Grado en Ingeniería Informática

Trabajo Final de Grado

Desarrollo de herramientas para la mejora de la experiencia de usuario en la biblioteca de la universidad

Autor:
Miguel Matey Sanz

Supervisor:
Andres Muñoz Zuluaga
Tutor académico:
Ismael Sanz Blasco

Fecha de lectura: 28 de junio de 2019
Curso académico 2018/2019

Resumen

En este documento se detalla todo el proceso de desarrollo de varios módulos de funcionalidad para una aplicación móvil Android para la biblioteca de la Universitat Jaume I de Castellón. El proyecto se realiza dentro de la estancia en prácticas en Ubik Geospatial Solutions S.L. en el grado en Ingeniería Informática de la Universitat Jaume I.

La aplicación busca mejorar la experiencia de los usuarios de la biblioteca, permitiendo la búsqueda de libros y de cabinas para conocer su localización así como su disponibilidad, y el geoposicionamiento del usuario dentro del edificio mediante tecnología Wi-Fi o BLE (*Bluetooth Low Energy*) dependiendo de la zona donde se encuentre el usuario.

Para llevar a cabo el desarrollo de estas funcionalidades se sigue una metodología de gestión tradicional en cascada para cada uno de los módulos a implementar. Para cada módulo se realiza el correspondiente seguimiento de la planificación, la definición de requisitos y el análisis, el diseño, y la implementación y pruebas.

Todos los objetivos iniciales del proyecto se han cumplido, teniendo incluso publicada la aplicación en la Google Play con parte de la funcionalidad implementada.

Palabras clave

Android, Node.js, Wi-Fi, BLE, Fingerprinting, Trilateración

Keywords

Android, Node.js, Wi-Fi, BLE, Fingerprinting, Trilateration

Índice general

Capítulo 1

Introducción

1.1. Contexto y motivación del proyecto

El proyecto que se presenta en este documento va a desarrollarse durante el periodo de prácticas externas del grado en Ingeniería Informática de la Universitat Jaume I (UJI), en la empresa Ubik Geospatial Solutions S.L. [33] (Ubik en adelante), situada en el Parque Científico, Tecnológico y Empresarial de la UJI.

Ubik se encarga de ofrecer soluciones web y aplicaciones móvil a problemas que requieren la integración de información geoespacial como, por ejemplo, el desarrollo de aplicaciones de *Smart Cities* o *Web Mapping*, la integración de datos e información geográfica, o el desarrollo de aplicaciones móviles basadas en la localización del usuario. El producto desarrollado durante este proyecto se corresponde con este último ámbito.

Así pues, en este proyecto se han desarrollado una serie de herramientas para un prototipo de aplicación móvil para dispositivos Android para la localización de libros y posicionamiento del usuario en la biblioteca de la universidad. El objetivo de esta aplicación será dar soporte en la búsqueda de libros por parte de los usuarios de la biblioteca (estudiantes, profesores, bibliotecarios...), además de proporcionar una estimación de la posición del usuario dentro del edificio para facilitar su orientación. Por lo tanto, la aplicación deberá dar soporte a:

- La búsqueda de libros presentes en la biblioteca mediante su título, autor, o topográfico.
- La búsqueda de cabinas de la biblioteca.
- La estimación de la posición del usuario a tiempo real con el mínimo retardo y mayor precisión posibles mediante diversas técnicas de localización en interiores.
- Facilitar la orientación del usuario dentro del edificio mediante la representación de lugares significativos en los mapas de la aplicación.

La motivación para el desarrollo de esta aplicación reside en su gran utilidad para los usuarios de la biblioteca, reduciendo el tiempo empleado en la búsqueda de los libros, ya que si bien estos

pueden localizarse manualmente mediante los topográficos, no es una tarea trivial descifrarlos. Además, se tiene la ventaja de poder realizar las búsquedas desde cualquier lugar se encuentre el usuario en la biblioteca o no, pudiendo saber si el libro se encuentra disponible o no, o bien si hay cabinas disponibles. Por otro lado, la aplicación también permitirá prescindir de los equipos habilitados para la búsqueda de libros distribuidos por el edificio y de sus costes de mantenimiento.

Otra fuente importante en la motivación del desarrollo de la aplicación es el grado de innovación tecnológica dentro de un ámbito con un gran interés para la investigación, como es el posicionamiento en interiores. Actualmente la localización en exteriores es un problema bien resuelto con las técnicas de GNSS (*Global Navigation Satellite System*). Sin embargo, para la localización en interiores no existe ninguna solución satisfactoria en el mercado, y el desarrollo del "GPS (*Global Positioning System*) de interiores", uno de los productos más codiciados por la industria de las tecnologías geoespaciales. Se estima que supondría un impacto económico de hasta $41.000 millones [20].

Además, el posicionamiento en interiores está abierto a un gran número de tecnologías y métodos de posicionamiento, donde la solución más apropiada depende del contexto, del entorno, y de la precisión requerida, entre otros requisitos [8]. En este proyecto se utilizan las intensidades de las señales Wi-Fi y BLE (*Bluetooth Low Energy*) en el posicionamiento, donde se emplean distintos enfoques:

- *KNN Fingerprinting* (k vecinos más próximos): técnica empleada en este proyecto con las señales Wi-Fi. Se crea una base de datos de muestras de las intensidades detectadas para cada emisor en varios puntos de referencia. Luego se usa esa base de datos para determinar la posición nuevas señales medidas. Esta es una técnica popular debido a su razonable margen de error (3-6 metros) y a que no conlleva costes de despliegue. Por otro lado, la señal se ve altamente afectada por el tránsito de personas o mobiliario, entre otros aspectos. [30].

- *Weighted Centroid* (Centroide ponderado): técnica empleada en este proyecto con las señales BLE. Se realiza un despliegue de balizas (también llamadas *beacon*), de las cuales se conocen sus posiciones. Al recibir señales, se obtienen las k señales de mayor intensidad, y como se conoce de qué *beacons* provienen se estima la posición del usuario. Esta técnica es popular ya que proporciona precisiones superiores a las empleadas en Wi-Fi, debido a poder tener una mayor densidad de antenas emitiendo señales a mayores frecuencias. Por otro lado, se debe tener en cuenta el coste del despliegue de estos *beacons*, así como de su mantenimiento.

Otro elemento motivador es la colaboración e interacción con expertos en materia de posicionamiento en interiores del grupo de investigación GEOTEC (Geospatial Technologies Research Group) de la UJI, en especial el doctor Joaquín Torres Sospedra y el estudiante de doctorado Germán Martín Mendoza Silva.

1.2. Objetivos del producto y del proyecto

El objetivo del producto a desarrollar es mejorar la experiencia de usuario en el marco del uso de algunos servicios de la biblioteca, como son la búsqueda de libros y de cabinas, además de conocer su disponibilidad. Por otra parte, el producto permitirá al usuario conocer su ubicación aproximada dentro del edificio para así facilitar su orientación. Esta aplicación permitirá a sus usuarios localizar libros y cabinas rápidamente, y desde un punto de vista estratégico supondrá para la biblioteca y para la universidad disponer de una aplicación novedosa y sin precedentes en España.

Por otra parte, el principal objetivo de este proyecto es la mejora de un prototipo de aplicación móvil Android que permite la búsqueda de libros y de cabinas y el posicionamiento del usuario dentro de la biblioteca de la UJI. Este objetivo puede desglosarse en los siguientes subobjetivos:

- Desarrollar un algoritmo *KNN Fingerprinting* para estimar posiciones mediante señales Wi-Fi, optimizado para ser ejecutado en terminales móviles, empleando técnicas de concurrencia para reducir los tiempos de cómputo.
- Desarrollar un algoritmo *Weighted Centroid* para estimar posiciones mediante señales BLE y algunas variantes para ser ejecutado en terminales móviles.
- Análisis de la configuración óptima de los *beacons* BLE para maximizar su duración de batería y minimizar el error de estimación.
- Desarrollo de un algoritmo de búsqueda de libros por topográfico.
- Desarrollo de un algoritmo de búsqueda de cabinas.
- Inclusión de elementos en los mapas que faciliten la orientación del usuario y mejoras en la interfaz de usuario a nivel de usabilidad.

Desde un punto de vista estratégico, el objetivo del proyecto es dotar a la UJI de una aplicación móvil útil que emplee técnicas novedosas de localización en interiores.

A nivel operacional y táctico, el producto facilitará la búsqueda de libros en la biblioteca de la universidad y la orientación de los usuarios dentro de la misma.

1.3. Alcance del proyecto

Por lo que respecta al alcance del proyecto, hay que tener en cuenta que no se empieza el desarrollo de la aplicación desde cero. Una aplicación de estas características sería complicada de desarrollar en tan poco tiempo. Por tanto, se parte de una aplicación prototipo que permite la búsqueda de libros limitada a títulos y autores y la estimación de la posición del usuario mediante consultas a un servidor que se encarga de realizar la predicción gracias a las redes Wi-Fi que detecta el dispositivo. El prototipo también cuenta con los módulos necesarios para mostrar el

mapa de la biblioteca y sus estanterías. El sistema actual se describirá más detalladamente en el Capítulo 2.

Por lo tanto, el alcance del proyecto se enmarca en la mejora del proceso de estimación de la posición del usuario, integrando todos los cálculos en el dispositivo, lo que permite prescindir del servidor que se encarga de las predicciones. También se implementará un sistema de estimación basado en *beacons Bluetooth Low Energy* (BLE) para conseguir una mayor precisión que con el sistema Wi-Fi actual. Este sistema solo estará disponible en la quinta planta del edificio, ya que conlleva el despliegue de una cantidad considerable de *beacons*, y requerirá de un proceso de investigación y pruebas para encontrar la mejor configuración y despliegue de los mismos. Además, se mejorará la búsqueda de libros, permitiendo buscar por los topográficos de los mismos, y se añadirá la búsqueda de cabinas. Finalmente se realizarán algunas mejoras en los mapas de la aplicación y la interfaz para añadir elementos representativos (servicios, escaleras...) para facilitar la orientación del usuario.

Como resumen, se definirá el alcance desde el ámbito organizacional, funcional y tecnológico.

En cuanto al alcance organizacional, la aplicación será empleada por usuarios externos a la empresa donde se va a desarrollar. Estos usuarios serán todas aquellas personas que hagan uso de la biblioteca de UJI y necesiten localizar libros en la misma.

Por lo que respecta al alcance funcional, la aplicación tendrá las siguientes funcionalidades:

- Se debe permitir la búsqueda de libros presentes en la biblioteca mediante el uso de sus topográficos (parciales o completos).
- Se debe permitir la búsqueda de cabinas de la biblioteca mediante su identificador.
- El sistema debe poder estimar la localización del usuario dentro del edificio de la UJI, empleando los recursos disponibles. Estos recursos serán las señales Wi-Fi de las antenas proveedoras de acceso a Internet del edificio y de los *beacons* BLE que se desplegarán en la quinta planta del edificio.

Finalmente, en cuanto al alcance tecnológico, se sustituirá del actual prototipo el servidor de estimación de posiciones de los usuarios llevando los cálculos a los terminales de los usuarios, reduciendo la latencia en las mismas. También se debe tener en cuenta la necesidad de acceso a Internet para la búsqueda de libros, ya que se deben realizar peticiones al catálogo de la biblioteca, y para la descarga de mapas y otros datos desde Firebase (plataforma con servicios en la nube para el desarrollo de aplicaciones, entre ellos, bases de datos) durante el primer inicio de la aplicación.

1.4. Estructura de la memoria

Los siguientes capítulos de este documento aportan información relevante sobre el proyecto, así como las etapas de su desarrollo. En el Capítulo 2 se describe la estructura del prototipo de partida sobre el cual se ha realizado el desarrollo, así como las tecnologías que este emplea. En

el Capítulo 3 se empieza con la planificación del proyecto, donde se define la metodología de desarrollo usada, planificación inicial, estimación de costes y el seguimiento del proyecto. En el Capítulo 4 se muestran los requisitos y el análisis para cada uno de los módulos desarrollados, y en el Capítulo 5 se muestra el diseño de cada uno de los módulos. Tras el análisis y el diseño, el documento sigue con la implementación y las pruebas en el Capítulo 6, donde para cada módulo se muestran ciertos detalles de la implementación así como las pruebas realizadas para validar su correcto funcionamiento. Finalmente, en el Capítulo 7 se encuentran las conclusiones tras la realización de este proyecto.

Capítulo 2

Estado inicial del sistema

2.1. Descripción del sistema inicial

Como se ha dicho en el apartado del alcance en el Capítulo 1, el desarrollo de este proyecto tiene como base una aplicación prototipo, la cual presenta las siguientes funcionalidades:

- Muestra del mapa de la biblioteca de la UJI, incluyendo las librearías, selector de planta y buscado de libros (véase Figura 2.1).
- Localización del usuario mediante GPS: esta funcionalidad solicita al sistema operativo Android determinar la localización del terminal haciendo uso del GPS. Al emplearse dentro de un edificio su confianza es mínima.
- Localización del usuario mediante la red Wi-Fi a la que está conectado el dispositivo: esta funcionalidad solicita al sistema operativo Android determinar la localización del terminal mediante los puntos de acceso Wi-Fi y antenas de cobertura que están al alcance del dispositivo.
- Localización del usuario mediante las intensidades de las señales Wi-Fi que ve el dispositivo: el dispositivo envía dichas intensidades capturadas al servidor encargado de calcular la estimación de la posición, y que una vez calculada se la envía al dispositivo.
- Posibilidad de selección del sistema de localización empleado (véase Figura 2.2): el sistema permite la activación y desactivación de cada uno de los sistemas de posicionamiento.
- Fusión de estimaciones con una determinada confianza dependiendo del método de estimación del que provienen.
- Búsqueda de libros mediante palabras clave (autor, palabra contenida en el título, topográfico completo) (véase Figura 2.3).

En la Figura 2.4 puede verse el esquema del sistema inicial, donde se muestran los diversos módulos que forman parte de la aplicación. Como puede verse, la aplicación tiene tres módulos principales:

Figura 2.1: Mapa de la biblioteca con selector de planta y buscador de libros.

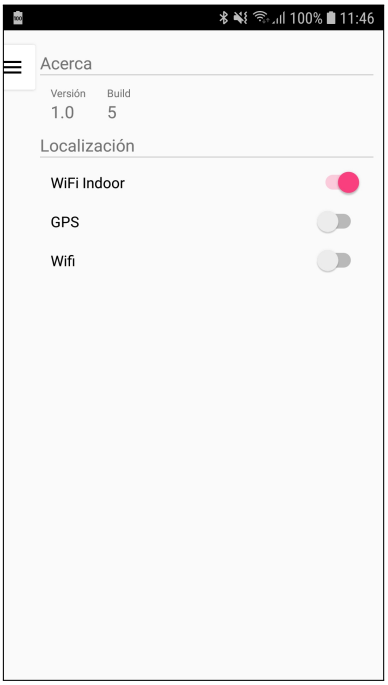
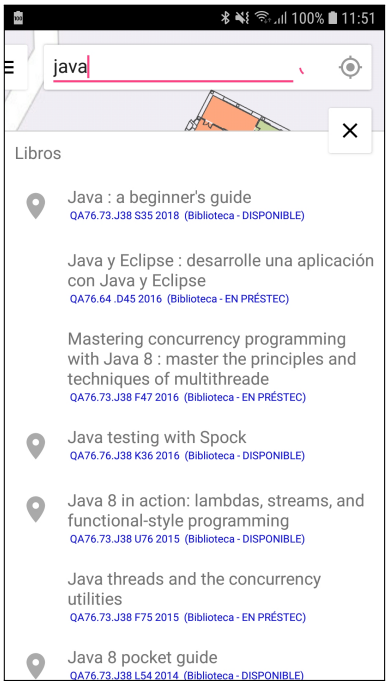


Figura 2.2: Ajustes con los interruptores de activación de cada sistema de posicionamiento.



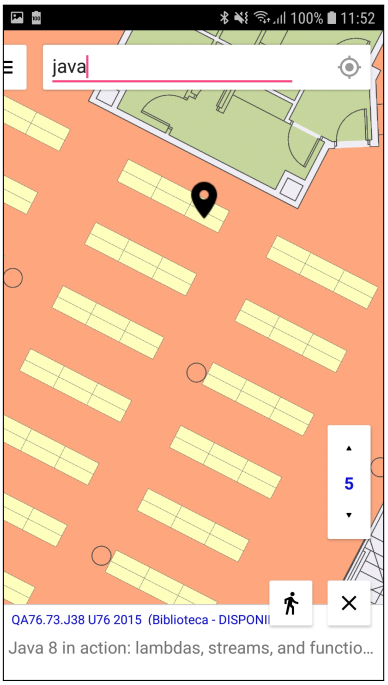


Figura 2.3: Búsqueda de libro con la palabra clave "java"(izquierda) y localización del libro seleccionado *Java 8 in action: lambdas, streams and functional-style programming* (derecha).

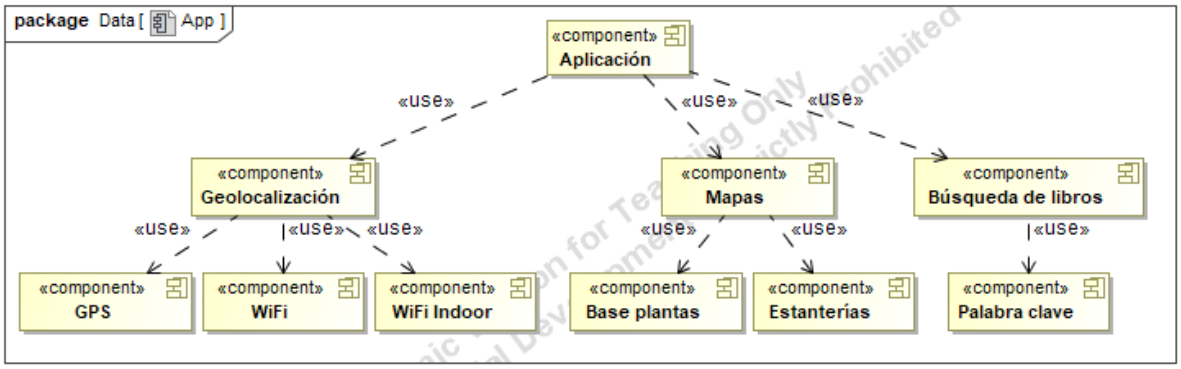


Figura 2.4: Esquema con los módulos del sistema inicial.

- Geolocalización: encargado de realizar las estimaciones del usuario. Este módulo está compuesto por los distintos componentes de localización. Durante el desarrollo se modificará el componente *Wi-Fi Indoor* (módulo 1) y se añadirá otro componente para estimaciones con señales BLE (módulo 2).

- Mapas: encargado de obtener los mapas de las plantas de la biblioteca junto con sus características y estanterías. En el módulo 3 se modificará el componente *Base plantas* para añadir elementos a éstos.

- Búsqueda de libros: encargado de realizar la búsqueda de libros mediante peticiones web al catálogo de la biblioteca. En el módulo 3 se añadirá un componente para la búsqueda por topográfico.

2.2. Tecnologías del proyecto

Las tecnologías más destacables empleadas en el proyecto son Android SDK, JDK, ArcGIS Runtime SDK for Android, ArcGIS Pro, JUnit, Mockito, Node.js, Swagger, Docker, Firebase y Git. En las siguientes subsecciones se describirán estas tecnologías.

2.2.1. Android SDK

Android SDK (*Software Development Kit*) proporciona herramientas de desarrollo de aplicaciones Android, como bibliotecas, documentación, un depurador, o un simulador de teléfonos [2]. En concreto se emplea la versión 26 del SDK, orientado para los dispositivos Android 8.0 o superior.

El entorno de desarrollo soportado es Android Studio [3], basado en IntelliJ IDEA [15].

2.2.2. JDK

El JDK (*Java Development Kit*) se requiere para el desarrollo de aplicaciones Android, ya que Java es el lenguaje de programación que se emplea. El JDK, al igual que Android SDK, proporciona las herramientas necesarias para el desarrollo de programas con el lenguaje Java [16]. En concreto, se emplea la versión 8 del lenguaje.

2.2.3. ArcGIS Runtime SDK for Android

ArcGIS Runtime SDK for Android proporciona las herramientas necesarias para incorporar en las aplicaciones Android mapas, geoprocesamiento, creación de rutas, entre otros. En concreto, se emplea la versión del SDK 10.2.9 [7].

Estas herramientas forman parte de ArcGIS, un conjunto de productos de software en el campo de los sistemas de información geográfica [5].

2.2.4. ArcGIS Pro

ArcGIS Pro es un software que forma parte de la familia de herramientas de ArcGIS, y que permite la creación, visualización, análisis y edición de mapas y datos geoespaciales [6]. Este producto se emplea en el proyecto para editar las capas de la *geodatabase* (estructura de datos nativa de ArcGIS para representar y administrar información geográfica) que contiene los datos de las plantas, estanterías y otros elementos de la biblioteca.

2.2.5. JUnit y Mockito

JUnit es un conjunto de bibliotecas que permiten realizar pruebas unitarias sobre aplicaciones Java, ejecutando clases de manera controlada para comprobar si los métodos bajo prueba actúan de la forma esperada. Se emplea la versión 4.12. Aunque el *framework* está orientado a realizar pruebas unitarias, también puede emplearse para realizar pruebas de integración y de aceptación.

Además se utiliza Mockito, un *framework* de pruebas que permite crear dobles de pruebas (*mocks*) para poder probar un determinado método sin sus dependencias reales.

2.2.6. Node.js

Node.js es un entorno de ejecución del lado del servidor que permite crear programas de red altamente escalables, como servidores web [22]. En el proyecto se emplea para el servicio de búsqueda de libros. El lenguaje utilizado es JavaScript, y como entorno de desarrollo integrado se emplea WebStorm [32].

2.2.7. Swagger

Swagger es un *framework* que proporciona herramientas para diseñar, documentar y consumir servicios REST (*Representational State Transfer*)[26]. En el proyecto se emplea Swagger UI, que permite interactuar con la API (*Application Programming Interface*) mediante una documentación generada automáticamente [28]. También se emplea Swagger Codegen para generar el código del cliente que realiza las llamadas a los servicios REST que expone el servidor [27].

2.2.8. Docker

Docker permite la automatización de despliegue de aplicaciones dentro de contenedores de software, proporcionando una capa adicional de abstracción y evitando el inicio y mantenimiento de máquinas virtuales [10]. Docker se emplea para para desplegar el servicio de búsqueda de libros en Node.js.

2.2.9. Firebase

Firebase es una plataforma para el desarrollo de aplicaciones web y aplicaciones móviles que proporciona servicios de mensajería y notificaciones (*Firebase Cloud Messaging*), de autenticación (*Firebase Auth*), o de almacenamiento de información (*Realtime Database* y *Firebase Firestore*) entre otros [12]. En concreto, en este proyecto se emplea *Firebase Firestore*, una base de datos basada en documentos, para almacenar las capas correspondientes a los mapas de la biblioteca e información relacionada con las localizaciones, y *Firebase Test Lab*, una infraestructura de prueba de aplicaciones basada en la nube [29].

2.2.10. Git

Git es un sistema de control de versiones, el cual mantiene un registro de los cambios que sufren un conjunto de ficheros, lo que permite en cualquier momento volver a una versión de los ficheros anterior [13]. Además, en este proyecto se emplea de forma distribuida con un repositorio en GitHub [14], cosa que es valiosa cuando se trabaja en equipo.

Capítulo 3

Planificación del proyecto

3.1. Metodología

En Ubik no se sigue una metodología de desarrollo fija y/o estándar, aunque si que entraría dentro de un proceso ágil. El supervisor del proyecto dio libertad en cuanto a la metodología a utilizar en el desarrollo. Debido a las características del proyecto, el conocimiento de las tecnologías y la disposición de unos requisitos estables y bien definidos se ha empleado una metodología tradicional o predictiva. En concreto se emplea el modelo de desarrollo en cascada, aunque con algunas variaciones:

- El desarrollo se compone de varias fases, las cuales contienen las actividades del modelo en cascada. Es decir, no habrá una única fase de análisis, una única fase de diseño, y una única fase de implementación, sino que habrán varias fases de análisis, de diseño y de implementación. Pese a que puede parecer que sería mejor emplear alguna metodología iterativa por el hecho de realizar varias veces las fases, estas son independientes entre sí, es decir, el análisis de un módulo no tiene que ver con el de otro, por lo que se descarta la metodología iterativa, que esta más basada en el refinamiento de las fases en cada iteración.
- Al finalizar cada fase, se dispondrá de un producto que aportará valor para el cliente. Debido a la independencia de los módulos a implementar, al finalizar el desarrollo de cada uno de los módulos se aportará valor añadido para el usuario.

3.2. Planificación inicial

A continuación se enumeran las tareas identificadas en la planificación inicial del proyecto, así como su duración estimada.

- Desarrollo de la propuesta técnica (Total: 40h)

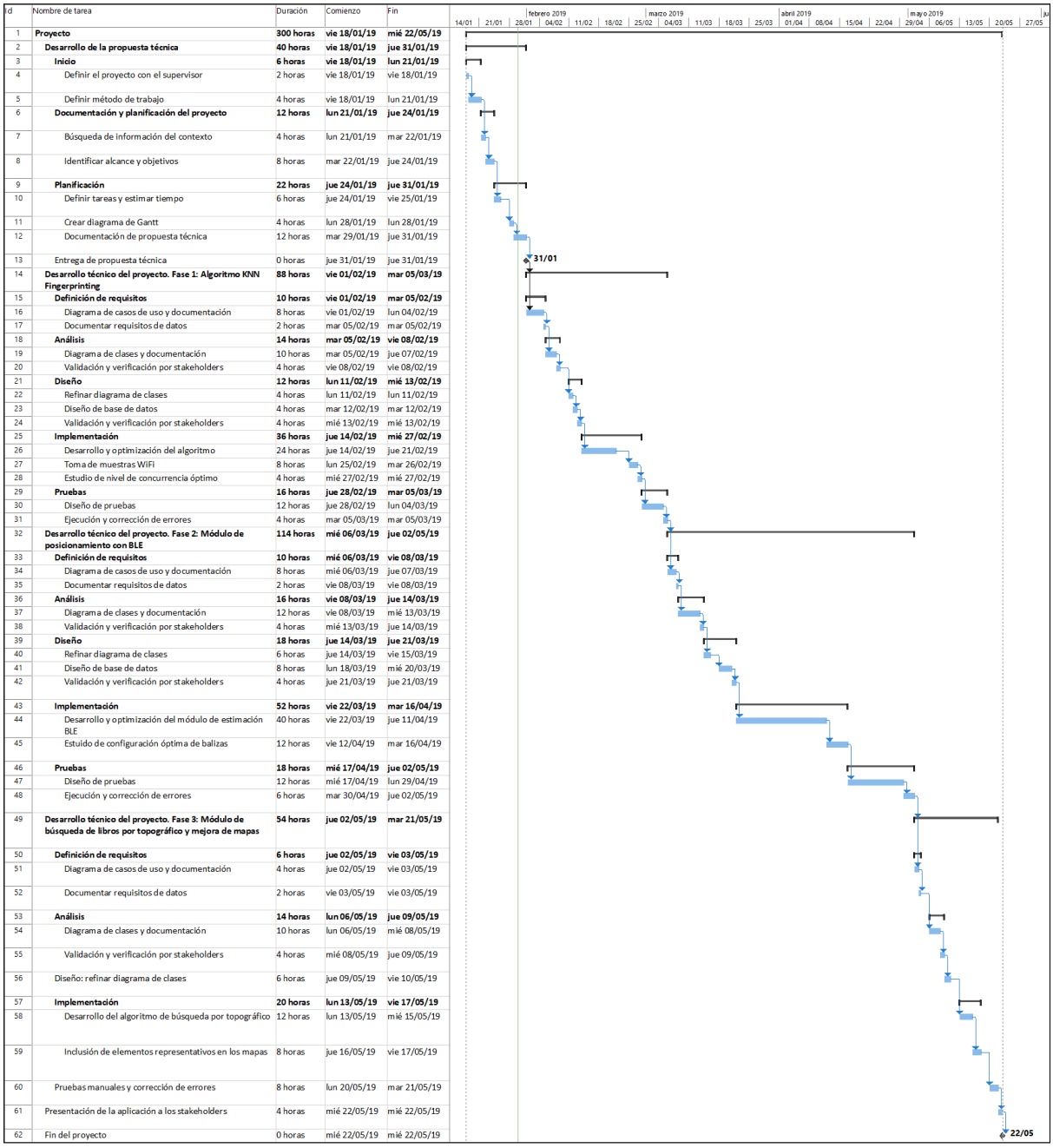
- Inicio
  - Definir el proyecto con el tutor y el supervisor (2h)
  - Definir método de trabajo (4h)
- Documentación y planificación del proyecto
  - Búsqueda de información del contexto (4h)
  - Identificar alcance y objetivos (8h)
- Planificación
  - Definir tareas y estimar tiempo (6h)
  - Crear diagrama de Gantt (4h)
  - Documentación de propuesta técnica (12h)
- Entrega de propuesta técnica (0h)

- Desarrollo técnico del proyecto. Módulo 1: Algoritmo *KNN Fingerprinting* (Total: 88h)
  - Definición de requisitos
    - Diagrama de casos de uso y documentación (8h)
    - Documentar requisitos de datos (2h)
  - Análisis
    - Diagrama de clases y documentación (10h)
    - Validación y verificación por *stakeholders* (4h)
  - Diseño
    - Refinar diagrama de clases (4h)
    - Diseño de base de datos (4h)
    - Validación y verificación por *stakeholders* (4h)
  - Implementación
    - Desarrollo y optimización del algoritmo (24h)
    - Toma de muestras Wi-Fi: huellas Wi-Fi en la biblioteca para realizar estimaciones (8h)
    - Estudio de nivel de concurrencia óptimo: rendimiento empleando distintas cantidades de hilos (4h)
  - Pruebas
    - Diseño de pruebas (12h)
    - Ejecución y corrección de errores (4h)

- Desarrollo técnico del proyecto. Módulo 2: Posicionamiento con BLE (Total: 114h)
  - Definición de requisitos
    - Diagrama de casos de uso y documentación (8h)
    - Documentar requisitos de datos (2h)
  - Análisis
    - Diagrama de clases y documentación (12h)
    - Validación y verificación por *stakeholders* (4h)
  - Diseño

        - Refinar diagrama de clases (6h)
        - Diseño de base de datos (8h)
        - Validación y verificación por *stakeholders* (4h)
    - Implementación
        - Desarrollo y optimización del módulo de estimación BLE (40h)
        - Estudio de configuración óptima de balizas: cantidad, distribución, frecuencia y potencia (12h)
    - Pruebas
        - Diseño de pruebas (12h)
        - Ejecución y corrección de errores (6h)
- Desarrollo técnico del proyecto. Módulo 3: Búsqueda de libros por topográfico y mejora de mapas (Total: 54h)
    - Definición de requisitos
        - Diagrama de casos de uso y documentación (4h)
        - Documentar requisitos de datos (2h)
    - Análisis
        - Diagrama de clases y documentación (10h)
        - Validación y verificación por *stakeholders* (4h)
    - Diseño: refinar diagrama de clases (6h)
    - Implementación
        - Desarrollo del algoritmo de búsqueda por topográfico (12h)
        - Inclusión de elementos representativos en los mapas (8h)
    - Pruebas manuales y corrección de errores (8h)
- Presentación de la aplicación a los *stakeholders* (4h)
- Fin del proyecto (0h)
- Documentación y presentación del TFG (Total: 150h)
    - Informes quincenales (4h)
    - Desarrollo de la memoria técnica (110h)
    - Entrega de memoria técnica (0h)
    - Preparación de la presentación oral (35h)
    - Presentación oral (1h)

En la Figura 3.1 se muestra la organización de las tareas correspondientes al desarrollo y su estructura temporal mediante un diagrama de Gantt. Aunque hay tareas que podrían realizarse paralelamente en un equipo de trabajo de varios miembros, estas se planifican de forma secuencial, ya que una única persona no puede trabajar en varias tareas a la vez. Como puede verse, la suma de tiempo en estas tareas asciende a 300 horas.



Id	Nombre de tarea	Duración	Comienzo	Fin
1	**Proyecto**	**300 horas**	**vie 18/01/19**	**mié 22/05/19**
2	**Desarrollo de la propuesta técnica**	**40 horas**	**vie 18/01/19**	**jue 31/01/19**
3	**Inicio**	**6 horas**	**vie 18/01/19**	**lun 21/01/19**
4	Definir el proyecto con el supervisor	2 horas	vie 18/01/19	vie 18/01/19
5	Definir método de trabajo	4 horas	vie 18/01/19	lun 21/01/19
6	**Documentación y planificación del proyecto**	**12 horas**	**lun 21/01/19**	**jue 24/01/19**
7	Búsqueda de información del contexto	4 horas	lun 21/01/19	mar 22/01/19
8	Identificar alcance y objetivos	8 horas	mar 22/01/19	jue 24/01/19
9	**Planificación**	**22 horas**	**jue 24/01/19**	**jue 31/01/19**
10	Definir tareas y estimar tiempo	6 horas	jue 24/01/19	vie 25/01/19
11	Crear diagrama de Gantt	4 horas	lun 28/01/19	lun 28/01/19
12	Documentación de propuesta técnica	12 horas	mar 29/01/19	jue 31/01/19
13	Entrega de propuesta técnica	0 horas	jue 31/01/19	jue 31/01/19
14	**Desarrollo técnico del proyecto. Fase 1: Algoritmo KNN Fingerprinting**	**88 horas**	**vie 01/02/19**	**mar 05/03/19**
15	**Definición de requisitos**	**10 horas**	**vie 01/02/19**	**mar 05/02/19**
16	Diagrama de casos de uso y documentación	8 horas	vie 01/02/19	lun 04/02/19
17	Documentar requisitos de datos	2 horas	mar 05/02/19	mar 05/02/19
18	**Análisis**	**14 horas**	**mar 05/02/19**	**vie 08/02/19**
19	Diagrama de clases y documentación	10 horas	mar 05/02/19	jue 07/02/19
20	Validación y verificación por stakeholders	4 horas	vie 08/02/19	vie 08/02/19
21	**Diseño**	**12 horas**	**lun 11/02/19**	**mié 13/02/19**
22	Refinar diagrama de clases	4 horas	lun 11/02/19	lun 11/02/19
23	Diseño de base de datos	4 horas	mar 12/02/19	mar 12/02/19
24	Validación y verificación por stakeholders	4 horas	mié 13/02/19	mié 13/02/19
25	**Implementación**	**36 horas**	**jue 14/02/19**	**mié 27/02/19**
26	Desarrollo y optimización del algoritmo	24 horas	jue 14/02/19	jue 21/02/19
27	Toma de muestras WiFi	8 horas	lun 25/02/19	mar 26/02/19
28	Estudio de nivel de concurrencia óptimo	4 horas	mié 27/02/19	mié 27/02/19
29	**Pruebas**	**16 horas**	**jue 28/02/19**	**mar 05/03/19**
30	Diseño de pruebas	12 horas	jue 28/02/19	lun 04/03/19
31	Ejecución y corrección de errores	4 horas	mar 05/03/19	mar 05/03/19
32	**Desarrollo técnico del proyecto. Fase 2: Módulo de posicionamiento con BLE**	**114 horas**	**mié 06/03/19**	**jue 02/05/19**
33	**Definición de requisitos**	**10 horas**	**mié 06/03/19**	**vie 08/03/19**
34	Diagrama de casos de uso y documentación	8 horas	mié 06/03/19	jue 07/03/19
35	Documentar requisitos de datos	2 horas	vie 08/03/19	vie 08/03/19
36	**Análisis**	**16 horas**	**vie 08/03/19**	**jue 14/03/19**
37	Diagrama de clases y documentación	12 horas	vie 08/03/19	mié 13/03/19
38	Validación y verificación por stakeholders	4 horas	mié 13/03/19	jue 14/03/19
39	**Diseño**	**18 horas**	**jue 14/03/19**	**jue 21/03/19**
40	Refinar diagrama de clases	6 horas	jue 14/03/19	vie 15/03/19
41	Diseño de base de datos	8 horas	lun 18/03/19	mié 20/03/19
42	Validación y verificación por stakeholders	4 horas	jue 21/03/19	jue 21/03/19
43	**Implementación**	**52 horas**	**vie 22/03/19**	**mar 16/04/19**
44	Desarrollo y optimización del módulo de estimación BLE	40 horas	vie 22/03/19	jue 11/04/19
45	Estuido de configuración óptima de balizas	12 horas	vie 12/04/19	mar 16/04/19
46	**Pruebas**	**18 horas**	**mié 17/04/19**	**jue 02/05/19**
47	Diseño de pruebas	12 horas	mié 17/04/19	lun 29/04/19
48	Ejecución y corrección de errores	6 horas	mar 30/04/19	jue 02/05/19
49	**Desarrollo técnico del proyecto. Fase 3: Módulo de búsqueda de libros por topográfico y mejora de mapas**	**54 horas**	**jue 02/05/19**	**mar 21/05/19**
50	**Definición de requisitos**	**6 horas**	**jue 02/05/19**	**vie 03/05/19**
51	Diagrama de casos de uso y documentación	4 horas	jue 02/05/19	vie 03/05/19
52	Documentar requisitos de datos	2 horas	vie 03/05/19	vie 03/05/19
53	**Análisis**	**14 horas**	**lun 06/05/19**	**jue 09/05/19**
54	Diagrama de clases y documentación	10 horas	lun 06/05/19	mié 08/05/19
55	Validación y verificación por stakeholders	4 horas	mié 08/05/19	jue 09/05/19
56	Diseño: refinar diagrama de clases	6 horas	jue 09/05/19	vie 10/05/19
57	**Implementación**	**20 horas**	**lun 13/05/19**	**vie 17/05/19**
58	Desarrollo del algoritmo de búsqueda por topográfico	12 horas	lun 13/05/19	mié 15/05/19
59	Inclusión de elementos representativos en los mapas	8 horas	jue 16/05/19	vie 17/05/19
60	Pruebas manuales y corrección de errores	8 horas	lun 20/05/19	mar 21/05/19
61	Presentación de la aplicación a los stakeholders	4 horas	mié 22/05/19	mié 22/05/19
62	Fin del proyecto	0 horas	mié 22/05/19	mié 22/05/19

Figura 3.1: Diagrama de Gantt con la lista de tareas junto con la planificación temporal y dependencias entre ellas.

Documentación y presentación del TFG	150 horas
Informes quincenales	4 horas
Desarrollo de la memoría técnica	110 horas
Entrega de memoría técnica	0 horas
Preparación de la presentación oral	35 horas
Presentación oral	1 hora

Figura 3.2: Actividades correspondientes a la documentación y presentación del TFG con su correspondiente estimación de coste temporal.

En la Figura 3.2 se muestran el tiempo necesario para la realización de las tareas correspondientes a la documentación y presentación de este documento. Como puede verse, la suma de tiempo en estas tareas asciende a 150 horas.

Aunque los requisitos sean muy claros, esta planificación se puede ver afectada por los numerosos inconvenientes que pueden surgir durante el desarrollo del proyecto, por lo que en la sección 3.5 se realizará el seguimiento del proyecto, indicando todos los inconvenientes y variaciones con respecto a la planificación.

3.3. Estimación de recursos y costes del proyecto

En esta sección se muestra la estimación de recursos y costes del proyecto, los cuales se clasifican en recursos humanos, de hardware, de software, e indirectos.

En primer lugar, el coste de recursos humanos está compuesto por las horas trabajadas por el autor de este documento en cada uno de los roles identificados. Para estimar este coste se tiene en cuenta los sueldos mínimos de media de cada uno de los roles en España [31]. Así pues, en la Tabla 3.1 se muestra el desglose de estos gastos:

Recurso	Horas	Coste	Total
Analista	108	11.50€/h	1242€
Diseñador	34	11.50€/h	391€
Programador	160	9.00€/h	1440€
Total			**3073€**

Tabla 3.1: Coste de recursos humanos (sin impuestos).

El coste total de la Tabla 3.1 se correspondería con el sueldo neto, por lo que faltaría añadirle los impuestos. Los impuestos estimados serían del 20 %. Por lo que el coste de recursos humanos pasaría a ser **3687.60€**

En segundo lugar, el coste de los recursos hardware para este proyecto esta compuesto por

el coste prorrateado del ordenador [23] y el *smartphone* [25] de desarrollo, donde supondremos que el ordenador y el *smartphone* podrán emplearse durante 5 y 3 años respectivamente antes de que tengan que ser reemplazados, y las balizas BLE [1]. En la Tabla 3.2 puede verse el coste total de los recursos hardware:

Recurso	**Cantidad**	**Coste**	**Total**
ASUS X556UV	4 (meses)	850€	56.66€
Samsung Galaxy A5 2017 SM-A520F	4 (meses)	350€	38.88€
Accent Systems iBKS 105	50	18€	900€
Total			**995.54€**

Tabla 3.2: Coste de recursos hardware.

En tercer lugar, el coste de los recursos software para el desarrollo del proyecto está compuesto por los productos de software empleados durante el mismo. Estos son WebStorm [32], IDE (*Integrated Development Environment*) para el lenguaje JavaScript, MagicDraw [18], software para el desarrollo de diagramas (diagrama de casos de uso, de clases...), y ArcGIS Pro, software para el tratado de los mapas [6]. En el caso de MagicDraw, el coste de la licencia es único, por lo que se prorratea el coste suponiendo que se pueda emplear para 20 proyectos. En la Tabla 3.3 se muestra el coste total de los recursos software:

Recurso	**Uso (meses)**	**Coste**	**Total**
WebStorm	1	129€/año	10.75€
MagicDraw (pago único)	-	800€	40€
ArcGIS Pro	0.5	2700€/año	112.50€
Total			**163.25€**

Tabla 3.3: Coste de recursos software.

Por último, los costes indirectos son aquellos costes causados por el propio desarrollo del proyecto. Este tipo de costes están relacionados con el lugar donde se produce el desarrollo y suelen ser el coste de la electricidad, luz, conexión a Internet, servicios de limpieza... Por lo general estos costes se presupuestan como un 20 % del coste de recursos humanos, es decir, el 20 % de 3687.60, **737.52€**.

Como resumen, el coste estimado del proyecto puede verse en la Tabla 3.4, donde el coste total asciende a **5583.91€**:

Concepto	**Coste**
Recursos humanos	3687.60€
Recursos hardware	995.54€
Recursos software	163.25€
Costes indirectos	737.52€
Total	**5583.91€**

Tabla 3.4: Estimación de costes del proyecto.

ID	Riesgo	Tipo	Ámbito
RG1	Cambio en los requisitos	General	Producto
RG2	Falta de comunicación con interesados	General	Proyecto
RG3	Mala gestión del tiempo	General	Proyecto
RE1	Falta de conocimiento con JavaScript	Específico	Proyecto
RE2	Tamaño de la aplicación	Específico	Proyecto

Tabla 3.5: Riesgos identificados.

3.4. Análisis de riesgos

Los riesgos que pueden presentarse durante el desarrollo pueden ser clasificados en función de si afectan al proyecto o al producto. Además, estos pueden ser riesgos generales, los cuales afectan a cualquier tipo de proyecto o producto; o específicos, que afectan este proyecto o producto en particular. Así pues, los riesgos identificados, su análisis y los planes de prevención y contingencia pueden verse en las Tablas 3.5, 3.6 y 3.7 respectivamente.

3.5. Seguimiento del proyecto

El seguimiento de este proyecto se ha realizado empleando Microsoft Project y el diagrama de Gantt de la planificación inicial (Figura 3.1). De forma semanal, se ha revisado el diagrama de Gantt para saber que tareas debían realizarse y en cuanto tiempo, y posteriormente, modificando el tiempo empleado en las tareas en caso de haber necesitado más o menos tiempo respecto al planificado.

La incidencia más importante con diferencia ha sido el desarrollo del Módulo 3 antes del Módulo 2. Esto fue debido a que al finalizar el Módulo 1 se mantuvo una reunión con los interesados de la aplicación, y los clientes manifestaron la necesidad de tener una versión operativa para presentarla a mediados de abril. Ante la imposibilidad de tenerla finalizada para dicha fecha, se decidió tener lista una versión de la aplicación con toda la funcionalidad de búsquedas y mejoras de usabilidad (Módulo 3). Este descuadre respecto a la planificación inicial puede verse claramente en el diagrama de Gantt de la Figura 3.3. Además de ésta, otras incidencias menos importantes fueron:

- Inclusión de una nueva tarea en la fase de implementación del Módulo 1 llamada *Configuración y familiarizarse con el código*. Esta tarea proviene de la necesidad de tener que configurar el entorno de desarrollo y de revisar el código existente para identificar las zonas que se deben tratar. Esta tarea no afectó a la planificación, ya que el tiempo que se empleó provenía de finalizar antes de tiempo otras tareas del Módulo 1.
- Durante el desarrollo del Módulo 3 apareció un nuevo requisito por el cual la aplicación debía permitir buscar las cabinas de estudio de la biblioteca. Así pues, se añadió una nueva tarea que descuadró un poco la planificación.
- Por falta de tiempo, a las tareas de pruebas del Módulo 2 (último módulo en realizarse) se

ID	Análisis
RG1	▪ Magnitud: Variable según la fase de aparición, mayor cuanto más avanzado este el proyecto. ▪ Descripción: Los requisitos representan la funcionalidad que deberá tener la aplicación. ▪ Impacto: Un cambio en los requisitos puede afectar a varias fases del proyecto, incluso a aquellas que ya se daban por finalizadas, lo cual puede afectar gravemente a la planificación.
RG2	▪ Magnitud: Variable según la necesidad de la comunicación. ▪ Descripción: La comunicación entre desarrolladores y cliente debe ser fluida para tratar correctamente todos los detalles de la aplicación. ▪ Impacto: Una mala comunicación puede producir malentendidos, lo que puede generar en el desarrollo de una funcionalidad no deseada.
RG3	▪ Magnitud: Baja, si se realiza un correcto seguimiento de la planificación, tomando las medidas oportunas. ▪ Descripción: Una mala planificación causada por la inexperiencia. ▪ Impacto: La mala gestión del tiempo puede provocar que el proyecto no finalice a tiempo.
RE1	▪ Magnitud: Media. ▪ Descripción: Poco o inexistente conocimiento de uno de los lenguajes de programación empleados en el proyecto. ▪ Impacto: Dependiendo de la velocidad de aprendizaje, pueden llegar a producirse retrasos importantes.
RE2	▪ Magnitud: Media-Alta. ▪ Descripción: En todo proyecto donde exista un producto inicial el desarrollador debe familiarizarse con este. ▪ Impacto: Para una aplicación de grandes dimensiones el tiempo de familiarización puede llegar a ser muy elevado. El desarrollador puede verse incluso sobrepasado. Esto puede producir grandes retrasos.

Tabla 3.6: Análisis de riesgos.

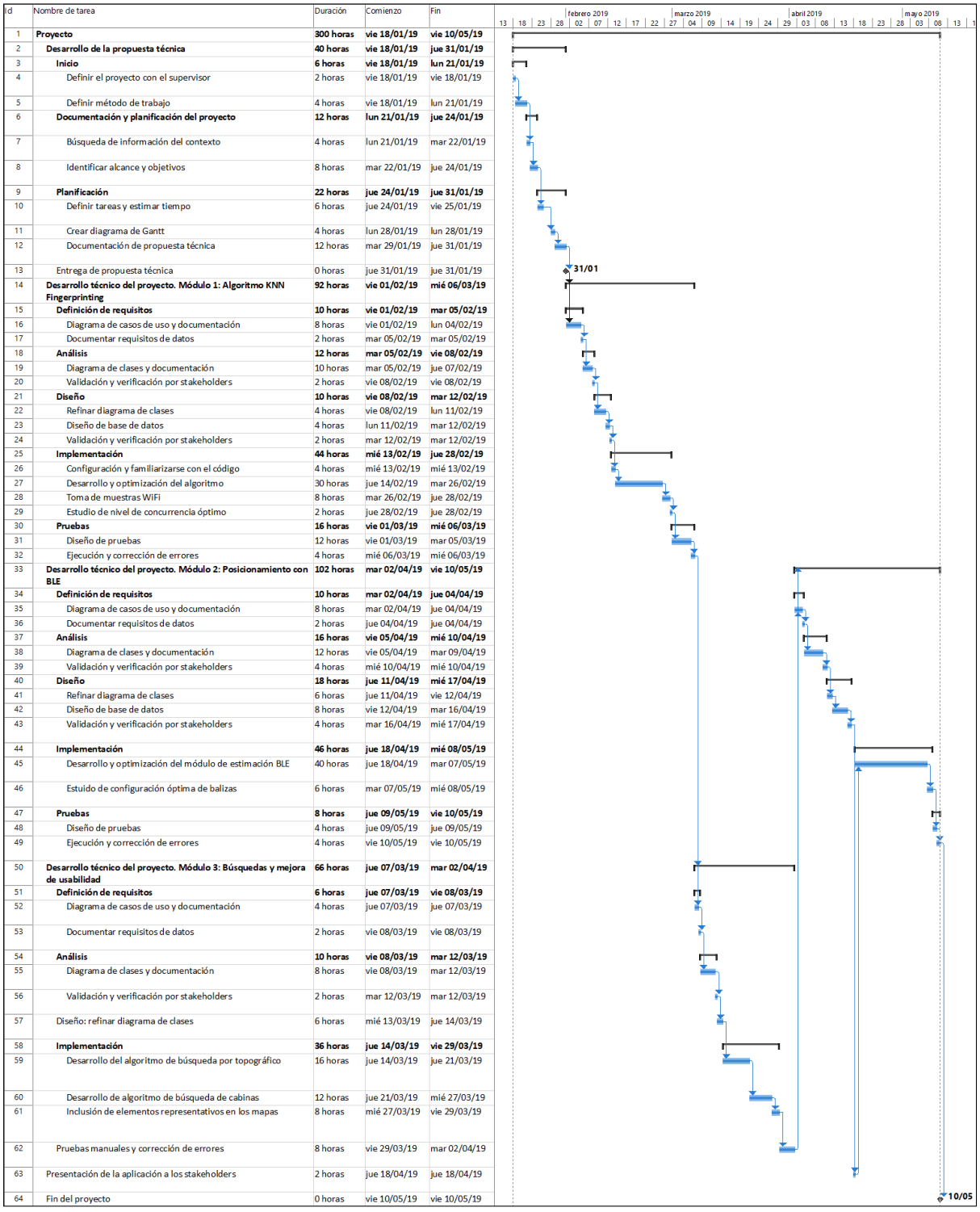
ID	Plan de prevención	Plan de contingencia
RG1	No se puede controlar completamente que no hayan cambios en los requisitos, pero al menos se debe asegurar que los requisitos definidos inicialmente son correctos para evitar cambios producidos por nuestra mala comprensión. También se deberán realizar reuniones periódicas para validar la consecución de los requisitos.	Se buscará reducir tiempo de algunas tareas no esenciales, como el diseño concienzudo de las pruebas unitarias/integración.
RG2	Mantener una buena relación con los interesados, mostrando desde el primer momento una alta motivación e interés, dejando claro la necesidad de una buena comunicación.	Concertar una reunión para tratar de solucionar los posibles problemas comunicativos, dejando clara la importancia de la comunicación entre cliente y desarrolladores en un proyecto.
RG3	Monitorización continua del desarrollo del proyecto y adaptación de la planificación extrapolando a módulos posteriores la experiencia adquirida en los iniciales.	Al producirse un gran desfase en la planificación, se buscará reducir el tiempo empleado a las tareas menos importantes del desarrollo.
RE1	Realización de cursos sobre el lenguaje en los tiempos libres (fuera de la estancia) del desarrollador para familiarizarse con el mismo.	Solicitar ayuda a miembros de la organización expertos en el lenguaje.
RE2	Comenzar a familiarizarse con la aplicación en los tiempos libres (fuera de la estancia) del desarrollador.	Solicitar ayuda y guía a miembros de la organización que hayan tratado con la aplicación.

Tabla 3.7: Medidas de prevención y contingencia de riesgos.

les ha dedicado mucho menos tiempo del planificado, especialmente en la tarea de diseño de pruebas.

- Inicialmente, se tomaron como fiestas la semana de la Magdalena (26-29 de marzo) y Pascua (23-26 de abril). Finalmente estas fiestas se suprimieron, por lo que el proyecto finalizó 8 días laborables antes de lo planificado.

En las Figuras 3.4, 3.5 y 3.6 se muestran los diagramas de Gantt de seguimiento de cada uno de los módulos desarrollados, donde se aprecia mejor las desviaciones respecto a la planificación.



Id	Nombre de tarea	Duración	Comienzo	Fin
1	**Proyecto**	**300 horas**	**vie 18/01/19**	**vie 10/05/19**
2	**Desarrollo de la propuesta técnica**	**40 horas**	**vie 18/01/19**	**jue 31/01/19**
3	**Inicio**	**6 horas**	**vie 18/01/19**	**lun 21/01/19**
4	Definir el proyecto con el supervisor	2 horas	vie 18/01/19	vie 18/01/19
5	Definir método de trabajo	4 horas	vie 18/01/19	lun 21/01/19
6	**Documentación y planificación del proyecto**	**12 horas**	**lun 21/01/19**	**jue 24/01/19**
7	Búsqueda de información del contexto	4 horas	lun 21/01/19	mar 22/01/19
8	Identificar alcance y objetivos	8 horas	mar 22/01/19	jue 24/01/19
9	**Planificación**	**22 horas**	**jue 24/01/19**	**jue 31/01/19**
10	Definir tareas y estimar tiempo	6 horas	jue 24/01/19	vie 25/01/19
11	Crear diagrama de Gantt	4 horas	lun 28/01/19	lun 28/01/19
12	Documentación de propuesta técnica	12 horas	mar 29/01/19	jue 31/01/19
13	Entrega de propuesta técnica	0 horas	jue 31/01/19	jue 31/01/19
14	**Desarrollo técnico del proyecto. Módulo 1: Algoritmo KNN Fingerprinting**	**92 horas**	**vie 01/02/19**	**mié 06/03/19**
15	**Definición de requisitos**	**10 horas**	**vie 01/02/19**	**mar 05/02/19**
16	Diagrama de casos de uso y documentación	8 horas	vie 01/02/19	lun 04/02/19
17	Documentar requisitos de datos	2 horas	mar 05/02/19	mar 05/02/19
18	**Análisis**	**12 horas**	**mar 05/02/19**	**vie 08/02/19**
19	Diagrama de clases y documentación	10 horas	mar 05/02/19	jue 07/02/19
20	Validación y verificación por stakeholders	2 horas	vie 08/02/19	vie 08/02/19
21	**Diseño**	**10 horas**	**vie 08/02/19**	**mar 12/02/19**
22	Refinar diagrama de clases	4 horas	vie 08/02/19	lun 11/02/19
23	Diseño de base de datos	4 horas	lun 11/02/19	mar 12/02/19
24	Validación y verificación por stakeholders	2 horas	mar 12/02/19	mar 12/02/19
25	**Implementación**	**44 horas**	**mié 13/02/19**	**jue 28/02/19**
26	Configuración y familiarizarse con el código	4 horas	mié 13/02/19	mié 13/02/19
27	Desarrollo y optimización del algoritmo	30 horas	jue 14/02/19	mar 26/02/19
28	Toma de muestras WiFi	8 horas	mar 26/02/19	jue 28/02/19
29	Estudio de nivel de concurrencia óptimo	2 horas	jue 28/02/19	jue 28/02/19
30	**Pruebas**	**16 horas**	**vie 01/03/19**	**mié 06/03/19**
31	Diseño de pruebas	12 horas	vie 01/03/19	mar 05/03/19
32	Ejecución y corrección de errores	4 horas	mié 06/03/19	mié 06/03/19
33	**Desarrollo técnico del proyecto. Módulo 2: Posicionamiento con BLE**	**102 horas**	**mar 02/04/19**	**vie 10/05/19**
34	**Definición de requisitos**	**10 horas**	**mar 02/04/19**	**jue 04/04/19**
35	Diagrama de casos de uso y documentación	8 horas	mar 02/04/19	jue 04/04/19
36	Documentar requisitos de datos	2 horas	jue 04/04/19	jue 04/04/19
37	**Análisis**	**16 horas**	**vie 05/04/19**	**mié 10/04/19**
38	Diagrama de clases y documentación	12 horas	vie 05/04/19	mar 09/04/19
39	Validación y verificación por stakeholders	4 horas	mié 10/04/19	mié 10/04/19
40	**Diseño**	**18 horas**	**jue 11/04/19**	**mié 17/04/19**
41	Refinar diagrama de clases	6 horas	jue 11/04/19	vie 12/04/19
42	Diseño de base de datos	8 horas	vie 12/04/19	mar 16/04/19
43	Validación y verificación por stakeholders	4 horas	mar 16/04/19	mié 17/04/19
44	**Implementación**	**46 horas**	**jue 18/04/19**	**mié 08/05/19**
45	Desarrollo y optimización del módulo de estimación BLE	40 horas	jue 18/04/19	mar 07/05/19
46	Estuido de configuración óptima de balizas	6 horas	mar 07/05/19	mié 08/05/19
47	**Pruebas**	**8 horas**	**jue 09/05/19**	**vie 10/05/19**
48	Diseño de pruebas	4 horas	jue 09/05/19	jue 09/05/19
49	Ejecución y corrección de errores	4 horas	vie 10/05/19	vie 10/05/19
50	**Desarrollo técnico del proyecto. Módulo 3: Búsquedas y mejora de usabilidad**	**66 horas**	**jue 07/03/19**	**mar 02/04/19**
51	**Definición de requisitos**	**6 horas**	**jue 07/03/19**	**vie 08/03/19**
52	Diagrama de casos de uso y documentación	4 horas	jue 07/03/19	jue 07/03/19
53	Documentar requisitos de datos	2 horas	vie 08/03/19	vie 08/03/19
54	**Análisis**	**10 horas**	**vie 08/03/19**	**mar 12/03/19**
55	Diagrama de clases y documentación	8 horas	vie 08/03/19	mar 12/03/19
56	Validación y verificación por stakeholders	2 horas	mar 12/03/19	mar 12/03/19
57	Diseño: refinar diagrama de clases	6 horas	mié 13/03/19	jue 14/03/19
58	**Implementación**	**36 horas**	**jue 14/03/19**	**vie 29/03/19**
59	Desarrollo del algoritmo de búsqueda por topográfico	16 horas	jue 14/03/19	jue 21/03/19
60	Desarrollo de algoritmo de búsqueda de cabinas	12 horas	jue 21/03/19	mié 27/03/19
61	Inclusión de elementos representativos en los mapas	8 horas	mié 27/03/19	vie 29/03/19
62	Pruebas manuales y corrección de errores	8 horas	vie 29/03/19	mar 02/04/19
63	Presentación de la aplicación a los stakeholders	2 horas	jue 18/04/19	jue 18/04/19
64	Fin del proyecto	0 horas	vie 10/05/19	vie 10/05/19

Figura 3.3: Diagrama de Gantt final.

Desarrollo técnico del proyecto. Módulo 1: Algoritmo KNN Fingerprinting	92 horas
Definición de requisitos	**10 horas**
Diagrama de casos de uso y documentación	8 horas
Documentar requisitos de datos	2 horas
Análisis	**12 horas**
Diagrama de clases y documentación	10 horas
Validación y verificación por stakeholders	2 horas
Diseño	**10 horas**
Refinar diagrama de clases	4 horas
Diseño de base de datos	4 horas
Validación y verificación por stakeholders	2 horas
Implementación	**44 horas**
Configuración y familiarizarse con el código	4 horas
Desarrollo y optimización del algoritmo	30 horas
Toma de muestras WiFi	8 horas
Estudio de nivel de concurrencia óptimo	2 horas
Pruebas	**16 horas**
Diseño de pruebas	12 horas
Ejecución y corrección de errores	4 horas

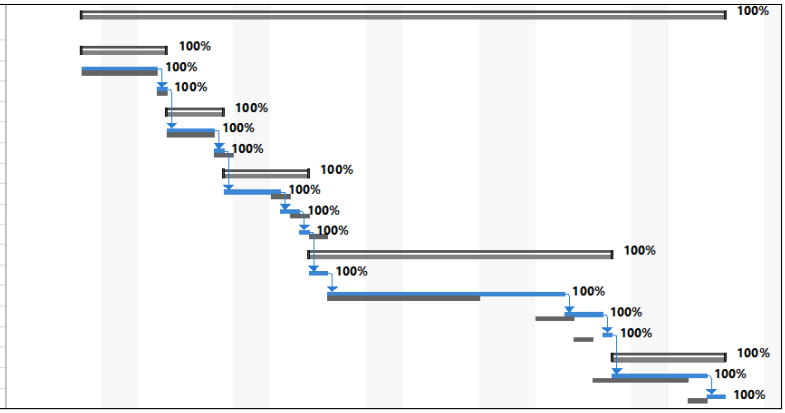


Figura 3.4: Diagrama de Gantt de seguimiento del Módulo 1.

Desarrollo técnico del proyecto. Módulo 2: Posicionamiento con BLE	102 horas
Definición de requisitos	**10 horas**
Diagrama de casos de uso y documentación	8 horas
Documentar requisitos de datos	2 horas
Análisis	**16 horas**
Diagrama de clases y documentación	12 horas
Validación y verificación por stakeholders	4 horas
Diseño	**18 horas**
Refinar diagrama de clases	6 horas
Diseño de base de datos	8 horas
Validación y verificación por stakeholders	4 horas
Implementación	**46 horas**
Desarrollo y optimización del módulo de estimación BL	40 horas
Estuido de configuración óptima de balizas	6 horas
Pruebas	**8 horas**
Diseño de pruebas	4 horas
Ejecución y corrección de errores	4 horas
Presentación de la aplicación a los stakeholders	4 horas

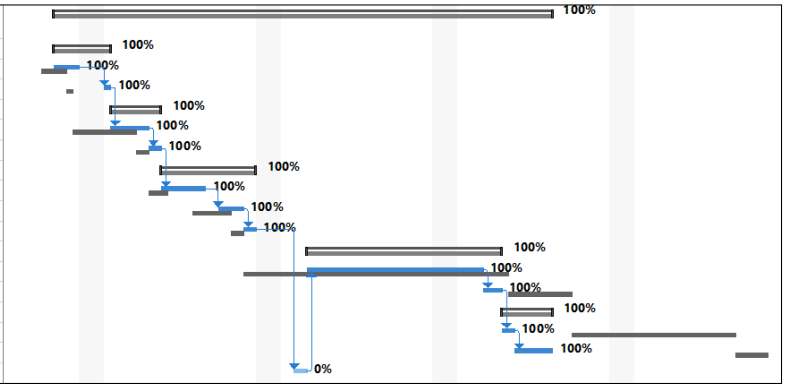


Figura 3.5: Diagrama de Gantt de seguimiento del Módulo 2.

Desarrollo técnico del proyecto. Módulo 3: Búsquedas y mejoras de usabilidad	66 horas
Definición de requisitos	**6 horas**
Diagrama de casos de uso y documentación	4 horas
Documentar requisitos de datos	2 horas
Análisis	**10 horas**
Diagrama de clases y documentación	8 horas
Validación y verificación por stakeholders	2 horas
Diseño: refinar diagrama de clases	6 horas
Implementación	**36 horas**
Desarrollo del algoritmo de búsqueda por topográfico	16 horas
Desarrollo de algoritmo de búsqueda de cabinas	12 horas
Inclusión de elementos representativos en los mapas	8 horas
Pruebas manuales y corrección de errores	8 horas
Fin del proyecto	0 horas

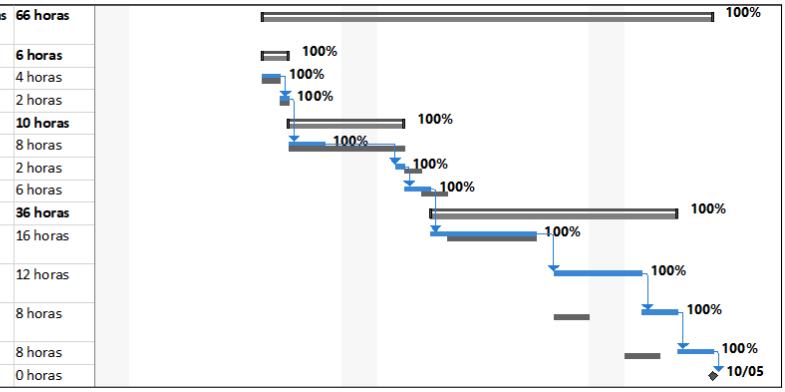


Figura 3.6: Diagrama de Gantt de seguimiento del Módulo 3.

Capítulo 4

Requisitos y análisis del sistema

4.1. Módulo 1: Algoritmo KNN Fingerprinting

En esta sección se muestran los requisitos, el diagrama de casos de uso y su documentación, y el análisis junto el diagrama de clases y su documentación, correspondiente al primer módulo del desarrollo.

El principal requisito de este módulo es el siguiente:

- El sistema debe poder estimar la posición del usuario mediante las señales Wi-Fi que recibe su dispositivo móvil. La estimación debe realizarse en el propio dispositivo del usuario.

Como se ha visto en el Capítulo 2, la actual aplicación es capaz de realizar las estimaciones, pero para ello hace uso de un servidor donde se realizan los cálculos. Así pues, este requisito implica la implementación del algoritmo de estimaciones en Java y el traslado de la base de datos de muestras al dispositivo.

4.1.1. Diagrama de casos de uso y documentación

A continuación se muestran dos diagramas de casos de uso para esta primera fase del proyecto. Estos diagramas se corresponderán con la funcionalidad del sistema inicial y la funcionalidad a desarrollar en el módulo 1, con el objetivo de contrastar ambas versiones.

En la Figura 4.1 se encuentra el diagrama de casos de uso correspondiente a la funcionalidad inicial. Como puede verse hay identificados los siguientes actores:

- Usuario: cualquier persona que hace uso de la aplicación, como estudiantes de la universidad, bibliotecarios, profesores...

- Servidor de estimaciones: corresponde con el servidor al que se le envían las señales Wi-Fi recolectadas y que proporciona al usuario una estimación de su posición mediante esas señales.

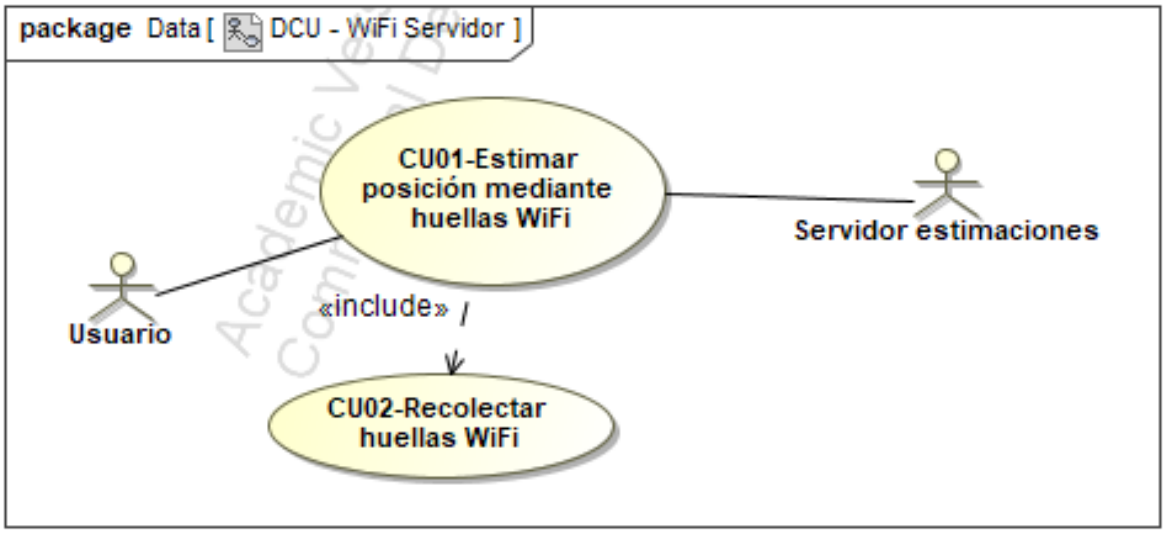


Figura 4.1: Módulo 1 - Diagrama de casos de uso con la funcionalidad existente.

También se pueden ver los casos de uso identificados. Aunque se podría haber identificado un solo caso de uso, se ha decidido identificar dos debido a la complejidad de la funcionalidad. Estos casos de uso son:

- Estimar posición mediante huellas Wi-Fi: caso de uso correspondiente a la estimación de la posición del usuario. Este caso de uso necesita de la existencia del siguiente.
- Recolectar huellas Wi-Fi: caso de uso correspondiente a la obtención de señales Wi-Fi con las que obtener la estimación.

En la Figura 4.2 puede verse el diagrama de casos de uso correspondiente a la funcionalidad que se va a implementar. La única diferencia con el diagrama de la Figura 4.1 es la eliminación del servidor.

En la Tablas 4.1 y 4.2 se encuentran la especificaciones de los casos de uso *Estimar posición mediante huellas Wi-Fi* y *Recolectar huellas Wi-Fi* respectivamente.

Finalmente, se definen los requisitos de datos para el desarrollo de la funcionalidad. Los requisitos de datos identificados son los siguientes:

- Localización: información correspondiente al lugar donde se tomen muestras para compararlas con las recogidas por el usuario a la hora de realizar la estimación.
- Antena: información correspondiente a las antenas Wi-Fi que pueden encontrarse en la biblioteca.

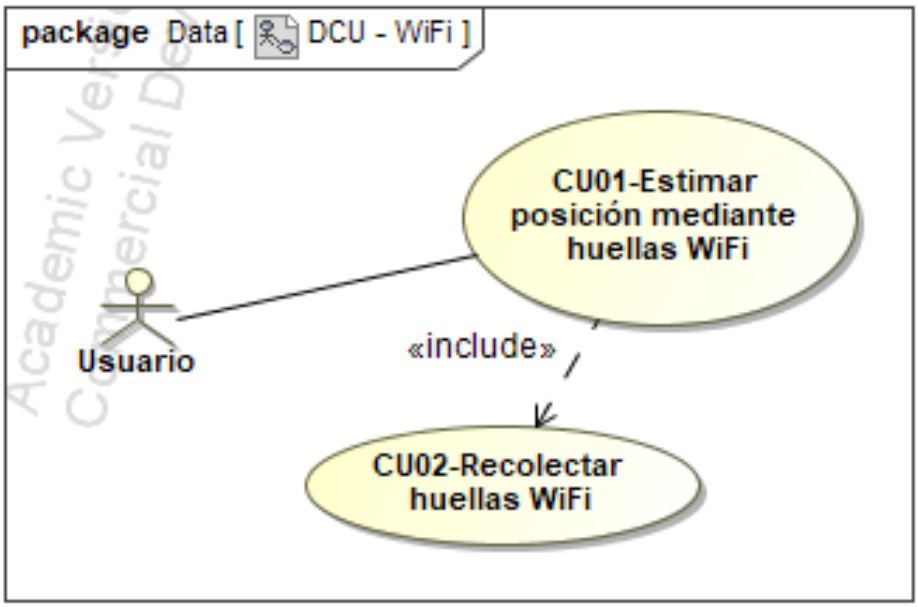


Figura 4.2: Módulo 1 - Diagrama de casos de uso de la funcionalidad a implementar.

ID: CU01 **Nombre:** Estimar posición mediante huellas Wi-Fi **Versión: 1.0** **Fuente:** Ubik
Descripción: este caso de uso representa la funcionalidad que permite estimar la posición del usuario mediante las señales Wi-Fi que recoge su dispositivo móvil.
Flujo de eventos básico: 1. El usuario solicita la estimación de su posición. 2. El módulo Wi-Fi obtiene la señales Wi-Fi que recibe el dispositivo. (CU02) 3. El módulo Wi-Fi calcula una posición estimada. 4. El usuario recibe la posición estimada. **Flujo de eventos alternativo:** **Precondiciones:** el usuario debe encontrarse dentro de la biblioteca.

Tabla 4.1: Módulo 1 - Especificación CU01-Estimar posición mediante huellas Wi-Fi.

ID: CU02 **Nombre:** Recolectar huellas Wi-Fi **Versión:** 1.0 **Fuente:** Ubik
Descripción: este caso de uso representa la funcionalidad que permite la recogida de las señales Wi-Fi que captura el dispositivo móvil del usuario.
Flujo de eventos básico: 1. El módulo Wi-Fi solicita las señales Wi-Fi al sistema Android 2. El sistema Android proporciona las señales al módulo Wi-Fi. 3. El módulo Wi-Fi transforma el conjunto de señales en un *fingerprint*. **Flujo de eventos alternativo:** **Precondiciones:** la antena Wi-Fi del dispositivo debe estar habilitada.

Tabla 4.2: Módulo 1 - Especificación CU02-Recolectar huellas Wi-Fi.

- Huella: información correspondiente a las señales Wi-Fi recogidas por el dispositivo del usuario.
- Estimación: información correspondiente al lugar donde probablemente se encuentra el usuario.

En las Tablas 4.3, 4.4, 4.5 y 4.6 se pueden encontrar las especificaciones de estos requisitos de datos.

ID: RD01 **Nombre:** Localización **Versión:** 1.0 **Fuente:** Ubik
Tipo: Registro en base de datos. **Datos específicos:** Latitud, longitud, planta, y listado de intensidades.
Latitud: latitud geográfica. **Longitud:** longitud geográfica. **Planta:** indicador de la planta de la biblioteca. **Listado de intensidades:** intensidades de las señales Wi-Fi recogidas en la localización.

Tabla 4.3: Módulo 1 - Requisito de datos - Localización.

4.1.2. Diagrama de clases y documentación

Teniendo en cuenta los casos de uso y los requisitos de datos especificados anteriormente, se han identificado las siguientes clases con sus respectivos atributos:

- Localización: longitud, latitud y planta.
- Señal: intensidad y canal.

ID: RD02
Nombre: Antena
Versión: 1.0
Fuente: Ubik
Tipo: Registro en base de datos.
Datos específicos: SSID (*Service Set Identifier*) y MAC (*Media Access Control*).
SSID: nombre de la red.
MAC: dirección física de la antena.

Tabla 4.4: Módulo 1 - Requisito de datos - Antena.

ID: RD03
Nombre: Huella
Versión: 1.0
Fuente: Ubik
Tipo: Entrada.
Datos específicos: listado de tuplas (intensidad, MAC)
Intensidad: intensidad de una señal Wi-Fi en decibelios.
MAC: dirección física de la antena a la que pertenece la señal.

Tabla 4.5: Módulo 1 - Requisito de datos - Huella.

ID: RD04
Nombre: Estimación
Versión: 1.0
Fuente: Ubik
Tipo: Salida
Datos específicos: Latitud, longitud y planta.
Latitud: latitud geográfica.
Longitud: longitud geográfica.
Planta: indicador de la planta de la biblioteca.

Tabla 4.6: Módulo 1 - Requisito de datos - Estimación.

- Antena: ssid y mac.

Junto con las clases también se han identificado las siguientes relaciones entre ellas:

- Tiene: asociación entre Localización y Señal.
- Corresponde: asociación entre Señal y Antena.

Cabe destacar que no se han identificado operaciones en las clases, ya que el único objetivo de éstas es el de almacenar información que será empleada para las estimaciones. Se podrían definir las operaciones básicas como constructores, *getters* o *setters*, pero hacerlo no aportaría ningún valor añadido al diagrama.

En la Figura 4.3 puede verse el diagrama de clases donde se representa toda esta información. Además, en las Tablas 4.7, 4.8 y 4.9 se muestra la especificación de cada una de las clases y sus correspondientes atributos.

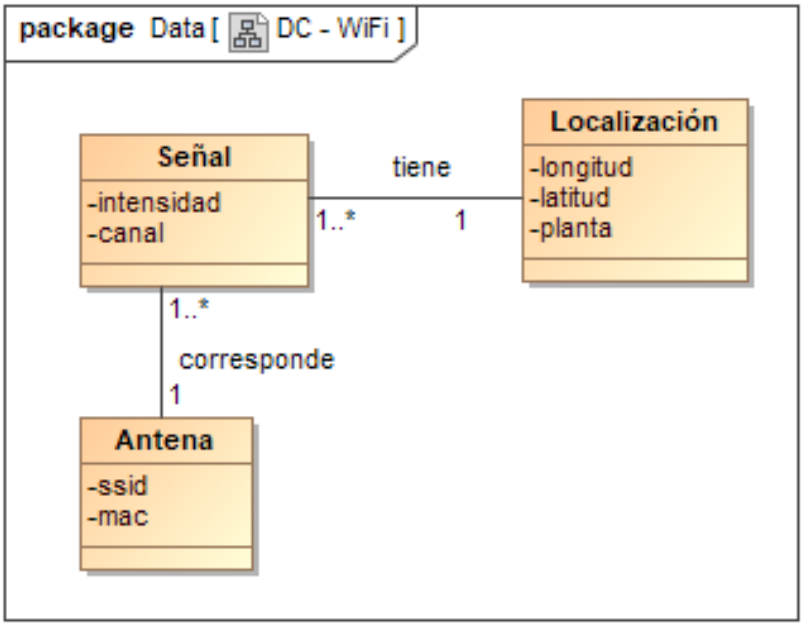


Figura 4.3: Módulo 1 - Diagrama de clases.

4.2. Módulo 2: Posicionamiento con BLE

En la siguiente sección se muestra el requisito, el diagrama de casos de uso y el diagrama de clases del análisis junto con sus respectivas documentaciones, correspondientes al segundo módulo del desarrollo.

Este segundo módulo, al igual que el primero, está compuesto por un solo requisito funcional:

Nombre: Localización **Versión: 1.0** **Fuente: Ubik**
Descripción: Los objetos de esta clase representan las localizaciones de la biblioteca donde se han tomado muestras de las señales visibles en ellas.
Atributos: -longitud: longitud geográfica del lugar. -latitud: latitud geográfica del lugar. -planta: planta del edificio.
Asociaciones: -tiene: con Señal $\rightarrow$ 1..* Desde una localización se ven una o muchas señales con cierta intensidad.

Tabla 4.7: Módulo 1 - Especificación de la clase Localización.

Nombre: Señal **Versión: 1.0** **Fuente: Ubik**
Descripción: Los objetos de esta clase representan la intensidad de la señal de una determinada antena que ha sido vista desde una determinada posición.
Atributos: -intensidad: valor en decibelios que indica la fuerza de la señal. -canal: frecuencia de emisión de la señal (2.4 o 5 GHz).
Asociaciones: -tiene: con Localización $\rightarrow$ 1 Una señal con cierta intensidad ha sido vista desde una localización. -corresponde: con Antena $\rightarrow$ 1 Una señal corresponde solo a una antena.

Tabla 4.8: Módulo 1 - Especificación de la clase Señal.

Nombre: Antena **Versión: 1.0** **Fuente: Ubik**
Descripción: Los objetos de esta clase representan a las antenas de la biblioteca de las cuales existen señales.
Atributos: -ssid: nombre identificador de la antena. -mac: dirección física de la antena
Asociaciones: -corresponde: con Señal $\rightarrow$ 1..* De una antena han sido vistas una o muchas señales.

Tabla 4.9: Módulo 1 - Especificación de la clase Antena.

- El sistema debe poder estimar la posición del usuario mediante las señales BLE que recibe su dispositivo móvil. La estimación debe realizarse en el propio dispositivo del usuario.

Este requisito implica la implementación del código necesario para realizar la captura de las señales BLE y el algoritmo de estimaciones. También incluye el tratamiento de la información sobre las balizas (identificación, latitud y longitud).

Además, también existe un requisito de calidad, orientado a la optimización de la configuración de las balizas, manteniendo un compromiso entre la frecuencia de emisión y potencia de las señales y la duración de la batería.

4.2.1. Diagrama de casos de uso y documentación

A continuación, en la Figura 4.4 se muestra el diagrama de casos de uso de este segundo módulo del proyecto. Como se puede observar se identifican los siguientes elementos:

- Actores:
  - Usuario: cualquier persona que hace uso de la aplicación, como estudiantes de la universidad, bibliotecarios, profesores...
- Casos de uso:
  - Estimar posición mediante señales BLE: se corresponde con la estimación de la posición del usuario dadas unas determinadas señales BLE. Este caso de uso requiere la existencia del siguiente.
  - Capturar señales BLE: se corresponde con la obtención de las señales BLE presentes en el entorno donde está el dispositivo, las cuales servirán para estimar la posición del usuario.

En las siguientes Tablas (4.10 y 4.11) puede verse la especificación de ambos casos de uso identificados, *Estimar posición mediante señales BLE* y *Capturar señales BLE*.

Finalmente, se identifican varios requisitos de datos para el desarrollo de la funcionalidad. Estos requisitos de datos son muy parecidos a los identificados en el anterior módulo debido a la similitud entre ambas funcionalidades. Estos requisitos pueden verse a continuación, junto con sus especificaciones en las Tablas 4.12, 4.13, 4.14:

- Baliza: información correspondiente a las balizas BLE que desplegadas en la biblioteca.
- Señal: información correspondiente a las señales BLE recogidas por el dispositivo del usuario.
- Estimación: información correspondiente al lugar donde probablemente se encuentra el usuario.

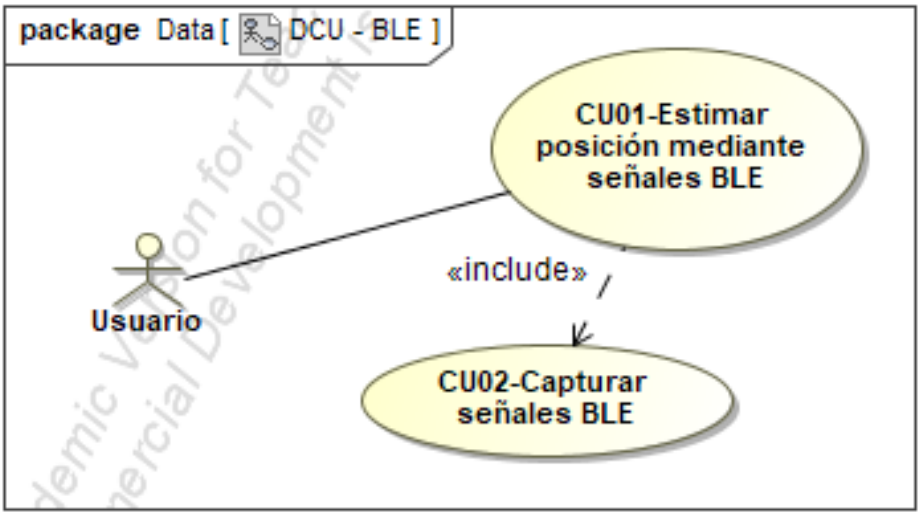


Figura 4.4: Módulo 2 - Diagrama de casos de uso con la funcionalidad a implementar en el módulo.

ID: CU01 **Nombre:** Estimar posición mediante señales BLE **Versión: 1.0** **Fuente:** Ubik
Descripción: este caso de uso representa la funcionalidad que permite estimar la posición del usuario mediante las señales BLE que recoge su dispositivo móvil.
Flujo de eventos básico: 1. El usuario solicita la estimación de su posición. 2. El módulo BLE obtiene la señales BLE que recibe el dispositivo. (CU02) 3. El módulo BLE calcula una posición estimada. 4. El usuario recibe la posición estimada. **Flujo de eventos alternativo:** **Precondiciones:** el usuario debe encontrarse dentro de la biblioteca.

Tabla 4.10: Módulo 2 - Especificación CU01-Estimar posición mediante señales BLE.

ID: CU02 **Nombre:** Captuar señales BLE **Versión: 1.0** **Fuente:** Ubik
Descripción: este caso de uso representa la funcionalidad que permite la recogida de las señales BLE que captura el dispositivo móvil del usuario.
Flujo de eventos básico: 1. El módulo BLE solicita las señales BLE al sistema Android. 2. El sistema Android proporciona las señales al módulo BLE. 3. El módulo BLE transforma las señales recibidas para poder ser tratadas. **Flujo de eventos alternativo:** **Precondiciones:** la antena Bluetooth del dispositivo debe estar habilitada

Tabla 4.11: Módulo 2 - Especificación CU02-Capturar señales BLE.

ID: RD01 **Nombre:** Baliza **Versión:** 1.0 **Fuente:** Ubik
Tipo: Registro en base de datos **Datos específicos:** Identificador, latitud, longitud y planta.
Identificador: identificador de la baliza. **Latitud:** latitud geográfica. **Longitud:** longitud geográfica. **Planta:** indicador de la planta de la biblioteca.

Tabla 4.12: Módulo 2 - Requisito de datos - Baliza.

ID: RD02 **Nombre:** Señal **Versión:** 1.0 **Fuente:** Ubik
Tipo: Entrada **Datos específicos:** Identificador, intensidad.
Identificador: identificador de la baliza. **Intensidad:** intensidad en dBm.

Tabla 4.13: Módulo 2 - Requisito de datos - Señal.

ID: RD03 **Nombre:** Estimación **Versión:** 1.0 **Fuente:** Ubik
Tipo: Salida **Datos específicos:** Latitud, longitud y planta.
Latitud: latitud geográfica. **Longitud:** longitud geográfica. **Planta:** indicador de la planta de la biblioteca.

Tabla 4.14: Módulo 2 - Requisito de datos - Estimación.

4.2.2. Diagrama de clases y documentación

A partir de los casos de uso y los requisitos de datos especificados, se han identificado dos clases de las que se tenga que mantener información. Una de ellas es Baliza, y contiene la información de interés de la baliza, como su identificador y posición. La otra clase es Campaña, que representa un conjunto de balizas desplegadas con una determinada configuración. En la Figura 4.5 puede verse el diagrama de clases, y en las Tablas 4.15 y 4.16 su correspondiente especificación.

Al igual que en el módulo anterior, no se han especificado operaciones en las clases debido a su escaso valor añadido. Por contra, cabe destacar que pese a que la funcionalidad del módulo anterior y este son muy similares (como ya se ha visto con los diagramas de casos de uso, Figuras 4.2 y 4.4) los diagramas de clases no lo son (Figuras 4.3 y 4.5). Esto es debido a que por la naturaleza de las técnicas de posicionamiento, *fingerprinting* y centroide ponderado (Wi-Fi y BLE respectivamente). Por otro lado, para las estimaciones con Wi-Fi se requiere tener una base de datos de entrenamiento; esto no es necesario con BLE, donde solo es necesario conocer las posiciones de las balizas BLE.

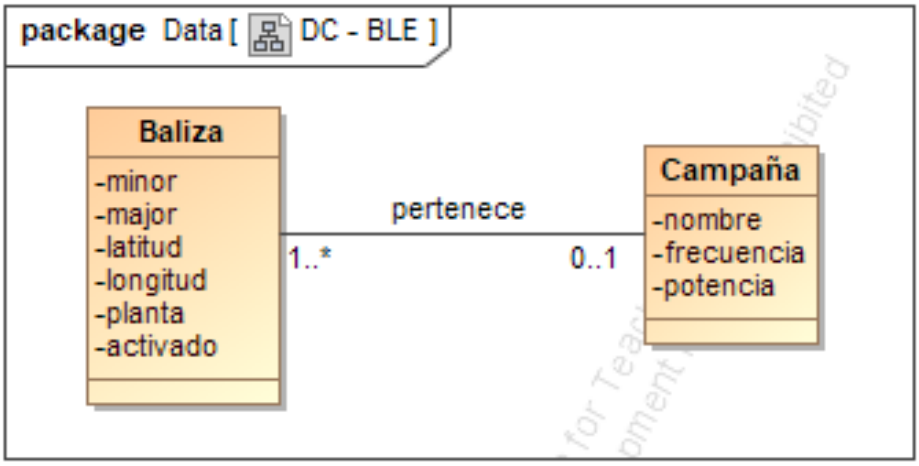


Figura 4.5: Módulo 2 - Diagrama de clases

Nombre: Baliza **Versión: 1.0** **Fuente: Ubik**
Descripción: Los objetos de esta clase representan a las balizas BLE.
Atributos: -minor: identificador de la baliza. -major: identificador de la baliza. -latitud: latitud geográfica donde se encuentra la baliza. -longitud: longitud geográfica donde se encuentra la baliza. -planta: planta del edificio donde se encuentra la antena. -activado: indica si la baliza esta activada
Asociaciones: -pertenece: con Campaña $\rightarrow$ 0..1 Una baliza solo puede pertenecer a una sola campaña

Tabla 4.15: Módulo 2 - Especificación de la clase Baliza.

Nombre: Campaña **Versión: 1.0** **Fuente: Ubik**
Descripción: Los objetos de esta clase representan a un conjunto de balizas desplegadas en la biblioteca.
Atributos: -nombre: nombre identificador de la campaña. -frecuencia: frecuencia de emisión de las balizas en milisegundos. -potencia: potencia de emisión de las balizas en dBm.
Asociaciones: -pertenece: con Baliza $\rightarrow$ 1..* Una campaña esta compuesta por una o varias balizas.

Tabla 4.16: Módulo 2 - Especificación de la clase Campaña.

4.3. Módulo 3: Búsquedas y mejora de usabilidad

Por lo que respecta al final del presente capítulo, en esta sección se muestran los requisitos, diagrama de casos de uso y diagrama de clases junto a sus correspondientes documentaciones, correspondientes al tercer módulo del desarrollo.

Los principales requisitos funcionales de este módulo son:

- El sistema debe permitir al usuario buscar libros mediante sus correspondientes topográficos tanto parciales como completos.
- El sistema debe permitir al usuario buscar las cabinas de la biblioteca mediante sus números identificadores.

Dichos requisitos suponen la implementación de dos servicios de búsqueda en la parte del servidor en Node.js, además del correspondiente uso de los servicios desde la aplicación Android.

Además, también existe un requisito no funcional, orientado a la mejora de la usabilidad con la inclusión de elementos representativos en los mapas y la mejora de algunas partes de la interfaz.

4.3.1. Diagrama de casos de uso y documentación

Seguidamente se muestra el diagrama de casos de uso de este último módulo del proyecto en la Figura 4.6. Como puede observarse, se identifican los siguientes actores:

- Usuario: cualquier persona que hace uso de la aplicación, como estudiantes de la universidad, bibliotecarios, profesores...
- Catálogo biblioteca: este actor representa al catálogo de la Universitat Jaume I [9], que es quien contiene la información sobre los libros y las cabinas.

También pueden verse los casos de uso identificados, en este caso, un caso de uso independiente para cada tipo de búsqueda debido a que representan funcionalidades distintas.

En las Tablas 4.17 y 4.18 se encuentran las especificaciones de los casos de uso identificados, *Buscar libros por topográfico* y *Buscar cabinas*.

Finalmente, los requisitos de datos identificados para el desarrollo de la funcionalidad son los siguientes, y sus especificaciones pueden encontrarse en las Tablas 4.19, 4.20, 4.21, 4.22:

- Localización: información correspondiente al lugar donde se encuentra el libro o cabina buscado.

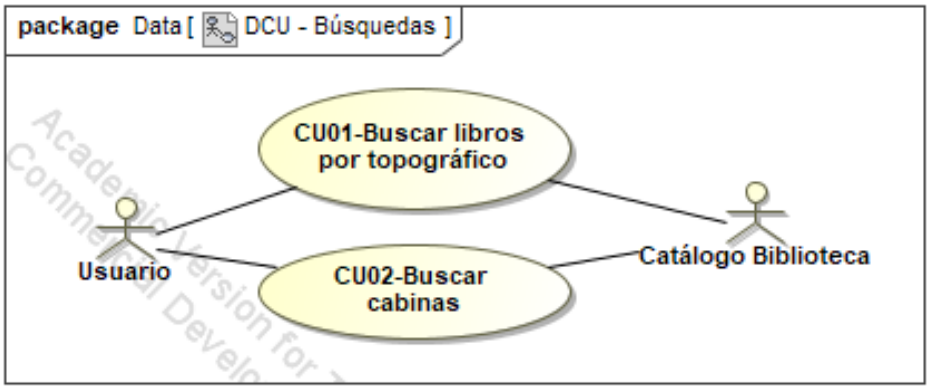


Figura 4.6: Módulo 3 - Diagrama de casos de uso con la funcionalidad a implementar en el módulo.

ID: CU01 **Nombre:** Buscar libros por topográfico **Versión: 1.0** **Fuente:** Ubik
Descripción: este caso de uso representa la funcionalidad que permite la búsqueda de libros dado sus topográficos parciales o completos.
Flujo de eventos básico: 1. El usuario indica el topográfico de un libro que desea encontrar. 2. El dispositivo realiza una petición REST al servidor con el topográfico indicado. 3. El servidor obtiene los datos del catálogo, los trata y los devuelve al dispositivo. 4. El usuario recibe en el dispositivo un listado de resultados que se ajustan a su búsqueda. **Flujo de eventos alternativo:** **Precondiciones:** el usuario debe tener conexión a Internet.

Tabla 4.17: Módulo 3 - Especificación CU01-Buscar libros por topográfico.

ID: CU02 **Nombre:** Buscar cabinas **Versión: 1.0** **Fuente:** Ubik
Descripción: este caso de uso representa la funcionalidad que permite la búsqueda de cabinas de la biblioteca dado su número identificador.
Flujo de eventos básico: 1. El usuario indica el número de la cabina que desea encontrar/consultar. 2. El dispositivo realiza una petición REST al servidor con el topográfico indicado. 3. El servidor obtiene los datos del catálogo, los trata y los devuelve al dispositivo. 4. El usuario recibe en el dispositivo un listado de resultados que se ajustan a su búsqueda. **Flujo de eventos alternativo:** **Precondiciones:** el usuario debe tener conexión a Internet.

Tabla 4.18: Módulo 3 - Especificación CU02-Buscar cabinas.

- Topográfico: elemento que identifica a un libro o conjunto de libros, ya sea parcial o completo.
- Identificador de cabina: elemento que identifica una cabina de la biblioteca.
- Elemento localizado: información correspondiente al elemento que se ajusta a la búsqueda del usuario.

ID: RD01 **Nombre:** Localización **Versión:** 1.0 **Fuente:** Ubik
Tipo: Registro en base de datos **Datos específicos:** Latitud, longitud y planta.
Latitud: latitud geográfica. **Longitud:** longitud geográfica. **Planta:** indicador de la planta de la biblioteca.

Tabla 4.19: Módulo 3 - Requisito de datos - Localización.

4.3.2. Diagrama de clases y documentación

Dados los casos de uso y los requisitos de datos identificados y especificados anteriormente, se han identificado las siguientes clases y sus respectivos atributos:

- Localización: id, latitud, longitud y altitud.
- Localizable: id, título, autor, estado y ubicación.

ID: RD02 **Nombre:** Topográfico **Versión:** 1.0 **Fuente:** Ubik
Tipo: Entrada **Datos específicos:** Latitud, longitud y planta.
Latitud: latitud geográfica. **Longitud:** longitud geográfica. **Planta:** indicador de la planta de la biblioteca.

Tabla 4.20: Módulo 3 - Requisito de datos - Topográfico.

ID: RD03 **Nombre:** Identificador de cabina **Versión:** 1.0 **Fuente:** Ubik
Tipo: Entrada **Datos específicos:** número identificador de cabina.
Identificador: número de tres cifras, comenzado por 1, 2, 3, o 5, que identifica las cabinas de estudio de la biblioteca

Tabla 4.21: Módulo 3 - Requisito de datos - Identificador de cabina.

ID: RD04 **Nombre:** Elemento localizado **Versión:** 1.0 **Fuente:** Ubik
Tipo: Salida **Datos específicos:** Información del elemento y su localización.
Información: identificador, nombre, autor (en caso de ser un libro)... **Localización:** latitud, longitud y planta donde se encuentra el elemento.

Tabla 4.22: Módulo 3 - Requisito de datos - Elemento localizado.

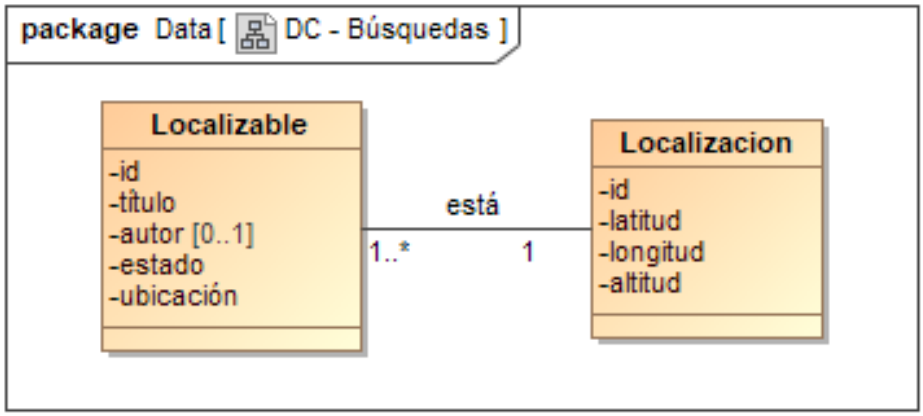


Figura 4.7: Módulo 3 - Diagrama de clases.

Estas dos clases se relacionan entre si mediante la relación *está*. Además, al igual que en los otros módulos, no se han identificado operaciones que aporten algún valor añadido al diagrama. Así pues, en la Figura 4.7 se muestra el diagrama de clases del presente módulo, y en las Tablas 4.23 y 4.24 puede verse su especificación.

Nombre: Localización **Versión: 1.0** **Fuente: Ubik**
Descripción: Los objetos de esta clase representan las localizaciones de la biblioteca donde se encuentran los libros y las cabinas.
Atributos: -id: identificador de la localización. -longitud: longitud geográfica del lugar. -latitud: latitud geográfica del lugar. -altitud: representa la planta del edificio.
Asociaciones: -está: con Localizable → 1..* En una localización hay uno o muchos elementos localizables (libros o cabinas).

Tabla 4.23: Módulo 3 - Especificación de la clase Localización.

Nombre: Localizable **Versión: 1.0** **Fuente: Ubik**
Descripción: Los objetos de esta clase representan los libros o cabinas, los cuales pueden ser localizados en un mapa al tener los datos de su localización.
Atributos: -id: identificador del elemento (topográfico o número de cabina). -título: nombre del elemento. -autor: en caso de ser un libro, autor del mismo. -estado: situación del elemento (disponible, en préstamo...). -ubicación: descripción textual de la ubicación del elemento. Normalmente será "Biblioteca".
Asociaciones: -está: con Localización $\rightarrow 1$ Un elemento localizable se encuentra solo en una localización.

Tabla 4.24: Módulo 3 - Especificación de la clase Localizable.

Capítulo 5

Diseño del sistema

5.1. Módulo 1: Algoritmo KNN Fingerprinting

En esta sección pueden verse los resultados de la fase de diseño del módulo 1. Estos resultados son el diagrama de clases de diseño (realizado a partir del diagrama de clases del análisis), y el diseño de la base de datos, que se utilizará para almacenar los datos necesarios para el funcionamiento del módulo.

5.1.1. Diagrama de clases de diseño

En la Figura 5.1 puede verse el diagrama de clases de diseño proveniente del análisis, donde se especifican los tipos de datos con los que se va a trabajar.

5.1.2. Diseño de base de datos

Como base de datos se emplean dos ficheros CSV (*Comma Separated Values*) que incluyen toda la información representada en el diagrama de clases de diseño de la figura 5.1. Se ha elegido este tipo de ficheros por su sencillez a la hora tanto de generarlos como de obtener la información que contienen. Estos ficheros en sí se almacenan en Firebase.

En primer lugar, se tiene un fichero que almacena los *fingerprints* de tal forma que cada fila del fichero se corresponde con un *fingerprint*. A su vez, cada columna del *fingerprint* se corresponde con la intensidad de la señal de una determinada antena de forma que todas las intensidades de las señales de una misma antena se encuentren en la misma columna. Para evitar columnas vacías, es decir, cuando en un *fingerprint* no se tenga una intensidad para una determinada antena, se indicará en dichas columnas un valor por defecto (-102). El fichero descrito se corresponde con la Tabla 5.1, donde A_n se corresponde con una determinada antena, y FP_k se corresponde con un *fingerprint*, compuesto por la posición donde fue tomado, P_k, y correspondientes intensidades para cada antena, I_{kn}.

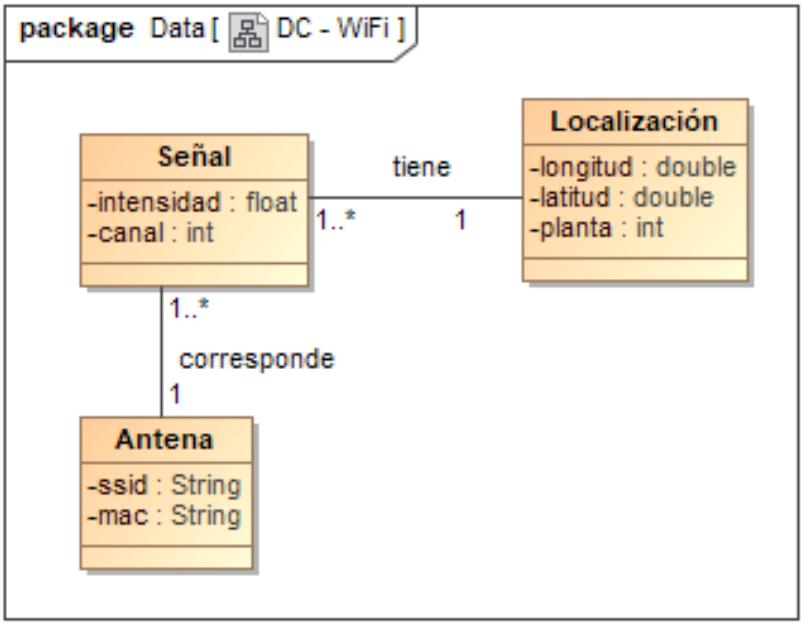


Figura 5.1: Módulo 1 - Diagrama de clases de diseño.

	A_1	A_2	...	A_n	P
FP_1	I_{11}	I_{12}	...	I_{1n}	P_1
FP_2	I_{21}	I_{22}	...	I_{2n}	P_2
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
FP_k	I_{k1}	I_{k2}	...	I_{kn}	P_k

Tabla 5.1: Módulo 1 - Estructura del fichero de almacenamiento de *fingerprints* de entrenamiento.

En segundo lugar, se tiene otro fichero que almacena la información sobre las antenas a las que pertenecen las señales recogidas en los *fingerprints*. Cada fila del fichero contiene la información referente a cada antena, y un número que indica en que columna del fichero de la Figura 5.1 se encuentran las intensidades de cada antena. La representación de este fichero puede verse en la Figura 5.2.

	$SSID$	CH	MAC	$INDEX$
A_1	$SSID_1$	CH_1	MAC_1	1
A_2	$SSID_2$	CH_2	MAC_2	2
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
A_n	$SSID_n$	CH_n	MAC_n	n

Tabla 5.2: Módulo 1 - Estructura del fichero de almacenamiento datos de antenas emisoras.

5.2. Módulo 2: Posicionamiento con BLE

En este apartado se muestran los resultados de la fase de diseño del módulo 2. En las siguientes subsecciones se encuentran el diagrama de clases de diseño realizado a partir del diagrama de clases del análisis y el diseño de la base de datos para almacenar los datos necesarios para el funcionamiento del módulo.

5.2.1. Diagrama de clases de diseño

En la Figura 5.2 se muestra el diagrama de clases de diseño donde se especifican los tipos de datos de los atributos con los que se va a trabajar.

Por lo que respecta a los atributos *frecuencia* y *potencia* de la clase Campaña, se deben tener en cuenta ciertas restricciones hardware:

- Frecuencia: también llamado intervalo de emisión, se indica en milisegundos en lugar de hercios. El valor debe estar entre los 100 y los 10000 milisegundos.
- Potencia: solo se podrán adoptar los valores -30, -20, -16, -12, -8, -4, 0, y +4 dBm.

5.2.2. Diseño de base de datos

Toda la información referente a las balizas y las campañas se almacena en Firebase Firestore. Esta información se organiza como un conjunto de documentos, donde cada documento se corresponde con una campaña con sus correspondientes atributos y un listado de las balizas que pertenecen a dicha campaña. Una pequeña representación puede verse en la Tabla 5.3

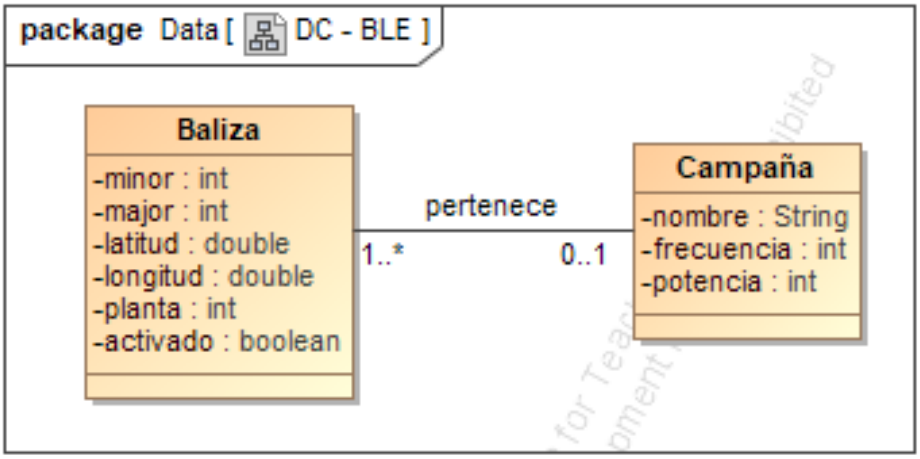


Figura 5.2: Módulo 2 - Diagrama de clases de diseño.

Campañas		
Biblioteca 5 planta	→	-Nombre -Frecuencia -Potencia -Balizas
Test laboratorio	→	-Nombre -Frecuencia -Potencia -Balizas
...	→	...

Tabla 5.3: Módulo 2 - Representación de la base de datos en Firebase Firestore.

5.3. Módulo 3: Búsquedas y mejora de usabilidad

En esta última sección del capítulo de diseño pueden verse los resultados de diseño del módulo 3. Estos resultados son el diagrama de clases de diseño y el diseño de la base de datos donde se almacenarán los datos necesarios de este módulo.

Cabe destacar que aunque en este módulo se va a realizar una pequeña modificación en la interfaz gráfica, no se han realizado prototipos. Esto es debido a que los cambios requeridos no afectan gravemente al actual diseño.

5.3.1. Diagrama de clases de diseño

En la Figura 5.3 puede verse el diagrama de clases de diseño a partir del diagrama de clases del análisis, donde se especifican los tipos de datos con los que se va a tratar. También se ha incluido una enumeración para especificar los posibles valores del atributo *estado* de la clase *Localizable*, los cuales pueden ser *disponible*, *préstamo*, *excluido* y *consulta*.

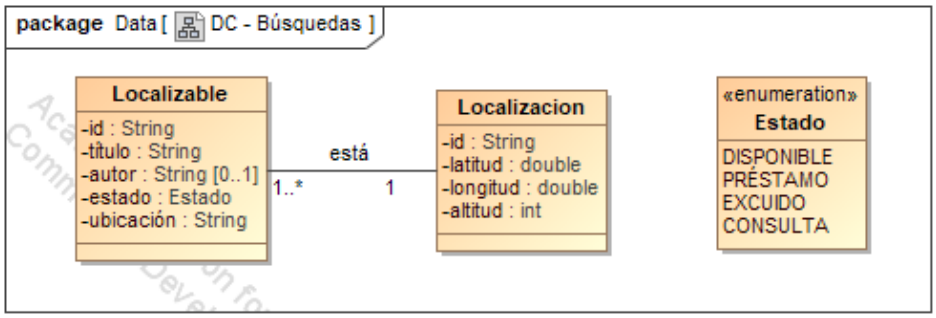


Figura 5.3: Módulo 3 - Diagrama de clases de diseño.

5.3.2. Diseño de base de datos

La información que se necesita almacenar son las localizaciones de la biblioteca, es decir, los datos que corresponderían a la clase *Localización* del diagrama de la Figura 5.3, ya que la información de la otra clase está almacenada en el catálogo de la UJI [9].

Está información se almacenará en Firebase Firestore, de forma que se tendrá una colección de documentos, donde cada documento corresponderá a una determinada localización (estantería, cabina) identificada por un determinado valor. Cada documento contendrá la latitud, longitud y altura de cada localización.

Capítulo 6

Implementación y pruebas

6.1. Módulo 1: Algoritmo KNN Fingerprinting

En esta sección se muestran algunos detalles de la implementación realizada, así como las pruebas unitarias, de integración y de validación del módulo ejecutadas.

6.1.1. Detalles de implementación

La implementación del módulo consiste en el desarrollo de un algoritmo KNN de *fingerprinting* concurrente para las estimaciones, junto con el código necesario para la descarga y lectura de los ficheros de base de datos y el estudio del nivel óptimo de concurrencia, y la toma de muestras Wi-Fi en el edificio de la biblioteca para construir la base de datos de *fingerprints*, con una aplicación proporcionada para dicho propósito.

En primer lugar, tenemos la parte de implementación propiamente dicha, donde se ha desarrollado el código necesario. Para la descarga de los ficheros de base de datos se descargan en primera instancia los metadatos de los ficheros en Firebase, y en caso de que los ficheros locales (si es que existen, en el primer arranque de la aplicación no existirán) estén desactualizados, se descargan los nuevos ficheros desde Firebase. Para la lectura de los ficheros de base de datos se han desarrollado dos clases, cada una para la lectura de uno de los ficheros identificados en la sección del diseño. El objetivo de estas clases es el almacenamiento de los datos contenidos en los ficheros en estructuras de datos Java para facilitar su posterior uso, en particular listas y diccionarios. Posteriormente, encontramos el algoritmo KNN de *fingerprinting*, desarrollado mediante dos clases, una de ellas extensora de la clase *Thread*. El flujo de ejecución y los pasos que realiza el algoritmo son los siguientes:

1. Al obtener una lectura de señales, se llama al método *estimate* pasando como parámetros un objeto que representa un *fingerprint* y un *listener* para notificar el resultado.
2. Se transforma el *fingerprint* recibido para adecuarse al modelo de *fingerprint* especificado en el diseño (Figura 5.1).

3. Se encuentran las K señales con mayor intensidad del *fingerprint*, así como a qué antena pertenecen.

4. Se crean y se arrancan los threads donde se producirán las comparaciones con la información de los ficheros de la base de datos. Los threads reciben su identificador, el número total de threads, el nuevo *fingerprint*, las estructuras de datos necesarias para los cálculos y una lista para recibir los resultados.

5. Cada thread, mediante un reparto de trabajo cíclico, compara las intensidades de las señales del nuevo *fingerprint* con las intensidades de las señales de los *fingerprints* almacenados, siempre que sus K señales de mayor intensidad correspondan a las mismas antenas. Posteriormente, en cada posición de la lista de resultados se anota el sumatorio de la diferencia de intensidades entre el nuevo *fingerprint* y con otro fingerprint almacenado.

6. Al finalizar los threads, se encuentran los K *fingerprints* almacenados con la menor diferencia de intensidades respecto al nuevo *fingerprint*. Posteriormente se obtiene la latitud, longitud y planta de estos *fingerprints* almacenados, y se realiza la media aritmética de estos datos. En este punto, ya se tiene una estimación de la posición del usuario calculada.

7. Finalmente, se construye un objeto con la información de la posición estimada y se envía (notifica) a través del *listener* recibido en un principio.

En segundo lugar, se realiza el estudio del nivel óptimo de concurrencia, es decir, el número óptimo de threads a utilizar durante el cómputo de las estimaciones. El Samsung Galaxy A5 cuenta con un procesador Exynos 7880, un octa-core con capacidad para tratar con hasta ocho threads [11]. Así pues, para comprobar el número óptimo de threads a emplear, se pone a trabajar al algoritmo con una base de datos de pruebas proporcionada, midiendo el tiempo de cómputo de las estimaciones con 2, 4, y 8 threads. El resultado de estos cómputos puede verse en la Figura 6.1, donde se presenta un gráfico CDF (*Cumulative Distribution Function*) con los resultados.

Estos resultados muestran que se consigue una reducción de tiempo en cualquier caso, siendo la mejor configuración el uso de 4 threads, ya que muestra un rendimiento idéntico a la configuración con 8 threads. Por otro lado, como puede verse en la Figura 6.1 las mejoras obtenidas tanto con las configuraciones de 2 y 4 threads son bastante modestas, consecuencia del tamaño de la base de datos de pruebas y algunas de las optimizaciones implementadas en el algoritmo (pe. solo comparar *fingerprints* cuyas señales más potentes sean de las mismas antenas).

Por último, se realiza la toma de muestras Wi-Fi en la biblioteca mediante una aplicación de captura (Figura 6.2) desarrollada por el estudiante de doctorado Germán Martín Mendoza Silva, con el objetivo de crear la base de datos de *fingerprints*. Durante esta fase se toman más de 2000 muestras entre las cinco plantas de la biblioteca, pertenecientes a diversos puntos del edificio, identificados por Germán.

6.1.2. Verificación y validación

Con el objetivo de comprobar el correcto funcionamiento del módulo implementado se realizan varías pruebas unitarias de caja negra y de integración en JUnit. Con el objetivo de validar

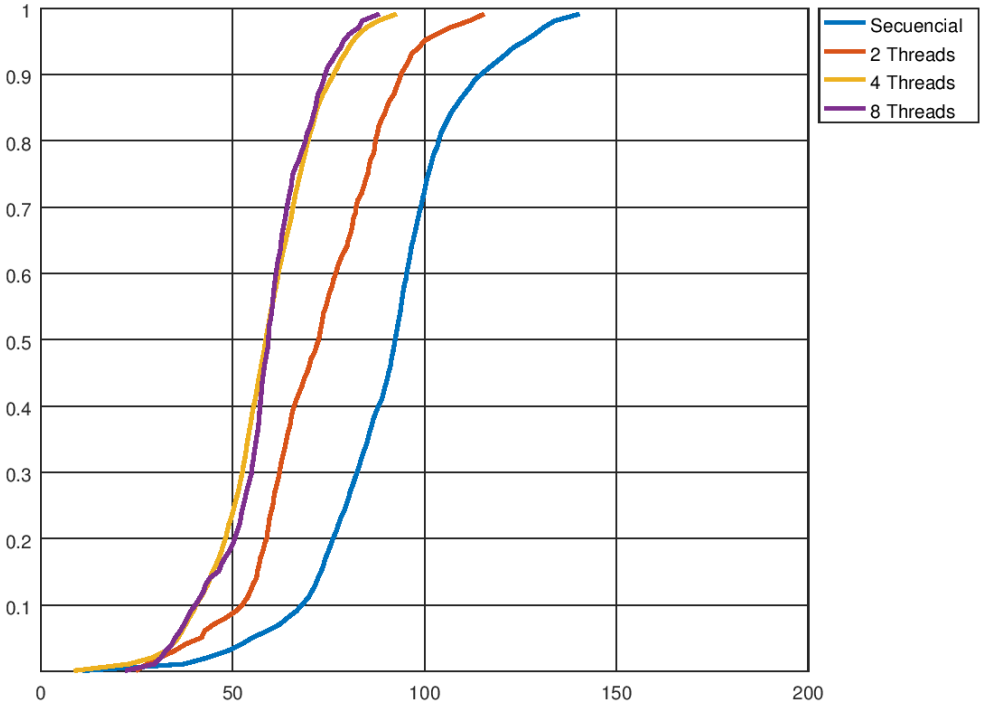


Figura 6.1: Gráfica CDF comparativa de tiempos de cómputo con varias configuraciones de uso de threads.

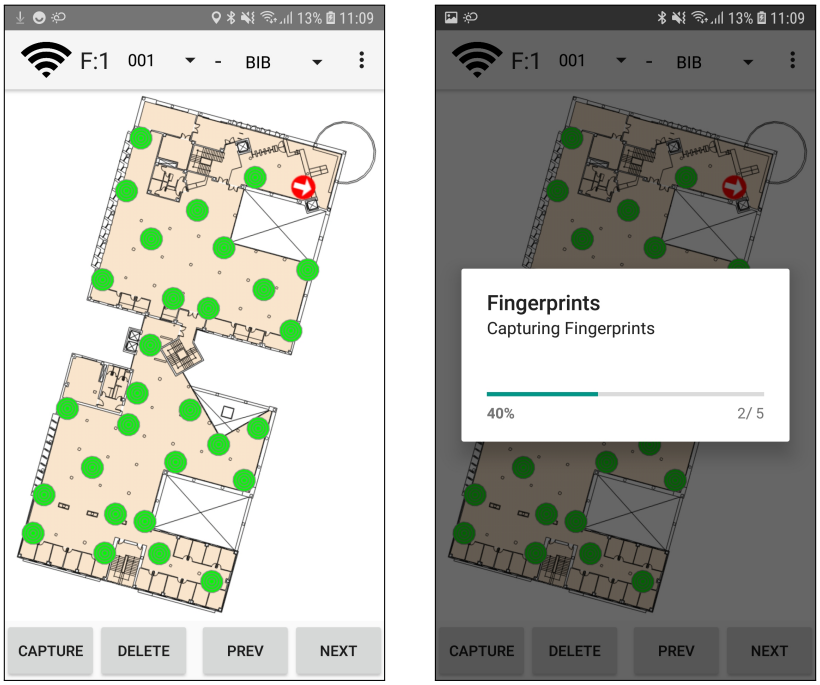


Figura 6.2: Aplicación de captura de señales Wi-Fi agrupadas en *fingerprints*.

el funcionamiento del sistema, se realiza una visita a la biblioteca y se observan las estimaciones de posición que proporciona la aplicación para determinar si el resultado es el esperado.

En primer lugar, se realizan unas pequeñas pruebas unitarias de caja negra con el objetivo de comprobar el correcto funcionamiento de la funcionalidad de lectura de información de los ficheros CSV de la base de datos. Al ser unas pruebas sencillas no ha sido necesario realizar el diseño de escenarios de casos de prueba.

En segundo lugar, se realizan unas pruebas unitarias de caja negra y de integración sobre el algoritmo de estimación de posiciones, así como de sus métodos auxiliares. En el caso de las unitarias, las pruebas se realizan sobre métodos aislado, mientras que en el caso de los de integración se comprueba el comportamiento completa de la clase de estimaciones con sus dependencias (clases de lectura de información de los ficheros CSV). Además, se ha realizado el diseño de los escenarios de casos de prueba para las pruebas de integración, los cuales pueden verse en la Tabla 6.1.

ID	Entrada (*fingerprint*)	Salida esperada (estimación)	Observaciones
1	[(UJI, 00:00, -90),(UJI, 00:01, -60), (UJI, 00:10, -70),(UJI, 00:11, -95), (UJI, 01:00, -60)]	(11/3, 9/3, 5)	Señales mínimas de misma intensidad
2	[(UJI, 00:00, -50),(UJI, 00:01, -80) ,(UJI, 00:10, -66),(UJI, 01:01, -78), (UJI, 01:00, -70)]	(8/3, 5/3, 3)	Señal de antena no presente en la BBDD
3	[(UJI, 00:00, -102),(UJI, 00:01, -50), (UJI, 00:10, -80),(UJI, 00:11, -102), (UJI, 01:00, -67)]	(11/3, 9/3, 5)	Número de señales validas igual a K
4	[(UJI, 00:00, -102),(UJI, 00:01, -50), (UJI, 00:10, -80),(UJI, 00:11, -70), (UJI, 00:01, -60)]	(11/3, 9/3, 5)	Dos señales distintas de la misma antena

Tabla 6.1: Diseño de pruebas de integración.

En todos las pruebas realizadas se ha empleado Mockito [21] para utilizar dobles de prueba para ciertas dependencias.

Finalmente, como aceptación del módulo implementado, se pone en marcha la aplicación en una visita a la biblioteca frente a las partes interesadas y se observan las estimaciones proporcionadas con respecto a nuestra posición real, mientras nos vamos desplazando por el edificio. El resultado de la prueba es satisfactorio, ya que se obtienen unas estimaciones aceptables para la cantidad de muestras recogidas durante la implementación, dando la capacidad al usuario de orientarse correctamente.

Estos resultados pueden verse en las imágenes de la Figura 6.3. En las imágenes se muestra como un círculo hueco la posición estimada por el dispositivo y en un cuadrado morado la posición real del usuario. (Nota: las imágenes han sido tomadas en un momento posterior del desarrollo, por lo que la interfaz gráfica se ve cambiada respecto a su apariencia en el estado inicial)

Figura 6.3: Comparativa de estimaciones Wi-Fi respecto a la posición real del usuario.

6.2. Módulo 2: Posicionamiento con BLE

En este apartado pueden verse los detalles de la implementación, las pruebas unitarias y de validación realizadas sobre este segundo módulo del proyecto.

6.2.1. Detalles de implementación

La implementación de este módulo esta compuesta por el desarrollo del código necesario para la descarga y lectura de la campaña activa (información de las balizas del despliegue), para la captura de las señales BLE del entorno, y el algoritmo *Weighted Centroid* para las estimaciones mediante Bluetooth. También se realiza un pequeño estudio para determinar la configuración y despliegue óptimos de las balizas.

En primer lugar, para la descarga de los ficheros se ha empleado una aproximación distinta a la realizada en el módulo anterior. En este caso, la información sobre el despliegue se descarga, haya habido cambios en él o no, al iniciar la aplicación. Esto es debido a que esta información no se almacena en ficheros internos, como en el caso de la base de datos de *fingerprints* del módulo anterior. Su descarga no supone un sobrecoste al tratarse de un conjunto pequeño de datos, a diferencia de la base de datos de *fingerprints*.

En segundo lugar, para la captura de las señales BLE hay dos posibles aproximaciones. La API de Android nos permite acceder al adaptador Bluetooth del dispositivo y ordenarle iniciar el escaneo. Para ello se le debe proporcionar un *callback* donde se reciben los resultados. Mediante el *callback* se pueden recibir las señales a medida que estas se detectan, o bien se pueden recibir en paquetes (lote de señales o *batch*) [24]. Para probar qué sistema ofrece mejores resultados se realizan ambas implementaciones con los ajustes más agresivos posibles (baja latencia, sin retardos, señales débiles...), donde se obtienen los siguientes resultados:

- Implementación continua: se recogen las señales de la forma esperada. Sin embargo, el recibir señales individuales complicará la implementación, ya que para el algoritmo se requerirá un conjunto de señales cercanas en el tiempo.
- Implementación por lotes: este modo sería el más adecuado para el sistema al recibir las señales por lotes. Sin embargo, debido a la configuración del escaneo, se observa que se obtienen lotes de señales repetidos. Es decir, al no tener un nuevo lote señales cuando es requerido, el adaptador devuelve el lote anterior como uno nuevo, lo cual no es válido ya que se necesita tener la información de las señales actualizada.

Debido a los inconvenientes de la implementación por lotes, se decide emplear la implementación continua. Esta decisión implica desarrollar una forma para mantener agrupadas las señales cercanas en el tiempo. Para ello, se implementa una lista con ventana de tiempo, es decir, una lista donde se encontrarán las señales que hayan sido recibidas como máximo en un determinado tiempo. Por ejemplo, con una ventana de 10 segundos, se mantendrán en la lista las señales recogidas como máximo 10 segundos atrás. Téngase en cuenta que el valor de la ventana es configurable y que todos los métodos que tratan con esta lista son *thread safe*. En la Figura 6.4 se ilustra este ejemplo.

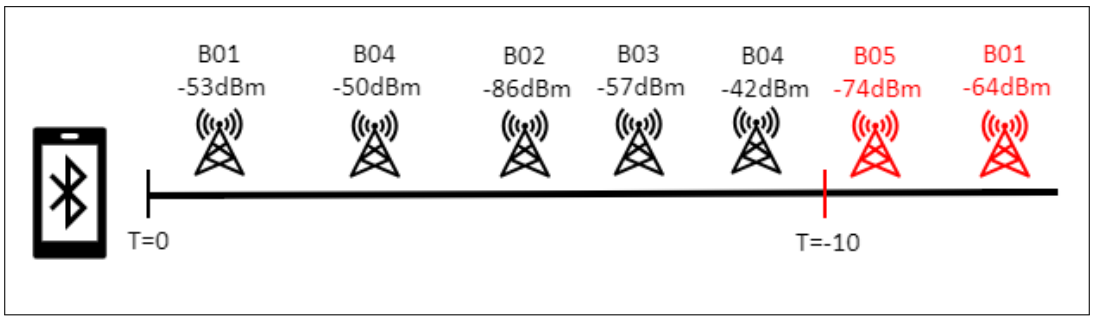


Figura 6.4: Ejemplo con ventana de 10 segundos, en rojo las señales descartadas.

Esta implementación abre otro pequeño problema: es posible que dentro de la ventana aparezcan varías intensidades de la misma baliza, como ocurre en la Figura 6.4 con B04. Para solventar dicho problema se decide realizar la media aritmética con las intensidades de aquellas balizas que aparezcan varias veces en la ventana. En el ejemplo, para B04 se obtendría una intensidad de -46dBm.

En tercer lugar, el algoritmo *Weighted Centroid* es el encargado de proporcionarnos una estimación. Cada unos determinados segundos (periodo de notificación configurable) se solicitan los datos sobre las señales capturadas que se encuentran dentro de la ventana de tiempo y dicho algoritmo proporciona una estimación. Los pasos que realiza el algoritmo son los siguientes:

1. Al finalizar el periodo de notificación, se llama al método *estimate* pasando como parámetros el listado de señales dentro de la ventana de tiempo y el listado de las balizas.

2. Se filtran las señales recibidas para asegurar que no exista ninguna que no pertenezca a una baliza (señales de ratones inalámbricos, *wearables*...).

3. Se identifican las K señales de mayor intensidad, así como las balizas a las que pertenecen.

4. Se calculan los pesos de las K señales con la fórmula $100^{\frac{rssi}{10}}$, variación de la presentada en [17]. Dicha fórmula se origina de 6.1, donde n es un factor relacionado con el entorno ($n = 1$ para entornos simples, $n < 1$ para entornos complejos) y en este caso se selecciona $n = 1/2$. En 6.2 se realiza el desarrollo donde se obtiene la fórmula indicada.

$$\begin{aligned} RSSI &= 10n\log(d) \qquad (6.1)\\ &= 10(\frac{1}{2})\log(d),\\ \log(d) &= \frac{2RSSI}{10} \qquad (6.2)\\ &= \log(10^{\frac{2RSSI}{10}}),\\ d &= 10^{\frac{2RSSI}{10}} = 100^{\frac{RSSI}{10}} \end{aligned}$$

5. Mediante los pesos (w_i) y la posición conocida de las balizas a las que pertenecen las K señales (Lat_i, Lon_i), se calcula la latitud y longitud de la estimación mediante la fórmula 6.3. En la Figura 6.5 se muestra una ilustración de ejemplo.

$$(Lat_{est}, Lon_{est}) = (\frac{\sum_{k=1}^{K} w_i * Lat_i}{\sum_{k=1}^{K} w_i}, \frac{\sum_{k=1}^{K} w_i * Lon_i}{\sum_{k=1}^{K} w_i}) \tag{6.3}$$

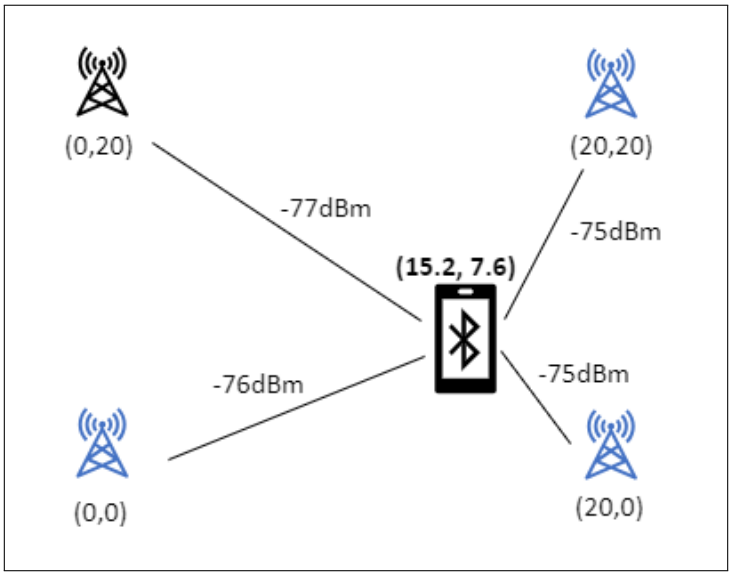


Figura 6.5: Ejemplo de centroide ponderado, en azul las balizas con señales más potentes.

Finalmente, para determinar la configuración óptima de las balizas se han realizado pruebas para observar el comportamiento de las señales. Como resultado se ha observado que la frecuencia de emisión es altamente importante para que el dispositivo móvil detecte la señal, ya que hay que tener en cuenta que es posible que una determinada emisión no sea detectada. La potencia de emisión, por contra, no afecta tan drásticamente al comportamiento de las señales. Así pues, como configuración se elige una frecuencia (intervalo de emisión) de 100 milisegundos (la mínima), y una potencia de emisión baja, de -16dBm, para intentar compensar el consumo de energía debido a la baja frecuencia de emisión. Esta configuración proporciona una autonomía a las balizas de aproximadamente 6 meses.

Además, el despliegue de las balizas (24 unidades) en la quinta planta de la biblioteca se ha realizado con el objetivo de tener una distancia mínima de 3 metros entre balizas para cubrir la máxima superficie posible (Figura 6.6).

6.2.2. Verificación y validación

Para comprobar el correcto funcionamiento del módulo implementado, se han realizado unas pequeñas pruebas unitarias sobre varios puntos clave del módulo. Por falta de tiempo, no se ha podido realizar el diseño de las pruebas para tener una batería de pruebas más robusta.

Estas pruebas se han realizado sobre la funcionalidad de la ventana de tiempo en la lista de señales, la obtención de las señales cuando varias provienen de la misma baliza, y sobre el

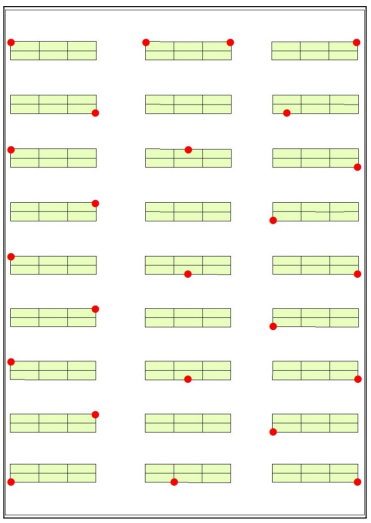


Figura 6.6: Despliegue de 24 balizas en la quinta planta de la biblioteca.

algoritmo de estimación. Tras la ejecución de las pruebas se encontró un error en el algoritmo de estimación, específicamente en el calculo de la nueva posición empleando los pesos calculados, donde la suma de las latitudes y longitudes no se dividía por la suma de los pesos, sino por el último peso calculado. El error se produjo por un despiste a la hora de usar las variables, y su solución consistió en emplear correctamente la variable donde se acumulaban los pesos.

Por otra parte, para validar el funcionamiento del sistema, se realiza una visita a la biblioteca para comprobar que las estimaciones proporcionadas por la aplicación mientras nos desplazamos por la zona donde están colocadas las balizas. El resultado es satisfactorio, incluso mejor de lo esperado. Los resultados pueden verse en la Figura 6.7, donde la posición estimada se muestra como un círculo hueco y la posición real por un cuadrado morado.

6.3. Módulo 3: Búsquedas y mejora de usabilidad

Por último, es esta sección se muestran los detalles de la implementación del correspondiente módulo así como las pruebas de validación realizadas.

6.3.1. Detalles de implementación

La implementación de este módulo tiene dos partes diferenciadas. Por un lado tenemos la implementación de las búsquedas de libros por topográfico y cabinas, y por otro la mejora de la interfaz de la aplicación.

En primer lugar, la búsqueda de libros por topográfico y cabinas se implementa como un servicio en Node.js. Este servicio realiza peticiones a la web del catálogo [9] y mediante *web scraping* se obtiene la información definida en el análisis y diseño de dicho módulo. Así pues, se realizan las siguientes acciones:

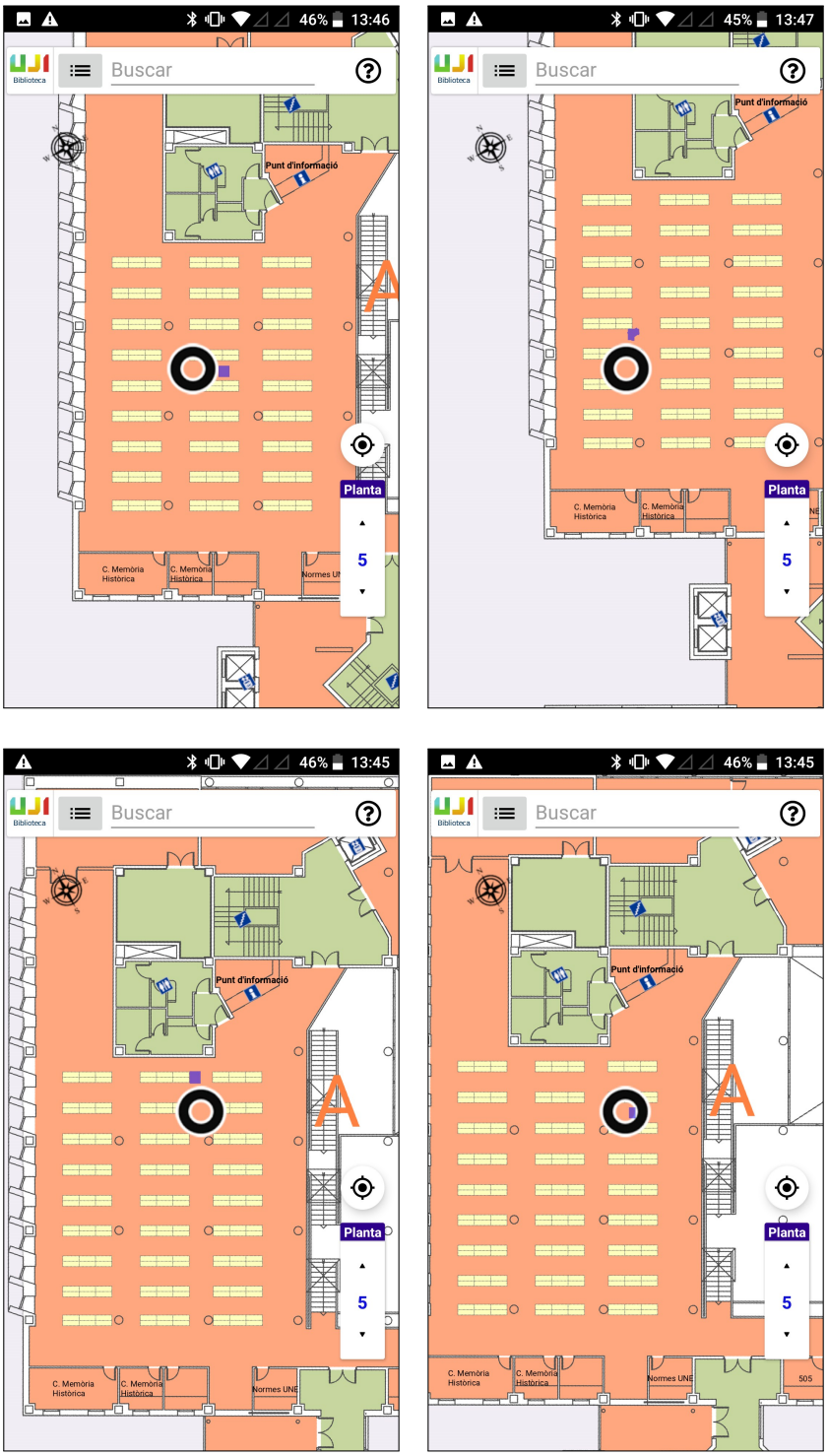


Figura 6.7: Comparativa de estimaciones BLE respecto a la posición real del usuario.

Figura 6.8: Resultado de una búsqueda por topográfico completo en el catálogo.

- Búsqueda por topográfico: dado el topográfico parcial o completo se realiza una petición al catálogo mediante la siguiente url:

  http://cataleg.uji.es/search*spi/?searchtype=c&searcharg=ARG&SORT=D&searchscope=1, donde ARG se sustituye por el topográfico que solicita el usuario.

  Dependiendo de si el topográfico es completo o parcial, el resultado de la búsqueda varia, por lo que conviene realizar las siguientes acciones:

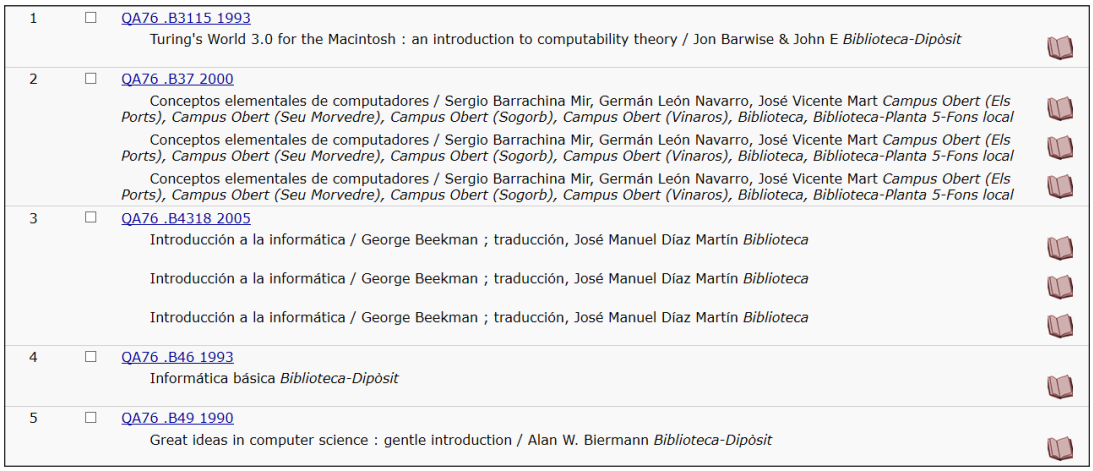
  - Si el topográfico es **completo**, se obtiene la web con toda la información relevante que se necesita (Figura 6.8), por lo que esta se extrae mediante *web scraping*.
  - Si el topográfico es **parcial** (caso más probable), se obtiene un listado de resultados que encajan con la consulta (Figura 6.9). Sin embargo, en este listado no tenemos los datos que necesitamos, por lo que debemos realizar las siguientes acciones:
    - Mediante *web scraping*, obtener los enlaces en el topográfico de cada uno de los primeros 20 resultados (limitación para reducir la latencia de la búsqueda).
    - Se realiza una nueva petición por cada uno de los resultados a la siguiente url: http://cataleg.uji.es/search S1*spi?/HREF, donde HREF es el enlace obtenido en el paso anterior.
    - El resultado de la petición anterior se trata como si se realizase una búsqueda por topográfico completo, ya que con las peticiones del paso anterior se obtienen resultados con la estructura de la Figura 6.8.

- Búsqueda de cabinas: dado el número identificador de 3 dígitos de una cabina se realiza una petición al catálogo mediante la siguiente url:

  http://cataleg.uji.es/search S1*cat?/a/a/1,1,1,E/holdings&FF=ccabina+ID, donde ID es el número identificado de la cabina.

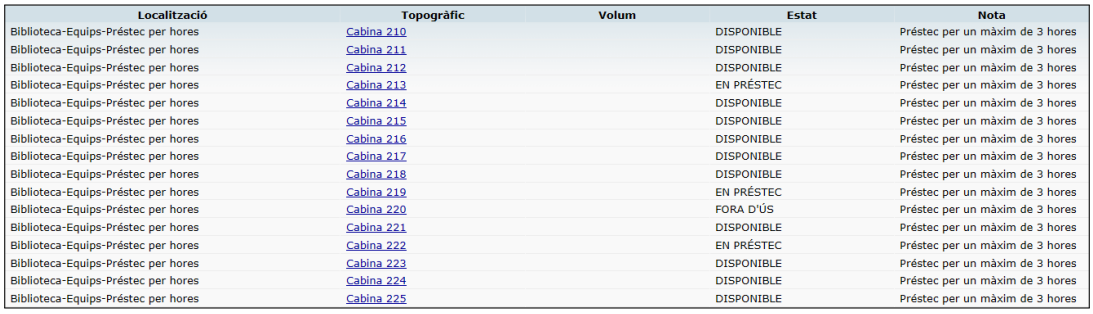
  Con esta petición se obtiene un listado todas con las cabinas de la misma planta a la consultada (Figura 6.10), y por decisión del cliente, se obtiene como resultado mediante *web scraping* la información de todas las cabinas del listado, no solo de la buscada por el usuario.

Por lo que respecta a la parte de la aplicación Android, al utilizarse Swagger [26] en el servidor, con Swagger Codegen [27] podemos generar automáticamente el código que realiza las peticiones REST desde la aplicación en forma de API. De forma, la aplicación simplemente



1	☐	QA76 .B3115 1993 Turing's World 3.0 for the Macintosh : an introduction to computability theory / Jon Barwise & John E *Biblioteca-Dipòsit*
2	☐	QA76 .B37 2000 Conceptos elementales de computadores / Sergio Barrachina Mir, Germán León Navarro, José Vicente Mart *Campus Obert (Els Ports), Campus Obert (Seu Morvedre), Campus Obert (Sogorb), Campus Obert (Vinaros), Biblioteca, Biblioteca-Planta 5-Fons local* Conceptos elementales de computadores / Sergio Barrachina Mir, Germán León Navarro, José Vicente Mart *Campus Obert (Els Ports), Campus Obert (Seu Morvedre), Campus Obert (Sogorb), Campus Obert (Vinaros), Biblioteca, Biblioteca-Planta 5-Fons local* Conceptos elementales de computadores / Sergio Barrachina Mir, Germán León Navarro, José Vicente Mart *Campus Obert (Els Ports), Campus Obert (Seu Morvedre), Campus Obert (Sogorb), Campus Obert (Vinaros), Biblioteca, Biblioteca-Planta 5-Fons local*
3	☐	QA76 .B4318 2005 Introducción a la informática / George Beekman ; traducción, José Manuel Díaz Martín *Biblioteca* Introducción a la informática / George Beekman ; traducción, José Manuel Díaz Martín *Biblioteca* Introducción a la informática / George Beekman ; traducción, José Manuel Díaz Martín *Biblioteca*
4	☐	QA76 .B46 1993 Informática básica *Biblioteca-Dipòsit*
5	☐	QA76 .B49 1990 Great ideas in computer science : gentle introduction / Alan W. Biermann *Biblioteca-Dipòsit*

Figura 6.9: Resultado de una búsqueda por topográfico parcial en el catálogo.



Localització	Topogràfic	Volum	Estat	Nota
Biblioteca-Equips-Préstec per hores	Cabina 210		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 211		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 212		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 213		EN PRÉSTEC	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 214		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 215		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 216		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 217		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 218		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 219		EN PRÉSTEC	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 220		FORA D'ÚS	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 221		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 222		EN PRÉSTEC	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 223		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 224		DISPONIBLE	Préstec per un màxim de 3 hores
Biblioteca-Equips-Préstec per hores	Cabina 225		DISPONIBLE	Préstec per un màxim de 3 hores

Figura 6.10: Resultado de una búsqueda de cabinas en el catálogo.

debe utilizar las llamadas de esta API para realizar las peticiones, aunque se siguen ciertas restricciones en cuando realizar cada petición:

- Búsqueda de cabina: se realiza esta petición si el texto de búsqueda comienza por la palabra *cabina* seguida de un espacio y 3 cifras. Además por decisión del cliente, también se realiza una búsqueda por palabra clave (ya implementada en el sistema inicial, véase Capítulo 2), pero priorizando los resultados de las cabinas.
- Búsqueda por topográfico: se realiza esta petición cuando el texto de búsqueda es mayor de dos caracteres, lo que permite reducir el número de peticiones y realizar un mejor filtrado desde el principio ya que los dos primeros caracteres indican la temática del libro.

En segundo lugar, por lo que respecta a la parte de la usabilidad, se han realizado las siguientes mejoras a petición de los clientes:

- Como mejora puramente estética se ha creado una pantalla de carga mientras de descargan ciertos datos de configuración de Firebase (Figura 6.11), cumpliendo con el Manual de identidad visual corporativa de la UJI [19].
- En la versión inicial, para mostrar los resultados una vez ocultos se debía seleccionar el cuadro de introducción de texto, y para ocultarlos se debía seleccionar uno, o bien seleccionar un botón con el símbolo X, que además eliminaba los resultados. Para solucionarlo se ha incluido un botón que permite mostrar y ocultar los resultados de una búsqueda (Figura 6.12).
- Se han incluido los siguientes elementos representativos en el mapa, visibles en la Figura 6.13:
  - Iconos: puntos de información, escaleras, ascensores y aseos.
  - Etiquetas: puntos de información, punto de préstamo y cabinas.
- Se ha incluido un texto de ayuda para ayudar a la utilización de la aplicación, como puede verse en la Figura 6.14, y en el Anexo A.

6.3.2. Verificación y validación

Para comprobar el correcto funcionamiento de la funcionalidad implementada se han realizado varias pruebas, algunas de ellas manuales y otras automatizadas.

En el caso de búsqueda, tanto de libros por topográfico como de cabinas, se han realizado las mismas búsquedas en el catálogo de la biblioteca y a través del servicio implementado para comprobar que los resultados obtenidos son los mismos (Figura 6.15 y 6.16).

En el caso de las mejoras de usabilidad, se han realizado simples inspecciones visuales con el fin de comprobar que el comportamiento de estas mejoras era correcto. Por ejemplo, en el caso

Figura 6.11: Pantalla de carga de aplicación con imagen de la biblioteca y marca UJI.

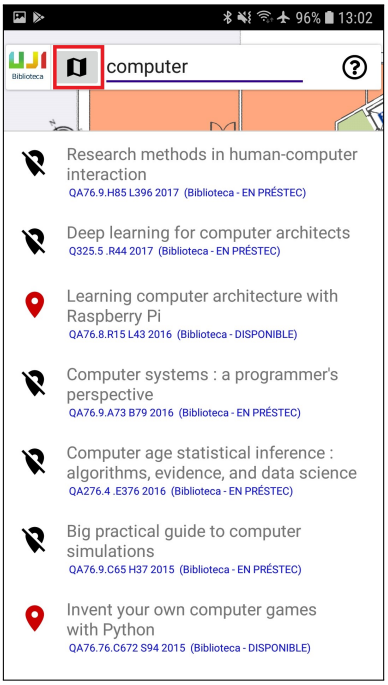


Figura 6.12: Botón para mostrar/ocultar resultados, resaltado en rojo.

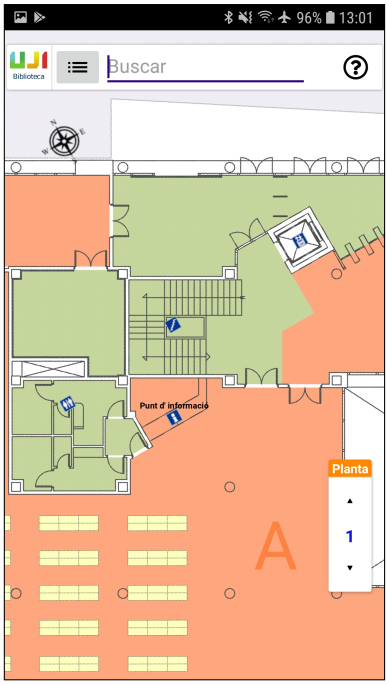


Figura 6.13: Iconos y etiquetas.

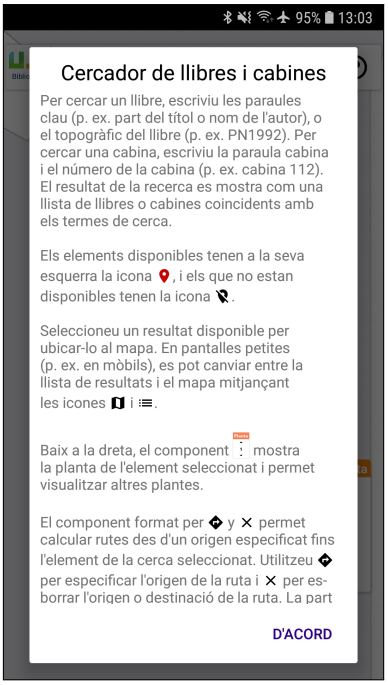


Figura 6.14: Pantalla de ayuda.

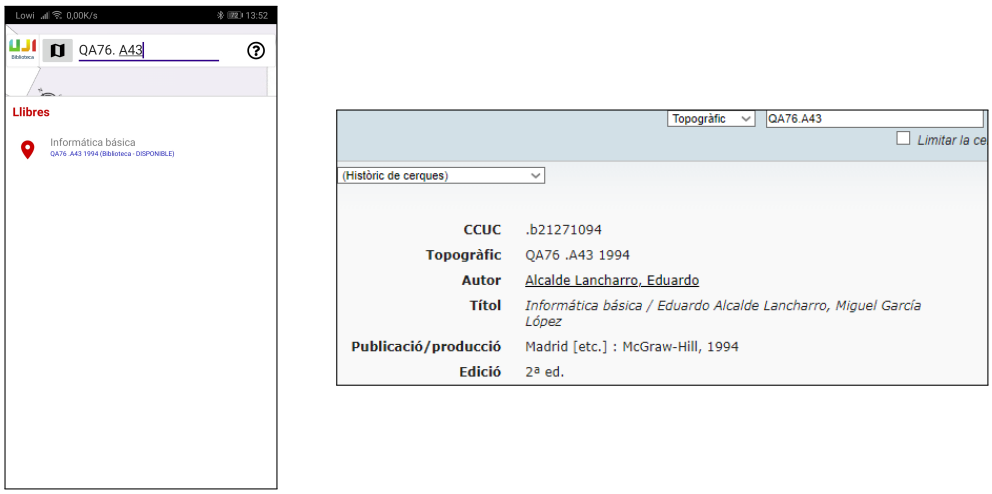


Figura 6.15: Resultado de búsqueda por topográfico. A la izquierda el resultado en la aplicación. A la derecha el resultado en el catálogo.

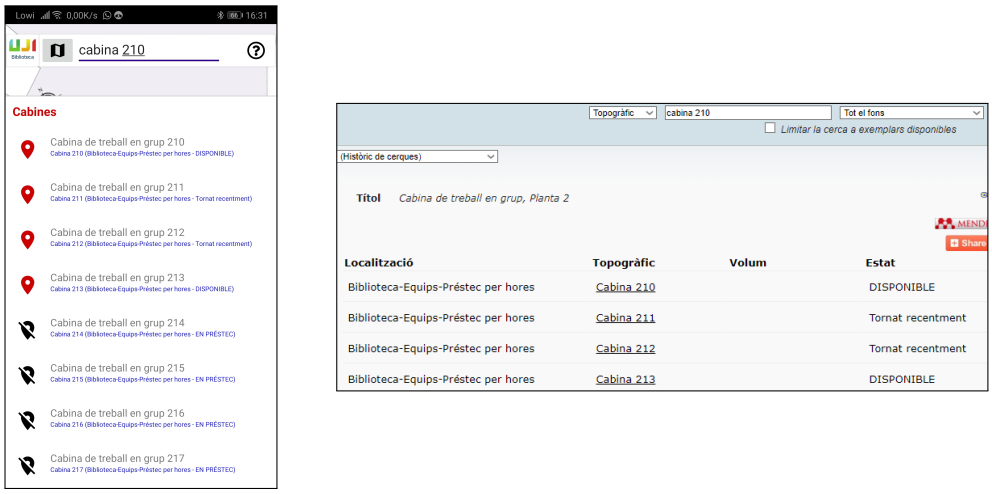


Figura 6.16: Resultado de búsqueda de cabinas. A la izquierda el resultado en la aplicación. A la derecha el resultado en el catálogo.

de las etiquetas de las cabinas, se comprobó que estas se actualizaban correctamente al cambiar de planta.

Finalmente, se han ejecutado pruebas automáticas en Firebase Test Lab [29]. Con esta herramienta se ha creado un script de pruebas sobre la aplicación. Test Lab utiliza dicho *script* y la *apk* (***A**ndroid **A**pplication Pac**k**age*, paquete de instalación para Android con extensión .apk) que se le proporciona para ejecutar las pruebas en terminales físicos y virtuales, proporcionándonos en poco tiempo los resultados compuestos por el registro de la aplicación, capturas de pantalla, un vídeo con la ejecución de las pruebas (Figura 6.17), estadísticas de rendimiento (Figura 6.18), posibles problemas...

Figura 6.17: Vídeo y secuencia de pasos que se ha realizado durante el test.

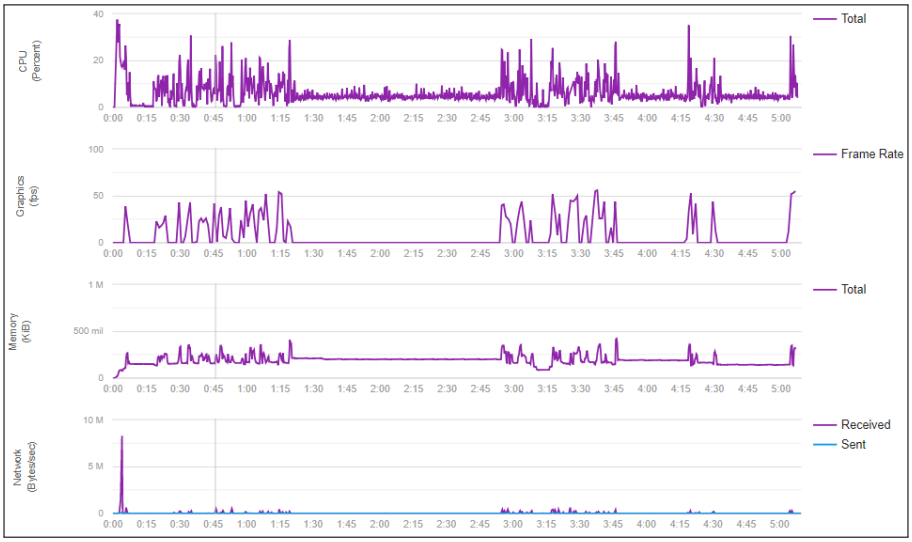


Figura 6.18: Gráficas de rendimiento: CPU, FPS, RAM y red.

6.4. Sistema final

En este apartado se muestra el resultado del desarrollo mediante el esquema final del sistema, y se demuestra la consecución de los objetivos y funcionalidades fijadas al comienzo del proyecto

6.4.1. Esquema del sistema final

En el apartado 2.1 se define el sistema inicial, y en concreto, en la Figura 2.4 se muestra el esquema del sistema inicial. A continuación, en la Figura 6.19 se muestra el esquema final del sistema tras la implementación de los módulos.

Como puede verse, el sistema cuenta con tres nuevos componentes, *BLE* (Módulo 2), *Topográfico* y *Búsqueda de cabinas* (Módulo 3). El Módulo 1 se correspondería con el componente *WiFi Indoor*, que recordemos, ya estaba presente en el sistema inicial, pero haciendo uso de un servidor de estimaciones.

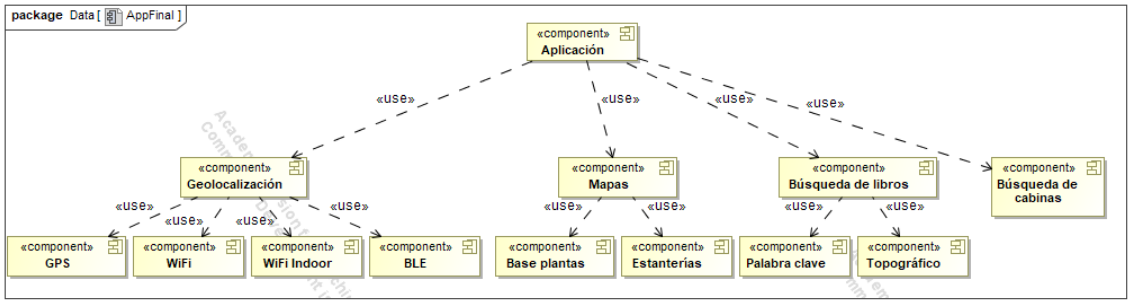


Figura 6.19: Esquema con los componentes del sistema final.

6.4.2. Consecución de objetivos

En el apartado 1.2 se definieron los objetivos del proyecto. A continuación, se indica de que manera se ha validado el cumplimiento de los mismos:

- Desarrollar un algoritmo *KNN Fingerprinting*: en la sección de verificación y validación del Módulo 1 (6.1.2) se muestra el proceso y las imágenes capturadas del funcionamiento del algoritmo *KNN* (Figura 6.3). De la misma forma, en la Figura 6.1 se muestra el estudio de concurrencia realizado para optimizar el algoritmo.
- Desarrollar un algoritmo *Weighted Centroid*: en la sección de verificación y validación del Módulo 2 (6.2.2) se muestra el proceso y las imágenes capturadas del funcionamiento del algoritmo *Weighted Centroid* (Figrua 6.7).
- Análisis de la configuración óptima de los *beacons* BLE: por falta de tiempo, el estudio de la configuración óptima de las balizas no pudo realizarse todo lo bien que se hubiera deseado. Aún así, en el apartado de la implementación del Módulo 2 (6.2.1) se comenta brevemente la mejor configuración de frecuencia-potencia-duración de batería encontrada.

- Desarrollo de un algoritmo de búsqueda de libros por topográfico, desarrollo de un algoritmo de búsqueda de cabinas, e inclusión de elementos en los mapas que faciliten la orientación del usuario y mejoras en la interfaz de usuario a nivel de usabilidad: en la sección de verificación y validación del Módulo 3 (6.3.2) se muestran las imágenes capturadas tanto del funcionamiento de las búsquedas de topográficos y cabinas (Figura 6.15 y 6.16), así como las mejoras de usabilidad (Figuras 6.11, 6.12, 6.13 y 6.14).

Capítulo 7

Conclusiones

En este proyecto se han desarrollado una serie de módulos para una aplicación para la biblioteca de la Universitat Jaume I, con el objetivo de mejorar la experiencia de los usuarios de la biblioteca. El proyecto ha sido un éxito, ya que la aplicación fue presentada en la biblioteca con motivo del día del libro y publicada en la Google Play [4], a fecha de redacción de este documento, con la funcionalidad correspondiente al Módulo 3, liberándose en nuevas versiones el resto de funcionalidad.

Desde un punto de vista técnico, todos los objetivos planteados al inicio del proyecto han sido cumplidos de forma satisfactoria. En concreto, los resultados del Módulo 2 en lo correspondiente a la precisión de las estimaciones fue muy bueno, incluso mejor de lo esperado.

Desde una perspectiva de gestión de proyectos, se ha realizado una buena gestión del mismo, sobreponiéndonos a variaciones drásticas en la planificación o la aparición de nuevos requisitos sin que el proyecto en general se viese afectado. Una vez finalizado el proyecto, y teniendo más experiencia con las metodologías ágiles (por la formación en el grado), es posible que Scrum hubiese sido una mejor metodología de gestión que la tradicional en cascada utilizada (aunque ésta haya funcionado bien).

Por último y personalmente, la realización de este proyecto y estancia me ha enriquecido a nivel profesional y personal, ya que me he integrado en la empresa correctamente con un equipo multidisciplinar y he trabajado en un proyecto real colaborando de forma cercana con los clientes. También quiero destacar la utilidad de los conocimientos transversales y de Java aportados en los estudios para el desarrollo del proyecto, aunque veo necesaria la inclusión en el grado de materias obligatorias sobre la formación en el lenguaje Javascript y en el desarrollo de aplicaciones móviles.

Bibliografía

[1] iBKS105 - Accent Systems. https://accent-systems.com/es/producto/ibks-105/. [Consulta: 19 de febrero de 2019].

[2] Android SDK. https://es.wikipedia.org/wiki/Android_SDK. [Consulta: 6 de Febrero de 2019].

[3] Android Studio. https://es.wikipedia.org/wiki/Android_Studio. [Consulta: 6 de Febrero de 2019].

[4] Biblioteca Universitat Jaume I - Aplicaciones en Google Play. https://play.google.com/store/apps/details?id=es.uji.geotec.campus&gl=ES. [Consulta: 8 de Mayo de 2019].

[5] ArcGIS. https://es.wikipedia.org/wiki/ArcGIS. [Consulta: 10 de Marzo de 2019].

[6] ArcGIS Pro. https://pro.arcgis.com/es/pro-app/get-started/overview-of-arcgis-pro.htm. [Consulta: 10 de Marzo de 2019].

[7] Guide-ArcGIS Runtime SDK for Android 10.2.9. https://developers.arcgis.com/android/10-2/guide/guide.htm. [Consulta: 10 de Marzo de 2019].

[8] Anahid Basiri. Indoor location based services challenges, requirements and usability of current solutions. *Computer Science Review*, 24:1–12, 2017.

[9] Catàleg de la biblioteca de la Universitat Jaume I. http://cataleg.uji.es/. [Consulta: 30 de Marzo de 2019].

[10] Docker. https://es.wikipedia.org/wiki/Docker_(software). [Consulta: 11 de Marzo de 2019].

[11] Samsung Exynos 7880. https://www.notebookcheck.net/Samsung-Exynos-7880-SoC-Benchmarks-and-Specs.208000.0.html. [Consulta: 15 de Febrero de 2019].

[12] Firebase - Google. https://firebase.google.com/. [Consulta: 7 de Febrero de 2019].

[13] Git - About Version Control. https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control. [Consulta: 7 de Febrero de 2019].

[14] Github - The world's leading software development platform. https://github.com/. [Consulta: 7 de Febrero de 2019].

[15] IntelliJ IDEA. https://es.wikipedia.org/wiki/IntelliJ_IDEA. [Consulta: 10 de Marzo de 2019].

[16] Java development tools. https://es.wikipedia.org/wiki/Java_Development_Kit. [Consulta: 6 de Febrero de 2019].

[17] Elena Simona Lohan, Jukka Talvitie, Pedro Figueiredo e Silva, Henri Nurminen, Simo Ali-Löytty, and Robert Piché. Received signal strength models for wlan and ble-based indoor positioning in multi-floor buildings. In *2015 International Conference on Location and GNSS (ICL-GNSS)*, pages 1–6. IEEE, 2015.

[18] MagicDraw: Architecture made simple. https://www.nomagic.com/products/magicdraw. [Consulta: 21 de Febrero de 2019].

[19] Manual d'identitat visual y corporativa. Servei de Comunicació i Publicacións. http://ujiapps.uji.es/ade/rest/storage/4XSPHOILUWABW75P8A1YIPSL48EKGKZS. [Consulta: 28 de Marzo de 2019].

[20] M&M18. *Location-Based Services and Real-Time Location Systems Market by Location Type, Software, Hardware, Service, Vertical And Region - Global Forecast to 2023*. Markets and Markets, 2018.

[21] Mockito. https://en.wikipedia.org/wiki/Mockito. [Consulta: 7 de Febrero de 2019].

[22] Node.js. https://es.wikipedia.org/wiki/Node.js. [Consulta: 10 de Marzo de 2019].

[23] Ordenador ASUS X556UV. https://www.asus.com/es/Laptops/ASUS-Vivobook-X556UV/overview/. [Consulta: 21 de Febrero de 2019].

[24] ScanCallback — Android Developers. https://developer.android.com/reference/android/bluetooth/le/ScanCallback.html#public-methods. [Consulta: 19 de Abril de 2019].

[25] Samsung Galaxy A5 2017 SM-A520F. https://www.samsung.com/es/smartphones/galaxy-a5-2017-a520/SM-A520FZKAPHE/. [Consulta: 21 de Febrero de 2019].

[26] Swagger. https://en.wikipedia.org/wiki/Swagger_(software). [Consulta: 10 de Marzo de 2019].

[27] Swagger Codegen — API Development Tools. https://swagger.io/tools/swagger-codegen/. [Consulta: 10 de Marzo de 2019].

[28] Swagger UI — API Development Tools. https://swagger.io/tools/swagger-ui/. [Consulta: 10 de Marzo de 2019].

[29] Firebase Test Lab — Firebase. https://firebase.google.com/docs/test-lab/?hl=es. [Consulta: 29 de Marzo de 2019].

[30] Torres-Sospedra, Joaquín. Analysis of Sources of Large Positioning Errors in Deterministic Fingerprinting. *Sensors*, 2017.

[31] Compara tu salario. https://tusalario.es/salario/comparatusalario#/. [Consulta: 19 de febrero de 2019].

[32] WebStorm: The Smartest JavaScript IDE. https://www.jetbrains.com/webstorm/. [Consulta: 21 de Febrero de 2019].

[33] Ubik Geospatial Solutions. http://www.ubikgs.com/. [Consulta: 22 de Enero de 2019].

Anexo A

Texto de ayuda de la aplicación

Cercador de llibres i cabines.

Per cercar un llibre, escriviu les paraules clau (p. ex. part del títol o nom de l'autor), o el topogràfic del llibre (p. ex. PN1992). Per cercar una cabina, escriviu la paraula cabina i el número de la cabina (p. ex. cabina 112). El resultat de la recerca es mostra com una llista de llibres o cabines coincidents amb els termes de cerca.

Els elements disponibles tenen a la seva esquerra la icona , i els que no estan disponibles tenen la icona .

Seleccioneu un resultat disponible per ubicar-lo al mapa. En pantalles petites (p. ex. en mòbils), es pot canviar entre la llista de resultats i el mapa mitjançant les icones i .

Baix a la dreta, el component mostra la planta de l'element seleccionat i permet visualitzar altres plantes.

El component format per y permet calcular rutes des d'un origen especificat fins l'element de la cerca seleccionat. Utilitzeu per especificar l'origen de la ruta i per esborrar l'origen o destinació de la ruta. La part de la ruta corresponent a la planta visualitzada es mostra en vermell i la resta en gris.