan correctamente de forma conjunta y que el resultado obtenido era el que esperábamos, por lo que podemos asegurar que el módulo funciona de forma correcta.

10 Implantación

Lo primero, sería necesario alojar la base de datos online en algún servidor, puesto que sin ella, nuestra aplicación perdería su funcionalidad online. Será necesario un servidor que sea capaz de manejar un número considerable de consultas y que esté operativo las 24 horas del día. Tras eso, se tendría que alojar la aplicación en algún gestor de aplicaciones Android para que la gente pudiese empezar a utilizarlo. Entre todos los gestores disponibles, la opción más recomendable sería utilizar Google Play³⁰, debido a que es la plataforma más popular y donde podría tener más difusión. Para poder alojar la aplicación, será necesario seguir los siguientes pasos:

- La aplicación tendrá que tener un tamaño máximo de 50MB y estar en formato APK³¹.
- Para almacenar la base de datos en Google Play (para poder descargarla e instalarla en la tarjeta SD mediante la aplicación MCRBD 3.0), será necesario alojarla como un archivo de expansión. Dicha opción nos permite alojar hasta dos archivos con tamaño máximo de 2GB.
- Captura de pantalla de la aplicación, con un mínimo de dos capturas y un máximo de ocho.
- Idioma en el que se encuentra nuestra aplicación.
- El nombre con el que queremos que aparezca reflejada nuestra aplicación en Google Play.
- Descripción de la aplicación que se verá en Google Play, con un máximo de 4.000 caracteres.
- Novedades de la aplicación, por ejemplo, cuando la actualicemos o hagamos algún cambio que queramos mostrar a los usuarios.
- Tipo de aplicación: Aplicación o Juego.
- Elegir la categoría³² de nuestra aplicación.

³⁰ <https://play.google.com/store>

³¹ *Application Package File*, es un paquete para el sistema operativo Android. Este formato es una variante del formato JAR de Java y se usa para distribuir e instalar componentes empaquetados para la plataforma Android, ya sean smartphones o tablets.

³² <https://support.google.com/googleplay/android-developer/answer/113475>

A la hora de instalarlo, no debería haber ningún problema, debido a que la aplicación ha sido testeada con una gran variedad de versiones y ha funcionado correctamente en todas, tal y como podemos apreciar en la tabla 10.

Las versiones probadas han sido las siguientes:

- 2.3.3
- 3.2
- 4.1.2
- 4.3
- 4.4.2

Se decidió probar en esas versiones, debido a que son las más utilizadas, por lo que se abarcaría una gran cantidad de usuario.

	MCR	BD
2.3.3	✓	✓
3.2	✓	✓
4.1.2	✓	✓
4.3	✓	✓
4.4.2	✓	✓

Tabla 10: Compatibilidades en diferentes versiones.

11 Gestión del proyecto

Nos disponemos a resumir los aspectos más relevantes de la gestión del proyecto. En particular, veremos la diferencia entre las horas planificadas y las horas reales que hemos invertido a la hora de hacer el proyecto. Esto sucede porque la planificación inicial no suele coincidir con la real, puesto que nunca se puede saber a ciencia cierta el tiempo que se tendrá que dedicar a cada parte. Hemos optado por representar el tiempo en diferentes tablas y gráficas, para poder apreciar de forma visual la diferencia entre ambas.

11.1 Comparativa entre esfuerzo planificado y real

11.1.1 Procesos tácticos

Actividad	Planificado	Tiempo real
Planificación	10	15
Gestión	5	4
Instalación	15	15
Reuniones	10	10
Total	40	44
Desviación		+10 %

Tabla 11: Procesos tácticos.

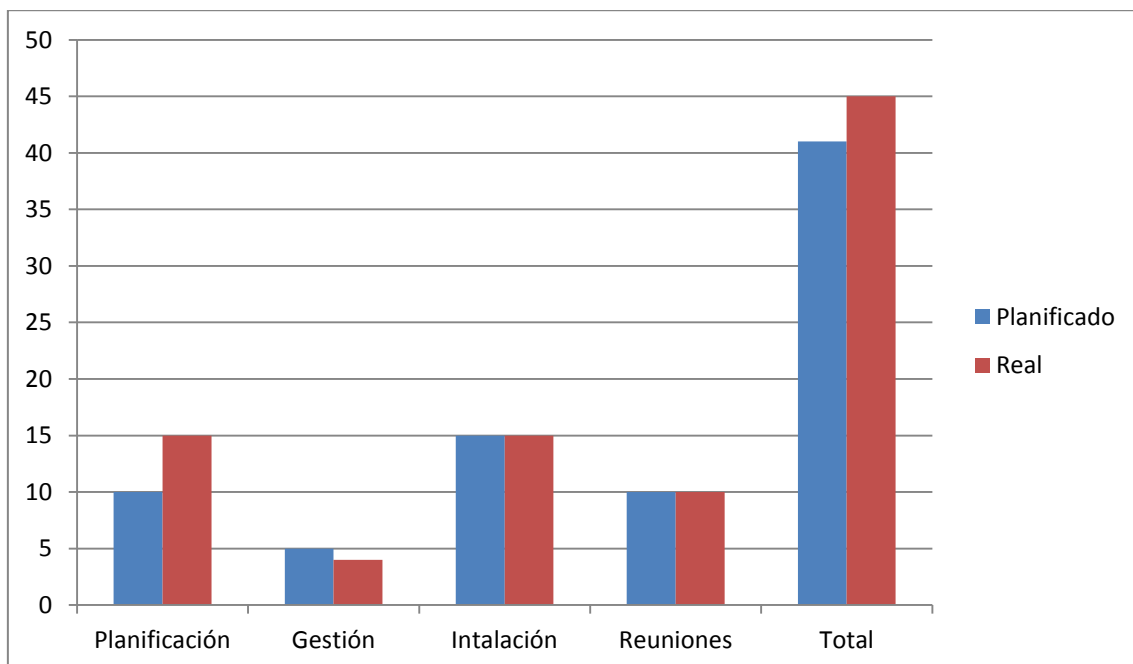


Figura 50: Gráfico de las horas planificadas frente a la reales en los procesos tácticos.

Podemos observar, en la tabla 11 y figura 50, que, en estos procesos, ha habido una desviación muy pequeña (+10%), puesto que se tenían las cosas más o menos claras desde el principio.

11.1.2 Procesos formativos

Actividad	Tiempo estimado	Tiempo real
Android	60	40
Curso Android	0	50
BD MCR 3.0	3	3
SQLite	2	5
Google Play	1	1
Algoritmos de compresión	6	3
App Wordnet	1	1
Alojamiento en la nube	3	1
Memoria	60	70
Presentación	6	5
Total	142	179
Desviación		+26 %

Tabla 12: Procesos formativos.

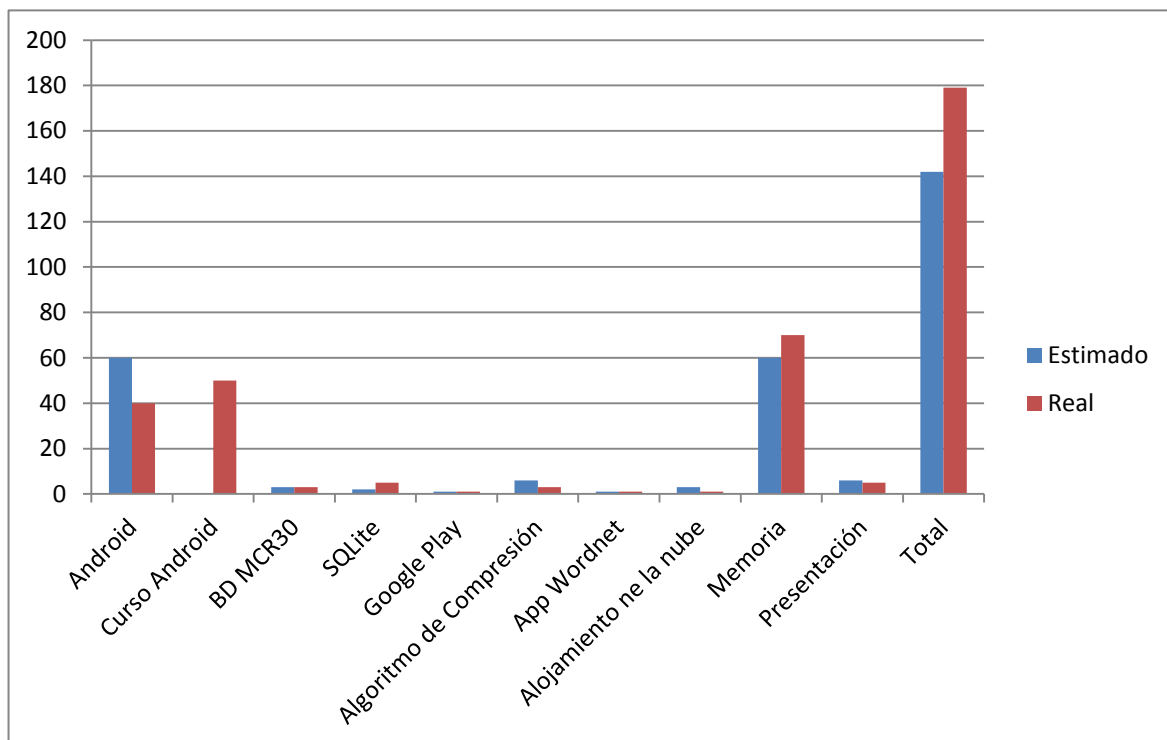


Figura 51: Gráfico de las horas planificadas frente a la reales en los procesos formativos.

En la tabla 12 y figura 51, podemos apreciar que la mayor desviación en estos procesos ha surgido del curso de Android realizado. En un principio, no estaba planificado realizarlo, pero tras conocer la página web de Coursera (la cual ofrece una gran variedad de cursos), se decidió cursar uno de los que ofrecían, para profundizar y entender mejor la programación en Android.

11.1.3 Procesos operativos

- 1ª Iteración

Actividad	Tiempo estimado	Tiempo real
Captura de requisitos	7	8
Análisis	5	7
Diseño	7	3
Implementación	60	80
Pruebas	3	4
Total	82	102
Desviación		+24,39 %

Tabla 13: Primera iteración.

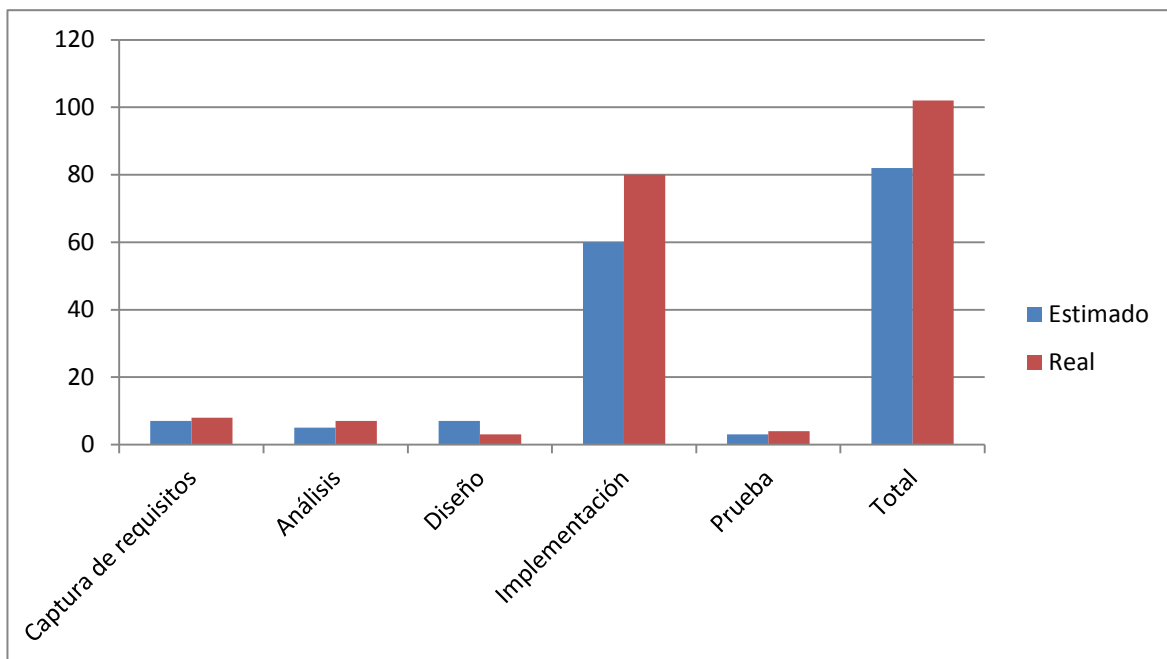


Figura 52: Gráfico de las horas planificadas frente a las reales en la primera iteración.

En la tabla 13 y figura 52, podemos observar que, en esta iteración, nos hemos desviado bastante del tiempo planificado frente al real (+24,39%). Esto ha sucedido por los siguientes motivos:

1. Se intentaron realizar técnicas de compresión que, finalmente, fueron descartada.
2. Problemas surgidos a la hora de realizar la conexión de la base de datos online.
3. El tiempo dedicado a transformar las tablas de la base de datos en MYSQL a SQLite, debido a que las operaciones son ligeramente diferentes entre ambos sistemas y no podía realizarse de forma automática.

- 2º Iteración

Actividad	Tiempo planificado	Tiempo real
Captura de requisitos	6	8
Análisis	4	6
Diseño	6	5
Implementación	70	90
Pruebas	5	8
Total	91	117
Desviación	+28,57 %	

Tabla 14: Segunda iteración.

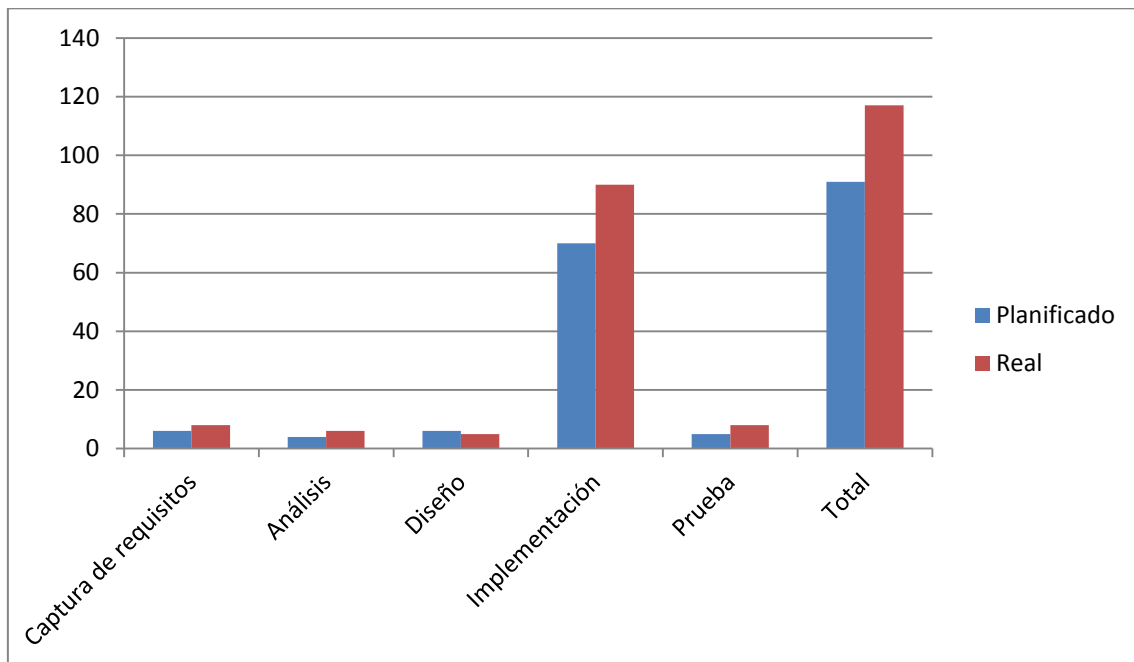


Figura 53: Gráfico de las horas planificadas frente a la reales en la segunda iteración.

En la tabla 14 y figura 53, podemos observar que, en esta iteración, nos hemos desviado bastante del tiempo planificado frente al real (+28,57%). Esto ha sucedido por los siguientes motivos:

1. Se tuvo que volver a construir la base de datos debido a que los acentos y las “ñ” no se habían codificado de forma correcta.
2. Al utilizar todas las tablas disponibles en la base de datos offline, vimos que ocupaban demasiado espacio, por lo que se decidió crear nuevas tablas y organizar la información de diferente manera.
3. Tras una reunión con el director, nos dimos cuenta de que las consultas no se hacían de forma correcta, por lo que se tuvo que volver a escribirlas.

- 3ª Iteración

Actividad	Tiempo planificado	Tiempo real
Captura de requisitos	8	10
Análisis	6	4
Diseño	6	8
Implementación	50	70
Pruebas	4	15
Total	74	107
Desviación	+44,59 %	

Tabla 15: Tercera iteración.

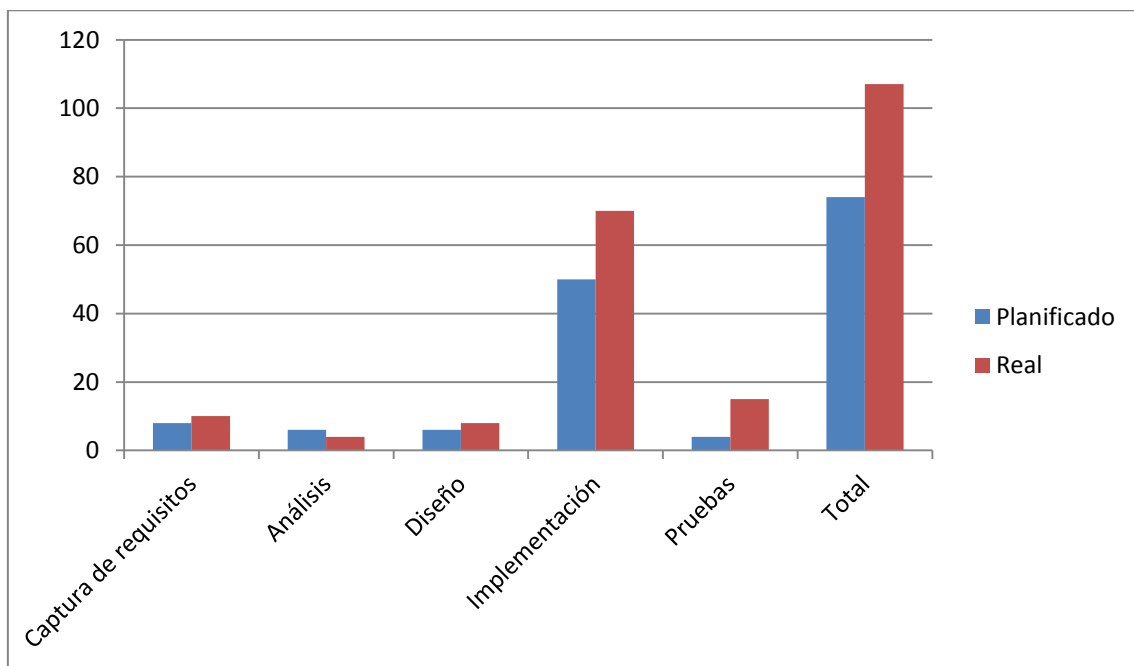


Figura 54: Gráfico de las horas planificadas frente a la reales en la tercera iteración.

En la tabla 15 y figura 54, se puede apreciar que, en esta iteración, se necesitó menos tiempo del planificado en la interfaz, gracias a que se consiguió hacer lo que se quería más rápido de lo esperado. Se tardó más tiempo en realizar todas las pruebas en la versión final de la aplicación, así como mejoras de la aplicación, por ejemplo: multi-idioma, instalación de la base de datos mediante una aplicación independiente, etc, siendo el desvío total de +44,59%.

11.2 Comparativa de las horas totales planificadas frente a las reales

Procesos	Tiempo planificado	Tiempo real
Tácticos	40	44
Operativos	247	326
Formativos	142	179
Total	430	549
Desviación	+27,67 %	

Tabla 16: Comparativa entre las horas planificadas y las reales

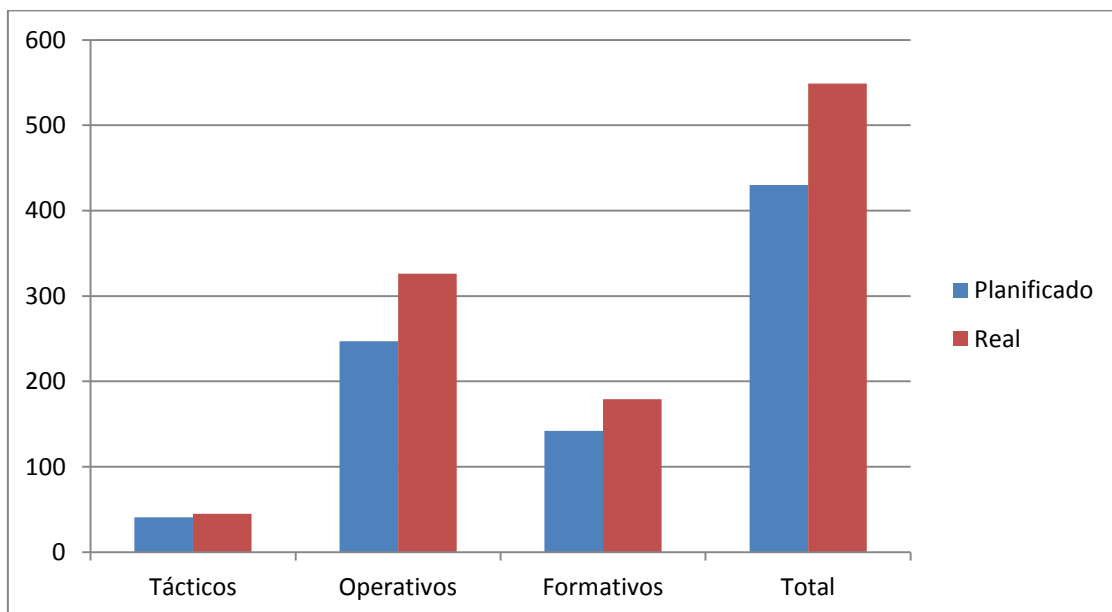


Figura 55: Gráfico entre las horas planificadas y reales.

En la tabla 16 y figura 55, podemos observar que ha habido una gran desviación (+27,67%), debida a que utilizábamos una tecnología desconocida, por lo que surgían problemas inesperados y la planificación no podía seguirse correctamente. Además de que en algunos casos, no se había planificado correctamente las horas necesarias para cada tarea.

11.3 Justificación de las desviaciones

Las mayores desviaciones vienen por el desconocimiento de la tecnología. En el caso de Android, se tuvo que dedicar bastante tiempo al aprendizaje del mismo, puesto que se desconocía completamente. Se realizó mediante tutoriales, manuales y el curso, que ayudó en gran medida al aprendizaje. No se vio necesario replanificar debido a que se intuía que se iban a dedicar bastantes horas al aprendizaje.

Otro de los problemas fue la base de datos, puesto que se perdió bastante tiempo por haber entendido de forma incorrecta la base de datos, así como por haber reducido el espacio que ocupaba la base de datos, que para una aplicación Android era excesivo. También se perdió tiempo a la hora de pasar los datos a la base de datos de Android, puesto que no se había realizado de forma correcta y se tuvo que reconstruir entera, perdiendo así un tiempo muy valioso.

Otro de los factores fue la incursión de nuevas funcionalidades a la aplicación, así como la creación de una aplicación independiente para la instalación de la base de datos en el dispositivo Android, debido a que en un principio, no se tenía planificado. Éstas mejoras se realizaron al observar que mejorarían de una forma significativa el funcionamiento de la aplicación y podría ser útil para una mayor gama de usuarios.

11.4 Incidencias

A lo largo del proyecto surgieron las siguientes incidencias:

- Enfermedad.

Descripción: Durante el desarrollo del proyecto, el alumno estuvo enfermo durante una semana, por lo que se retrasó el desarrollo.

Solución: Tras recuperarse, se repartió a lo largo de las siguientes semanas las horas que se tendrían que haber dedicado, pudiendo así recuperar las horas perdidas.

- USB estropeado.

Descripción: Por alguna razón desconocida, el usb que se utilizaba para almacenar la copia de seguridad, no dejaba seguir guardando información en él.

Solución: Aunque aún se tenía toda la información en el portátil, se decidió copiar el contenido del USB en otro ordenador, formatearlo y volver a copiar los antiguos archivos en él, así como las versiones más actuales. Gracias a que toda la información

la teníamos en el ordenador, puesto que esta solamente era la copia de seguridad, no se perdió nada y se pudo trabajar con normalidad.

12 Conclusiones

12.1 Objetivos cumplidos

Podemos sentirnos satisfechos, ya que hemos cumplido todos los objetivos propuestos para este proyecto. Esperamos que esta aplicación sirva a un gran número de personas y que sea tan útil como la versión online (MCR 3.0). En resumen, los objetivos cumplidos han sido los siguientes:

- Funcionamiento, tanto online, como offline.
- Resultados disponibles en cinco idiomas.
- Aplicación multilinguaje.
- Obtención de significados, traducciones, sinónimos, antónimos hiperónimos e hipónimos.

El proyecto me pareció interesante desde el principio, debido a que tenía que utilizar una tecnología que desconocía pero que quería aprender, puesto que actualmente es bastante popular. El mayor reto fue familiarizarme con todos los elementos necesarios para poder empezar a desarrollar, elegir qué tecnología usar y cómo planificarlo. Gracias a tutoriales, manuales y a un curso que realicé sobre Android, pude hacerme una idea de cómo empezar a realizarlo y qué es lo que quería o podía desarrollar. Uno de los temas que más me costó fue la base de datos, ya que en un principio entendí de forma incorrecta el funcionamiento en ciertas áreas de la misma. También tuve problemas a la hora de importar los datos a la base de datos que maneja Android (debido que el programa que utilizaba no lo realizaba correctamente). Aun así, estoy muy contento, ya que he tenido la oportunidad de pelearme con todos estos problemas y adquirir el conocimiento que de otra forma no habría podido conseguir.

Esta experiencia también me ha servido para aprender a gestionar un proyecto de éstas características, de principio a fin. Como hemos podido ir viendo en el desarrollo, el tiempo planificado no se correspondía con el real, pero gracias a esta experiencia, podemos establecer una planificación mejor en futuros proyectos. También se ha visto la importancia de tener un diseño claro desde el principio (tanto para la interfaz como para la base de datos), para no andar realizando muchos cambios y perder tiempo de forma innecesaria.

Estoy muy satisfecho con el resultado del proyecto, puesto que se han cumplido todos los objetivos propuestos durante el mismo. También estoy contento de la oportunidad que he tenido de ampliar mi conocimiento hacia zonas que desconocía completamente y que puede que utilice en un futuro.

12.2 Trabajos futuros

12.2.1 Alojamiento en internet

Alajar la aplicación en algún gestor de aplicaciones Android (como puede ser Google Play) para que una mayor cantidad de usuarios puedan acceder a ella, así como disponer de un servidor para almacenar la base de datos de la aplicación.

12.2.2 Actualizaciones

Mantener la aplicación actualizada para una mayor compatibilidad con las futuras versiones del sistema operativo Android, así como para ir mejorándola poco a poco, si fuese preciso. Esto es algo crucial, puesto que si la aplicación no es adaptada a nuevas tecnologías, puede que surjan incompatibilidades y deje de funcionar.

12.2.3 Otros sistemas operativos

Llevar esta aplicación a otros sistemas operativos, como puede ser iOS, para una mayor difusión y disponibilidad. Esto es necesario si queremos ampliar el mercado en el que nuestra aplicación funcionaría.

12.2.4 Ampliar base de datos

Añadir más significados y ejemplos a la base de datos, debido a que, en algunos casos, no se disponía de una gran cantidad de información. En el caso del inglés, se disponía de gran información, pero para el resto, no se disponía de tanto, por lo que en algunos casos, podría parecer escaso y poco útil.

13 Bibliografía

1) Android documentación:

- <http://www.androidcurso.com/index.php/tutoriales-android-fundamentos/37-unidad-6-multimedia-y-ciclo-de-vida/158-ciclo-de-vida-de-una-actividad>
- <http://androidzone.org/2012/08/estructura-de-un-proyecto-android/>
- <http://www.startcapps.com/blog/cuota-de-mercado-de-los-smartphones-2013/>
- <http://ganaclin.blogspot.com.es/2012/11/estructura>

2) Aprendizaje Android:

- <http://developer.android.com/intl/es/index.html>
- http://www.sgoliver.net/blog/?page_id=3011
- https://www.coursera.org/specialization/mobilecloudcomputing/2?utm_medium=catalogSpec

3) Información general:

- www.wikipedia.es

4) Documentación WordNet y MCR 3.0:

- <http://adimen.si.ehu.es/web/MCR>
- <http://adimen.si.ehu.es/cgi-bin/wei/public/wei.consult.perl>
- <http://adimen.si.ehu.es/web/Resources>
- Artículo: Construcción de una base de conocimiento léxico multilingüe de amplia cobertura: Multilingual Central Repository (German Rigau, Aitor Gonzalezjz-Aguirre)
- <http://wordnet.princeton.edu/>