Generador interactivo de instrucciones de guía sobre plataformas móviles

Departamento de ingeniería del software e inteligencia artificial
Facultad de Informática
Universidad Complutense de Madrid

Víctor Gutiérrez Rodríguez
Juan Diego Lozano Martín
Víctor Manuel Pose Murga

Directores
Gonzalo Méndez Pozo
Pablo Gervás Gómez-Navarro

Madrid a 23 de junio de 2014

Generador interactivo de instrucciones de guía sobre plataformas móviles

Proyecto de fin de carrera de ingeniería en informática

Departamento de ingeniería del software e inteligencia artificial

Facultad de informática

Universidad Complutense de Madrid

Autores

Víctor Manuel Pose Murga
Víctor Gutiérrez Rodríguez
Juan Diego Lozano Martín

Profesores directores

Gonzalo Méndez Pozo
Pablo Gervás Gómez-Navarro

Curso 2013 / 2014

AUTORIZACIÓN

Se autoriza a la Universidad Complutense de Madrid a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a sus autores, tanto la propia memoria, como el código, los contenidos audiovisuales incluso si incluyen imágenes de los autores, la documentación y/o el prototipo desarrollado.

_____

Víctor Gutiérrez Rodríguez
Juan Diego Lozano Martín
Víctor Manuel Pose Murga

AGRADECIMIENTOS

Ha sido un año muy largo y duro, con muchos quebraderos de cabeza para sacar el proyecto adelante, pero finalmente se ha conseguido y con bastante éxito.

Por ello queremos agradecer a nuestros compañeros sus buenos consejos en los momentos más complicados y su ayuda a la hora de resolver ciertos problemas.

Al personal de la Facultad de Informática por facilitarnos la movilidad por todo el edificio.

En especial, agradecer a nuestras familias por todo el sacrificio realizado desde que comenzamos la carrera.

Finalmente a Pablo Gervás y en especial a Gonzalo Méndez por su dedicación, apoyo, enorme paciencia y sabios consejos.

Gracias a todos.

RESUMEN

Hoy en día existen múltiples aplicaciones para dispositivos móviles capaces de guiarnos de un punto cualquiera de la ciudad a otro, sugiriendo distintos itinerarios en función del tráfico o medio de transporte.

Pero, ¿qué ocurre si en lugar de encontrarnos en una ciudad, nos encontramos en el interior de un edificio? Todos nos hemos vuelto locos alguna vez buscando una consulta en un hospital, un pasillo en un supermercado o un aula en una facultad. Hay que buscar alternativas allí donde el sistema de posicionamiento global (GPS) no llega.

Así es como nace este proyecto de fin de carrera, que viene desarrollándose desde hace unos años en la Facultad de Informática de la Universidad Complutense de Madrid. Utilizando una tecnología tan extendida hoy en día como es el Wi-Fi, es capaz de estimar nuestra posición dentro del edificio, calcular la ruta mínima hasta nuestro destino y generar unas intuitivas instrucciones que nos guíen hasta el mismo.

Con las mejoras implementadas, la aplicación es capaz de guiarnos por todo el edificio, reconociendo el cambio de planta. Y no sólo por el de la Facultad de Informática, ya que gracias a una completa reestructuración y reorganización del sistema, puede hacerse fácilmente extensible a cualquier otro edificio que deseemos.

Gracias al reconocimiento de voz basta con decir el lugar al que queremos ir y el nuevo sistema de generación de instrucciones nos guiará paso a paso, mediante unas simples, intuitivas y eficaces instrucciones que evitarán que nos perdamos en nuestro recorrido.

Todo lo que se requiere por tanto, es algo tan común hoy en día como una red Wi-Fi y un móvil con sistema operativo Android, para dar solución al problema del posicionamiento y guía en interiores.

Palabras clave:

- Posicionamiento en interiores
- Generación de instrucciones
- Tecnología Wi-Fi
- Android

ABSTRACT

Nowadays there are multiple applications for mobile devices capable of guiding us from any point of the city to another, suggesting different routes depending on traffic or transportation.

But what if instead of being in a city, we are inside a building? We all have gone crazy sometime looking for a room in a hospital, an aisle in a supermarket or a classroom in college. We must find alternatives where global positioning system (GPS) fails .

That is how this project was born. It has been under development for a few years at the Facultad de Informática, Universidad Complutense de Madrid. Using a widespread technology nowadays such as wi-fi, we are able to estimate our position within the building, calculate the minimum path to our destination and generate some intuitive instructions to guide us.

With the implemented improvements, the application is able to guide us through the whole building, recognizing the change of the floor. But not only inside the Facultad de Informática, thanks to a complete restructuring and reorganization of the system, it can be easily extended to any other building.

Thanks to the voice recognition system, the user just have to say the place he wants to go and the new instructions generation system will guide step by step through some simple, intuitive and effective instructions that will prevent him to get lost in his path.

All that is required therefore to solve the problem of positioning and guidance indoors is something as common as a Wi -Fi network and a mobile phone with Android operating system.

Keywords:

- Indoor Positioning
- Instructions generation
- Wi-Fi Technology
- Android

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO I

INTRODUCCIÓN

1.1 MOTIVACIÓN

A raíz de la llegada, y posterior masificación, del GPS (Global Positioning System)[3] surge la problemática de la localización en interiores. Este nuevo sistema proporciona una ubicación precisa (con unos pocos metros de margen de error) y continúa en exteriores. Así su uso se extendió rápidamente a todo tipo de sistemas de transporte (barcos, aviones, automóviles). Pasó de esta forma a ser un sistema de uso cotidiano, el cual resultaba muy práctico para resolver ciertas necesidades diarias de gran cantidad de gente.

Su inclusión en los teléfonos inteligentes, o smartphones, potenció su uso personal en cualquier tarea de ubicación. Sin embargo, cuando se trataba de interiores esto fallaba. La alta precisión necesaria y la poca cobertura (el sistema GPS necesita visibilidad directa entre el dispositivo a ubicar y los satélites) hicieron que su uso fuera inviable para esta tarea.

Así se requieren formas distintas para conseguir una ubicación precisa dentro de edificios. Mirando su infraestructura se puede observar que la práctica mayoría de ellos disponen de un sistema de redes inalámbricas que, hoy en día, solo sirven para la conexión entre equipos informáticos y el acceso a internet. Sin embargo también es verdad que gran parte de los teléfonos móviles actuales poseen en su interior una tarjeta de red inalámbrica Wi-Fi. Por ello se plantea el uso de estas redes para el posicionamiento en interiores, creando un sistema viable y capaz de aprovechar al máximo las posibilidades que la tecnología y las infraestructuras ya existentes nos otorgan.

La creación de un sistema de posicionamiento y guía en interiores ha de ser una prioridad. Es una aplicación que puede llegar a ser fundamental en nuestro día a día, al igual que lo es el GPS, para ayudar a los usuarios a moverse con facilidad por grandes edificios que desconocen

1.2 ESTRUCTURA DEL DOCUMENTO

Este documento se encuentra organizado en diferentes capítulos. Este primer capítulo contiene una introducción y el contexto en el que nos encontramos.

El segundo capítulo, el **Estado de la cuestión**, recoge la situación actual de los sistemas de localización en interiores, así como las diferentes compañías que trabajan en proyectos referentes a ello. Además de los diferentes proyectos de fin de carrera en los que nos hemos apoyado para la realización de éste.

Tras describir el entorno en que se desarrolla este proyecto y desde dónde partimos, se pasa al tercer capítulo, **Objetivos**, donde se establecen y delimitan claramente unos objetivos a cumplir en el presente proyecto.

El cuarto capítulo, **Características del Proyecto**, muestra una visión del proyecto en sí, se describe la arquitectura modular y funcional del proyecto, entrando en detalle en cada uno de sus componentes.

El quinto capítulo, **Gestión del Proyecto**, se especifican los riesgos encontrados en la elaboración del proyecto. Así como la planificación del trabajo que se ha llevado a cabo.

El sexto capítulo, **Diseño e Implementación**, explica con gran detalle cada una de las partes que conforman el sistema para lograr su funcionalidad. Además, se recoge una visión general y evolución del sistema operativo Android, seguido de un estudio de la situación actual y aplicaciones que podemos encontrar de los sistemas por voz,

En el séptimo capítulo, **Resultados y Trabajos Futuros**, se evalúan los resultados obtenidos, tanto los positivos como los negativos. Se llega a unas conclusiones sobre el proyecto y se establece una línea de trabajo futuro.

A continuación se encuentran los Apéndices. El **Apéndice A** recoge un manual de instalación de la aplicación en los distintos entornos así como la puesta en funcionamiento del mismo. El **Apéndice B** recoge un manual de usuario de la aplicación. Y el **Apéndice C** es una guía para la adaptación de la aplicación para poder ser utilizada en otros edificios.

Por último, se encuentra la **Bibliografía y referencias**. Donde se recogen las referencias utilizadas para la realización de este proyecto.

CAPÍTULO II

ESTADO DE LA CUESTIÓN

2.1 SISTEMAS DE LOCALIZACIÓN EN INTERIORES

2.1.1 Introducción

En esta sección se hablará sobre los diferentes sistemas de localización existentes así como de su adecuación para usarlos en la aplicación que este trabajo aborda. En todos los casos, se pensó y valoró su uso a través de un dispositivo móvil con el fin de localizar al usuario dentro de un edificio. El principal problema que se encontró fue la precisión que la mayoría de sistemas proporcionaban.

2.1.2 Posicionamiento GPS

El posicionamiento GPS hace referencia al posicionamiento usado mediante la tecnología GPS o *Global Positioning System*[3]. Como tal, el posicionamiento utilizado en la tecnología GPS es la triangulación, tratado en la siguiente sub sección, por lo que aquí nos ceñiremos a hablar sobre el sistema de posicionamiento global.

Este sistema está formado por una red constante de veinticuatro satélites en órbita alrededor de la Tierra que se usan para calcular la posición de un dispositivo mediante triangulación. La exactitud con la que el GPS es capaz de calcular dicha posición varía según el sistema GPS usado. Así podemos hablar de aproximadamente quince metros de error en un GPS como los de los dispositivos móviles en el 95% del tiempo, o de un error de un metro en sistemas que usen como apoyo otros como WAAS (*Wide Area Augmentation System* [7]), EGNOS (*European Geostationary System Navigation Overlay Service* [8]) o MSAS (*Multi-functional Satellite Augmentation System* [9]).

Este sistema de posicionamiento es inviable cuando se trata de localización en interiores, ya que se requiere una conexión directa, sin barreras arquitectónicas de por medio, entre el dispositivo receptor y los satélites.

2.1.3 Posicionamiento mediante Triangulación

Por triangulación entendemos el método de posicionamiento que usan los sistemas GPS para localizar la posición exacta del dispositivo en cuestión. Este método necesita de, por lo menos, tres satélites localizados que se usarán como puntos de referencia (de ahí su nombre). Por lo general, esto no suele constituir un problema ya que siempre habrá ocho satélites dentro del "campo visual" de cualquier receptor GPS. Las posiciones de los satélites usados para la triangulación se saben de antemano gracias a una base de datos.

Hay que remarcar que la precisión de este método se verá mermada por encontrarnos bajo diversas superficies, tanto túneles como tejados u otros objetos similares. La mayor precisión se conseguirá en el exterior, lejos de obstáculos que reduzcan la capacidad de captar las señales enviadas por los satélites.

El funcionamiento de este método es el siguiente:

Cuando el receptor detecta el primer satélite, se genera una esfera a su alrededor cuyo radio será la distancia desde el receptor hasta dicho satélite. De este modo, el receptor se encontrará en un punto de la superficie de esa esfera, aún por determinar.

Repetimos el proceso con otro satélite. Al crearse esa segunda esfera, el dispositivo receptor se encontrará en alguno de los puntos de corte de ambas esferas, por lo que el resto de puntos se descartan.

De nuevo, se utiliza un tercer satélite de modo que se crea una nueva esfera que cortará a las anteriores. De este modo, con el corte de las tres esferas, y teniendo en cuenta que el dispositivo se encuentra en la superficie terrestre, tendremos el punto concreto buscado.

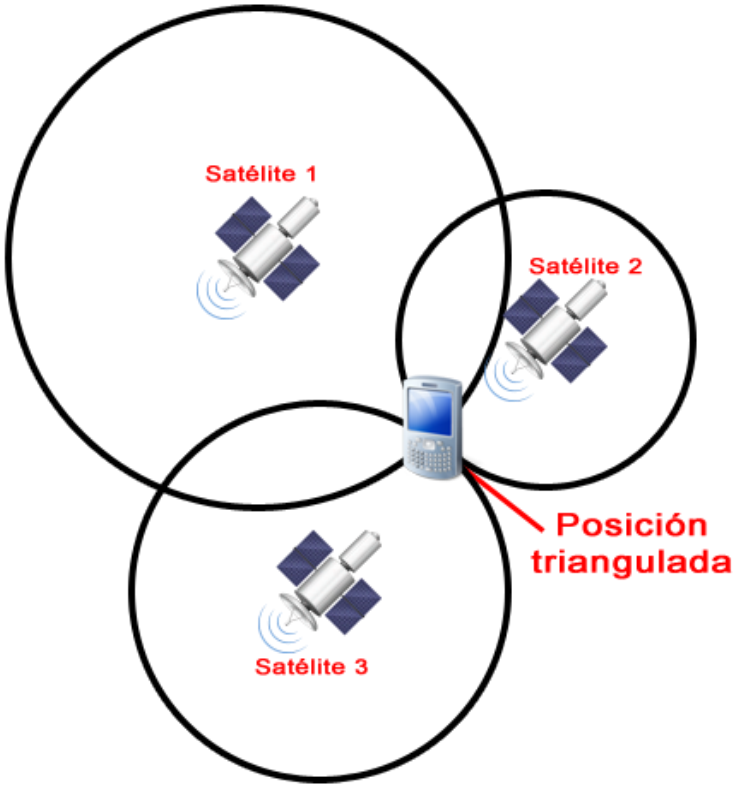

Figura 1: Esquema de triangulación

En caso de querer conocer la altitud, bastará con usar un cuarto satélite como referencia y repetir el proceso [4].

Este tipo de localización tiene la misma problemática que los GPS cuando se trata de interiores.

2.1.4 Posicionamiento Wif-fi

El sistema Wi-Fi es una red de tipo WLAN, es decir, una red inalámbrica de área local (*Wireless Local Area Network*). Este tipo de redes existen bajo el estándar IEEE 802.11, existiendo diversos tipos a su vez: IEEE 802.11b, IEEE 802.11g e IEEE 802.11n. Estos estándares funcionan a través de la banda de 2.4 GHz con velocidades de hasta 11 MBit/s, 54 MBit/s y 300 MBit/s, respectivamente. Al trabajar en la frecuencia de 2.4 GHz, pueden existir interferencias con otras tecnologías, como el Bluetooth, por lo que se habilitó un nuevo estándar que trabaja en la frecuencia de 5 GHz, el IEEE 802.11a.

En este tipo de redes se hace indispensable la separación de dos tipos de dispositivos: dispositivos de distribución o red, como routers, puntos de acceso y repetidores, y dispositivos

terminales o receptores. Usar Wi-Fi como sistema de localización en interiores supone que hay alguna antena o dispositivo de red Wi-Fi al que el dispositivo móvil se pueda conectar o, por lo menos, pueda captar su señal. Puesto que hoy en día pocas son las zonas en las que un edificio no tenga algún tipo de señal Wi-Fi fija, se puede tomar esta premisa como cierta.

La idea de localización a través de señales Wi-Fi es la siguiente: un dispositivo capta la señal Wi-Fi de un dispositivo de red. Una vez conocida su dirección MAC, el dispositivo se conecta a una base de datos en la que hay almacenadas diferentes coordenadas cartesianas asociadas a diferentes señales MAC (en caso de haberlas) y la intensidad que la señal Wi-Fi del dispositivo de red alcanza en ese punto. De este modo, se puede interpolar entre la intensidad captada por el dispositivo y la guardada en la base de datos para conocer la posición en la que se encuentra el dispositivo de red [5].

Esta localización es suficientemente fiable como para poder localizar una posición en el interior de un edificio con un error relativamente pequeño (entre cero y un metro). Además, no necesita de una conexión permanente sino que, una vez conocida la intensidad de la señal y la dirección MAC, se puede calcular la posición. Éstos son motivos más que suficientes para hacer del sistema de localización por Wi-Fi el adecuado para la aplicación que aborda este trabajo.

2.1.5 Posicionamiento Bluetooth

El Bluetooth es una red WPAN, es decir, una red inalámbrica de área personal (*Wireless Personal Area Network*). Este tipo de red permite la transmisión de voz y datos entre diversos dispositivos, usando un enlace por radiofrecuencia en la banda de 2.4 GHz.

Este tipo de conexión destaca, sobretodo, por su bajo consumo, tanto en los receptores como en los transmisores. El principal problema de esta red radica en la velocidad de transmisión respecto a otro tipo de redes inalámbricas y en su rango de alcance.

Hablando de las últimas versiones, Bluetooth v4.0, nos encontramos con velocidades teóricas de hasta 24 Mbit/s con un radio de alcance medio de 10 metros. Hay que decir que esta versión de Bluetooth no usa dicha tecnología en todo su conjunto sino que se basa en Wi-Fi para la transmisión de datos como tal, dejando la conexión Bluetooth para la negociación de la conexión.

De este modo, centrándonos en la tecnología Bluetooth clásica, dejando a un lado el apoyo Wi-Fi, este tipo de conexiones necesitarían un interceptor cada 10 metros que consiguiese establecer una conexión con internet de modo que pudiesen transmitirse los datos pertinentes.

Así mismo, la transmisión de los datos desde/hasta el dispositivo en cuestión, se llevaría a cabo bajo una velocidad práctica de 2.1 Mbits/s, bastante más lejos de las velocidades alcanzables a través de tecnología Wi-Fi.

En conclusión, aunque el Bluetooth ofrece un muy buen consumo respecto a otras tecnologías, sus problemas en cuanto a infraestructura, velocidad de transferencia y la necesidad de interceptores, hacen que no sea una tecnología apta para el desarrollo de este proyecto.

2.2 APLICACIONES SIMILARES

2.2.1 Google Indoor Maps[1]

La nueva aplicación de mapas de Google es capaz de posicionarnos en interiores de grandes superficies.

Para ello es necesario haber almacenado el mapa del edificio en la aplicación, por lo que por el momento sólo está disponible en algunos aeropuertos o grandes centros comerciales de Estados Unidos o Japón.

Una vez tenemos almacenado el mapa, el posicionamiento se realiza mediante la red wi-fi, captando la intensidad de la señal en diferentes puntos. Como ocurre en este proyecto, este posicionamiento no resulta todo lo preciso que se desearía.

La aplicación es capaz de distinguir la planta en la que nos encontramos, además de nuestra orientación, gracias a la brújula y los sensores de movimiento de nuestro terminal.

[1] https://www.google.com/maps/about/partners/indoormaps/

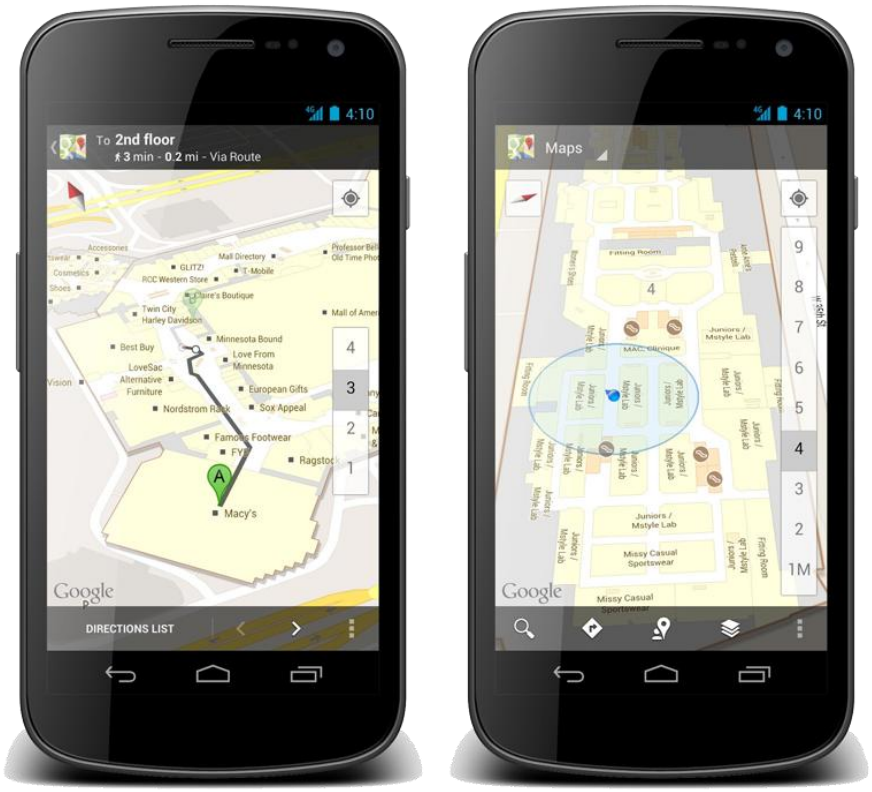

Figura 2: Google Indoor Maps

2.2.2 WifiSLAM[2]

Recientemente Apple ha comprado la compañía WifiSLAM. Esta compañía lleva años desarrollando una aplicación de localización en interiores utilizando también el posicionamiento mediante triangulación de las señales wi-fi, lo que proporciona un error de unos 2.5 metros de posicionamiento.

Con WifiSLAM no es necesario realizar un mapeo previo del edificio. Para recabar la información del mismo, utiliza las propias trayectorias que son grabadas por los usuarios. Así construye una gran base de datos de mapas y caminos del edificio[3].

[2] https://angel.co/wifislam

[3] http://thenextweb.com/apple/2013/03/26/what-exactly-wifislam-is-and-why-apple-acquired-it/

2.2.3 BeeMe[4]

BeMee es un sistema de localización y navegación en interiores que utiliza tecnología Bluetooth 4.0. Permite analizar, mediante técnicas de robótica, la posición y ocupación de personas dentro de grandes infraestructuras. Gracias a esta información se podrán tomar medidas para mejorar la eficiencia y productividad en grandes organizaciones usando para ello las últimas tecnologías inalámbricas.

Es capaz de determinar la posición de una persona en el interior del edificio, utilizando algoritmos en el campo de la robótica. Los datos extraídos son convertidos en información, creando así un canal de comunicación personalizado entre la persona y el entorno que le rodea. Requiere de la instalación de unos dispositivos electrónicos en diferentes puntos del edificio para que el algoritmo pueda determinar la posición del usuario.

Está creado por Jorge García Bueno y Alejandro Martín, dos emprendedores gestados en la Universidad Carlos III de Madrid, que se han unido a Marco Doncel para crear este sistema.

2.2.4 Satec: Localización de interiores: RFID[5]

SATEC es uno de los principales integradores en bases de datos corporativas, integradas con cualquiera de las tecnologías que se usan actualmente para la localización de personas u objetos en entornos de interior dónde el GPS no llega. Se usan las tres tecnologías punteras en este momento: WiFi, RFID y Zigbee[6].

Estas etiquetas inteligentes pueden utilizarse, por ejemplo, en las camillas de un hospital para identificar no solo la camilla libre u ocupada sino también su lugar de destino. También puede servir para identificar material necesario para un hospital indicando incluso el contenido o cualquier otro dato que deba almacenarse e integrarse posteriormente con la base de datos del hospital para una gestión rápida que facilite la toma de decisiones y mejore la productividad del mismo.

[4] http://www.bemee.es/
[5] http://www.satec.es

2.3 TRABAJO PREVIO

Para la realización de este proyecto se han requerido varios proyectos englobados en el Proyecto MILES (TIN2009-14659-C03) del departamento de Ingeniería del Software e Inteligencia Artificial de la Facultad de Informática de la UCM. De ellos se ha conseguido utilizar parte de su funcionalidad para adaptarla a la nueva funcionalidad de este proyecto.

2.3.1 Avanti: Sistema de asistencia a la evacuación de incendios 2009-2010

Proyecto de Sistemas Informáticos realizado por Enrique López Mañas, Francisco Javier Moreno, Javier Plá Herrero [1].

El fin de este proyecto era la creación de un sistema de evacuación de incendios. Una completa e intuitiva plataforma que pudiera ser utilizada en los ensayos de evacuación.

Combinaba la tecnología de posicionamiento wi-fi para el posicionamiento en interiores. La predicción del movimiento gracias a los sensores de los terminales Android. El acelerómetro del terminal se utilizó para recabar información que permitiera estimar el movimiento del usuario, basado en estadísticas y mediciones previas, para así poder guiar al usuario en zonas donde no llega una buena señal wi-fi.

La Realidad Aumentada para poder mostrar la información en el propio entorno y verla desde la pantalla del dispositivo móvil, lo que le da un gran atractivo visual. De esta manera, se intenta contribuir a la informatización de los ensayos de incendios, con el fin de analizar posibles fallos en los protocolos de seguridad a seguir.

La funcionalidad del proyecto AVANTI requerida para la elaboración del presente proyecto ha sido la referente al posicionamiento en interiores utilizando tecnología wi-fi. Con ello se ha podido establecer la posición del usuario dentro del edificio de manera automática y una vez establecido un destino, poder guiarle controlando en todo momento su posición para garantizar que sigue el camino correcto.

2.3.2 Sistema de guía por voz en interiores 2012-2013

Proyecto de Trabajo de Fin de Grado realizado por Mariana Martín- Calderín de la Villa [2].

Este proyecto establece las bases para la creación de un sistema de guía en interiores.
Se realizaron cambios significativos respecto al proyecto AVANTI. Como es la adaptación del proyecto a la nuevas versiones Android, que incluyen mejoras en los sistemas del acelerómetro y la brújula.

Utilizando el mismo sistema de posicionamiento por wi-fi que venía desarrollándose en proyectos previos, se consiguió calcular una ruta mínima entre dos puntos para poder guiar al usuario por la primera planta de la Facultad de Informática. Generando un listado de instrucciones basadas en nuestra orientación y distancia en metros.

Además, para facilitar al usuario el manejo de la aplicación, se implantaron los sistemas de reconocimiento y reproducción de voz, utilizados para captar el destino deseado y la lectura de las instrucciones generadas

CAPÍTULO III

OBJETIVOS

Este proyecto se enmarca dentro del Proyecto Miles (TIN2009-14659-C03) del Departamento de Ingeniería del Software e Inteligencia Artificial de la Facultad de Informática de la UCM y tiene como objetivo integrar la funcionalidad realizada en otros proyectos anteriores, así como la implementación de nueva funcionalidad para lograr la realización final de un sistema de guía por voz en interiores.

Se desarrollará un sistema capaz de localizar personas en el interior de un edificio, utilizando como infraestructura la red inalámbrica del propio edificio. Para el desarrollo de la aplicación se utilizará Android, sistema operativo que nos permitirá el reconocimiento de voz, para que el usuario pueda indicar el lugar deseado, y la reproducción de voz, que se utilizará, una vez calculada la trayectoria, para indicar al usuario la ruta que ha de seguir.

El presente proyecto tiene como objetivo la ampliación de estas funcionalidades, para desarrollar un completo sistema de guía en interiores.

Partimos de un sistema que es capaz de guiarnos únicamente por la primera planta de la Facultad de Informática, por lo que el **primer objetivo** es estudiar y crear un sistema capaz de reconocer en qué planta nos encontramos e indicarnos en cualquier momento si tenemos que subir o bajar para continuar nuestro camino, para poder así guiarnos por todo el edificio de la facultad.

El **segundo objetivo** que se establece, es la completa reestructuración en el sistema que maneja la información relativa a la estructura del edificio. Se trata de un sistema genérico que permita a la aplicación ser utilizada no sólo en la Facultad de Informática, si no en cualquier edificio que se desee.

Como **tercer objetivo** se tiene la completa renovación del sistema de generación de instrucciones, para hacerlas interactivas. Estas instrucciones se darán en un lenguaje más natural y utilizando elementos de referencia del entorno, para facilitar su entendimiento. Se proporcionarán paso a paso para evitar que el usuario se pierda.

CAPÍTULO IV

CARACTERÍSTICAS DEL PROYECTO

4.1 INTRODUCCIÓN

En este apartado describiremos todas las características del proyecto en su conjunto. A grandes rasgos está dividido en dos módulos principales, conectados entre sí. Estos son un programa que genere las instrucciones, que se ejecutará en un **servidor**, una aplicación **cliente,** ejecutada en un terminal Android, que proporcionará al usuario final las funciones deseadas (posicionamiento, búsqueda de rutas…)

Esta división es necesaria para liberar a la aplicación cliente de una gran carga de trabajo. Los cálculos más pesados como manejar la estructura del edificio, calcular la ruta mínima o generar las instrucciones de guía, se realizan en el servidor, que posee mayor capacidad de procesamiento.

En la aplicación cliente se podrá establecer el destino al que se desea ir, ya sea por voz o por escrito. La idea original de usar sólo voz se vio extendida para poder introducir texto directamente, haciendo la aplicación más accesible. Cuando el servidor reciba la información y genere las instrucciones, estas se recibirán en el dispositivo, reproduciéndolas por voz.

El algoritmo usado para el posicionamiento bajo redes Wi-Fi es el algoritmo de los k-vecinos más cercanos, o k-nn[15] por sus siglas en inglés, el cual se ha reutilizado partiendo de los proyectos anteriores: “AVANTI: Sistema de asistencia a la evacuación de incendios” [1] y “Sistema de guía por voz en interiores” [2].

Por otro lado, en el servidor se recoge la información de las diferentes plantas del edificio por el que el usuario se va a mover, a través de diferentes archivos preparados para tal fin. De esta forma, cuando reciba la información de la aplicación cliente, de dónde a dónde se quiere ir, se

consultarán estos datos para realizar una búsqueda y obtener el mejor camino posible desde el inicio al final, entendiendo como “el mejor camino” aquel que permita llegar de una forma más directa. Esta ruta será encapsulada en una serie de instrucciones que el usuario de la aplicación móvil pueda entender.

Cliente y servidor seguirán en contacto para recalcular, en caso necesario, una nueva ruta hasta el destino. Esto puede deberse a que el usuario se haya desviado de la ruta o a que haya llegado a algún punto crítico de la misma, como un cruce de pasillos, un aula o similares. Igualmente, cada pocos segundos, la aplicación cliente enviará información sobre su estado actual al servidor para que éste pueda volver a realizar el cálculo mencionado.

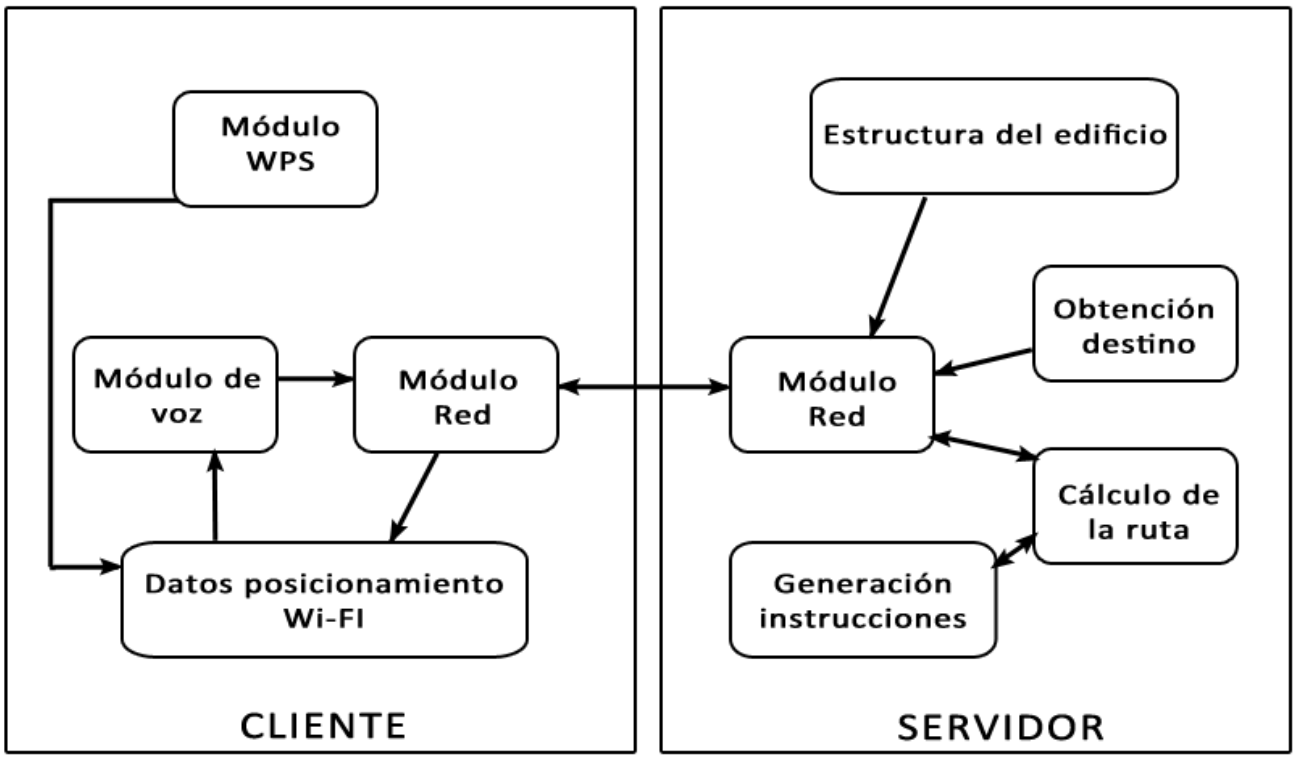

Figura 3: Esquema modular de la aplicación

La **Figura 3** la estructura de la aplicación desde un punto de vista modular.

El cliente y el servidor se comunican mediante un módulo de red, donde se produce un intercambio de datos, necesario para el cálculo de la posición actual y la ruta a seguir.

Los datos del posicionamiento Wi-Fi son cargados en el cliente, y el módulo WPS se encarga de estimar nuestra posición, generando las coordenadas correspondientes.

El módulo de voz se encarga de recibir el destino deseado por el usuario y la posterior lectura de las instrucciones recibidas.

Con todos estos datos enviados al servidor, éste carga la estructura del edificio y la posición del destino deseado. A continuación, calculará la ruta mínima desde la posición actual del usuario a ese destino y mediante la generación de unas instrucciones interactivas, se irá guiando al usuario.

4.2 CLIENTE

Para desarrollar la aplicación cliente nos basamos en el proyecto "Sistema de guía por voz en interiores" [2] que a su vez extiende las funcionalidades de "AVANTI: Sistema de asistencia a la evacuación de incendios" [1]. En la última versión implementada de este proyecto el sistema de guía funcionaba únicamente en la planta primera de la facultad de informática. Nuestra tarea en el cliente ha consistido básicamente en facilitar posibles extensiones permitiendo que el sistema admita más de una planta, además del cambio de edificio completo.

Entre las ventajas de estas mejoras se encuentra la extensibilidad que permitirá usar este sistema de guiado ya no solo en la Facultad de Informática de la Universidad Complutense sino en cualquier edificio con el simple requisito de que posean una red de antenas WiFi, el cual es actualmente superado en gran cantidad de edificios tanto públicos (universidades, bibliotecas, ministerios...) como privados (centros comerciales, bloques de oficinas, etc).

Sin embargo, esta generalización del sistema supone un desafío ya que gran parte de los nuevos usuarios pueden no estar familiarizados con la informática ni el mundo de la programación. Por ello debemos facilitar el trabajo que supone la puesta en funcionamiento del sistema y su uso cotidiano.

4.3 SERVIDOR

Debido a la naturaleza del proyecto, se hace indispensable la existencia de un servidor en el que se ejecuten las partes más pesadas del cálculo para no sobrecargar el dispositivo móvil así como para tener un punto al que acceder desde cualquiera de dichos dispositivos. Las características del servidor están indicadas en el capítulo sobre la implementación.

Como se ha dicho anteriormente, en el servidor se ejecuta una única aplicación que se encarga de:

- Permanecer a la escucha para cualquier conexión de un dispositivo móvil
- Calcular la mejor ruta desde donde se encuentre el dispositivo hasta su destino
- Enviar la información de vuelta al dispositivo

Visto así, la tarea que realiza la aplicación del servidor no parece un trabajo excesivo. Sin embargo, hay que tener en cuenta que se podría querer desplazar desde una punta a otra del edificio, lo que podría implicar un desplazamiento a través de diferentes pasillos, escaleras o aulas. Además, cada poco tiempo, el dispositivo refrescará su posición, mandando las nuevas coordenadas al servidor. En caso de habernos salido del recorrido preestablecido, se recalculará el mismo. Es decir, la búsqueda óptima del camino a recorrer, requiere un tiempo y unos recursos que bien pueden eliminarse del dispositivo, disminuyendo en gran medida el consumo de energía del mismo.

Además de encontrar el camino mínimo entre la posición origen del dispositivo y la destino, la aplicación servidor se encarga también de generar las instrucciones a seguir por el usuario. Dichas instrucciones se generan una vez conocido el recorrido de la siguiente forma:

- Si hay que recorrer un camino largo, como pueda ser de un extremo a otro de un pasillo, el mismo se dividirá en dos, indicando la instrucción un movimiento recto hasta a algún punto característico (extintor, banco, fuente...)
- Pasado el anterior filtro, si existe un cambio de sentido, se indicará el camino hasta el mismo y se indicará el cambio de sentido (girar, subir escaleras...)
- Una vez llegado hasta el hito indicado por los anteriores filtros, se “lanzará” la siguiente instrucción

Este proceso se repite hasta que se llegue al destino.

CAPÍTULO V

GESTIÓN DEL PROYECTO

5.1 ANÁLISIS DE REQUISITOS

Vamos a hacer un estudio de los requisitos del proyecto, tanto para la aplicación cliente como para el servidor. Se analizarán los requisitos funcionales así como los no funcionales.
Los requisitos funcionales describen los servicios que proporciona el sistema (funciones), la respuesta del mismo ante determinadas entradas y su comportamiento ante diferentes situaciones. Mientras que los requisitos no funcionales especifica los criterios que se deben usar para juzgar el funcionamiento de un sistema, en lugar de un comportamiento específico.

5.1.1 Cliente: Aplicación móvil

Requisitos funcionales

- Debe ser capaz de acceder a bajo nivel a la capa de comunicación: para poder lograr el posicionamiento Wi-Fi, la aplicación necesitará acceder a datos de bajo nivel sobre los puntos de acceso con el fin de realizar la triangulación necesaria.
- Debe ser capaz de procesar la entrada, el destino, que el usuario indique por voz al sistema: para lograr el sistema por voz, la aplicación deberá realizar el reconocimiento de voz correctamente sobre el destino que indique el usuario
- Debe ser capaz de reproducir la ruta mínima al usuario para llegar al destino: igual que el requisito anterior para que el sistema de voz funcione correctamente, la aplicación debe ser capaz de reproducir la ruta mínima calculada al usuario.
- Debe ser capaz de posicionar al usuario en el mapa con la máxima precisión posible: para lograr un control más exacto de la situación actual del usuario

Requisitos no funcionales: Software

- Versión Android desde la 2.3 hasta la 4.2

Requisitos no funcionales: Hardware

- Se trabajará con un Samsung Galaxy S4 y un Samsung Galaxy Note 3. Se requiere un terminal que disponga de las siguiente características como mínimo: 1Ghz de procesador, 512MB RAM y 3Mb de espacio libre.

5.1.2 Servidor

Requisitos funcionales

- Debe proporcionar conexión en tiempo real con el terminal móvil
- Debe calcular y generar la trayectoria mínima que debe seguir el usuario, establecidos un origen y un destino: la funcionalidad principal de este proyecto es la generación de las rutas y las trayectorias que el usuario debe seguir, por lo que es un requisito trivial para el éxito del proyecto.
- Debe ser capaz de generar una serie de instrucciones en lenguaje natural a partir de la ruta calculada: al igual que el anterior, se convierte en un requisito indispensable para el éxito del proyecto, pues necesitamos que el usuario obtenga la lista de instrucciones para poder llegar al destino.
- Debe hacer sencillo una posible ampliación del proyecto: en un futuro puede ser posible una ampliación del proyecto o añadir nueva funcionalidad, por lo que se debe proporcionar una arquitectura lo más modular que sea posible. Se intentará definir los módulos lo mejor posible y separar lógicamente cada capa de acceso para facilitar la tarea de los futuros programadores.

Requisitos no funcionales: Software

- El sistema deberá funcionar sobre la versión 7 de JAVA. Java Runtime Environment (JRE) v.7.

Requisitos no funcionales: Hardware

- Se requiere un PC de gama media con al menos 1GB de RAM.

5.2 GESTIÓN DE RIESGOS

El análisis de riesgos informáticos es un proceso que comprende la identificación de activos informáticos, sus vulnerabilidades y amenazas a los que se encuentran expuestos así como su probabilidad de ocurrencia y el impacto de las mismas, a fin de determinar los controles adecuados para aceptar, disminuir, transferir o evitar la ocurrencia del riesgo.

Analizaremos los riesgos de una manera proactiva, esto es, identificándolos antes de que puedan ocurrir y proponiendo alternativas para evitarlos, así como posibles soluciones si finalmente se llegan a producir.

Una vez no se haya podido evitar un riesgo, lo identificaremos, así como sus causas y se asignarán recursos para evitar que suponga un problema para el correcto desarrollo del proyecto.

Para cada uno de los riesgos que a continuación se presentan, se realiza en un primer momento un listado recogiendo el nivel de criticidad que presenta cada riesgo en base a la relación entre la probabilidad y la severidad de los mismos. Este nivel de criticidad se toma en base a la técnica SQAS-SEI [14] dónde se establecen los siguientes valores:

Probabilidad/ Severidad	Frecuente	Probable	Ocasional	Remoto	Improbable
Catastrófico	IN	IN	IN	A	M
Crítico	IN	IN	A	M	B
Serio	A	A	M	B	T
Menor	M	M	B	T	T
Insignificante	M	B	T	T	T

IN: Intolerable A:Alto B:Bajo T:Tolerable

Tabla 1: Técnica SQAS-SEI

5.2.1 Análisis de riesgos

Resumimos los riesgos encontrados en la siguiente tabla, de acuerdo a estos parámetros:

- Nombre
- Probabilidad
- Severidad
- Nivel de Criticidad

Quedan organizados en función de su nivel de criticidad. Los riesgos más graves serán aquellos de nivel de criticidad más alto:

Nombre	Probabilidad	Severidad	N. Criticidad
Abandono de un integrante del equipo	Probable	Crítica	Intolerable
Tiempo de dedicación escaso en exámenes o vacaciones	Frecuente	Crítica	Alto
Falta de comprensión del trabajo previo	Probable	Crítica	Alto
Estimaciones poco realistas	Probable	Seria	Medio
Problemas con el terminal móvil	Frecuente	Menor	Bajo
Problemas con el servidor	Ocasional	Menor	Bajo
Retraso en una entrega	Ocasional	Seria	Tolerable
Problemas con el Wi-fi	Frecuente	Menor	Tolerable

Tabla 2: Nivel de Criticidad de los Riesgos en función de la Probabilidad-Severidad

5.2.2 Estudio de los riesgos

A continuación haremos una descripción más detallada de cada riesgo teniendo en cuenta además sus posibles causas, indicadores, prevención y plan de mitigación.

Abandono de un integrante del equipo

Descripción: Uno o varios miembros del equipo abandonan el proyecto.
Probabilidad: Probable
Severidad: Crítica
Nivel de Criticidad: Intolerable
Posibles causas:

- Incompatibilidad con otras actividades

Indicadores:

- Falta de asistencia las reuniones
- No realización del trabajo asignado

Prevención:

- Comprometerse desde el inicio a realizar el proyecto
- Dar prioridad a la realización del proyecto frente a otras actividades

Plan de mitigación:

- Redistribución de las tareas asignadas a cada miembro
- Cada miembro debe conocer en profundidad la totalidad del proyecto

Tiempo de dedicación escaso en exámenes o vacaciones

Descripción: Durante un periodo de exámenes o vacaciones, se dedica menos tiempo a la realización del proyecto.
Probabilidad: Frecuente
Severidad: Crítica
Nivel de Criticidad: Alto
Posibles causas:

- Acumulación de trabajo en época de exámenes
- Falta de dedicación en periodos de vacaciones

Indicadores:

- El proyecto avanza más lento de lo esperado en estas épocas

Prevención:

- Reparto del trabajo apropiado para tener en cuenta estos parones

Plan de mitigación:

- Reorganización de la planificación
- Mayor dedicación fuera de estas épocas

Falta de comprensión del trabajo previo

Descripción: El proyecto viene realizándose en años anteriores, entender y comprender el trabajo realizado por sus autores puede llegar a ser muy costoso si no está bien explicado.
Probabilidad: Probable
Severidad: Crítica
Nivel de Criticidad: Alto
Posibles causas:

- Código no comentado apropiadamente

Indicadores:

- El proyecto avanza más lento de lo esperado en sus inicios

Prevención:

- Hablar si es posible con las personas encargadas de su realización

Plan de mitigación:

- Dedicar más tiempo al estudio del trabajo ya existente

Estimaciones poco realistas

Descripción: La planificación del trabajo y estimaciones de entregas no se corresponden con la realidad y se producen retrasos.

Probabilidad: Probable

Severidad: Seria

Nivel de Criticidad: Medio

Posibles causas:

- Falta de dedicación al proyecto
- Planificación errónea de los objetivos.

Indicadores:

- Acumulación reiterada de retrasos en las entregas

Prevención:

- Recalcular la planificación a medida que se avanza en el proyecto
- Estimaciones periódicas
- Control de objetivos cumplidos

Plan de mitigación:

- Recalcular las estimaciones
- Establecer objetivos más realistas

Problemas con el terminal móvil

Descripción: El dispositivo móvil no siempre funciona como se espera, lo que ocasiona problemas a la hora de probar la aplicación.

Probabilidad: Frecuente

Severidad: Menor

Nivel de Criticidad: Bajo

Posibles causas:

- Versión Android incompatible con la aplicación
- El terminal tiene errores desconocidos

Indicadores:

- La aplicación no se instala correctamente

Prevención:

- Disponer de varios terminales para las pruebas

Plan de mitigación:

- Usar los terminales proporcionados por la facultad

Problemas con el servidor

Descripción: Dificultades a la hora de conectar la aplicación cliente con el servidor.
Probabilidad: Ocasional
Severidad: Menor
Nivel de Criticidad: Bajo
Posibles causas:

- Los puertos de conexión están cerrados
- El servidor está caído

Indicadores:

- El cliente se detiene inesperadamente cuando intenta conectar

Prevención:

- Buscar un servidor fiable

Plan de mitigación:

- Usar el servidor proporcionado por la facultad

Retraso en una entrega

Descripción: No se ha realizado todo el trabajo previsto para una fecha concreta.
Probabilidad: Ocasional
Severidad: Seria
Nivel de Criticidad: Tolerable
Posibles causas:

- Falta de dedicación al proyecto
- Se encontraron más problemas de los previstos en la realización del trabajo

Indicadores:

- El proyecto ha avanzado menos de lo esperado

Prevención:

- Establecer objetivos realistas a corto plazo

Plan de mitigación:

- Dedicar el doble de esfuerzo para la siguiente entrega

Problemas con el Wi-Fi

Descripción: El Wi-Fi de la facultad es muy inestable e irregular en las distintas zonas, lo que causa errores a la hora de probar la aplicación o medir las intensidades.

Probabilidad: Frecuente
Severidad: Menor
Nivel de Criticidad: Tolerable
Posibles causas:

- El servicio está caído
- No hay antenas suficientes
- Zonas poco accesibles o con muchos obstáculos

Indicadores:

- El cliente no recibe señal Wi-Fi
- Se corta la señal Wi-Fi

Prevención:

- Disponer de conexión 3G

Plan de mitigación:

- Trabajar con la conexión 3G mientras no se tenga una buena señal Wi-Fi

5.3 PLANIFICACIÓN DEL PROYECTO

El proyecto se planificó en base a unos objetivos que había que cumplir. Estos consistían en la reestructuración del sistema de medidas y división del edificio, la generación interactiva y por pasos de las instrucciones y por último, ser capaces de detectar cambios de planta para generar instrucciones que guíen por todo el edificio.

Se establecieron reuniones semanales con el director del proyecto para concretar los objetivos a corto plazo, debatir los problemas que surgían y buscar soluciones.

Al ser un proyecto que viene desarrollándose desde hace varios años, la primera tarea que teníamos entre manos era el estudio de todo ese trabajo previo, entender cada parte del código, ver cuáles eran las partes que podríamos reutilizar y las que no. La falta de claridad y comentarios explicativos hizo que esta tarea nos llevase más tiempo del inicialmente previsto. Este estudio del trabajo previo nos ocupó aproximadamente un mes, hasta que pudimos empezar a desarrollar nueva funcionalidad.

Otro de los aspectos fundamentales en la planificación del trabajo es la necesidad de encontrarse dentro del edificio para poder realizar las diferentes pruebas, lo que ha resultado un lastre a la hora de trabajar desde casa.

Con todo esto, el primer objetivo a abordar fue la completa reestructuración de la organización que se hace del edificio, para poder así hacerlo extensible a otros edificios de manera muy

sencilla.
El segundo objetivo fue dar una solución al problema de cambiar de una planta a otra, para que la aplicación pudiera guiarnos por todo el edificio.

Estos dos objetivos han sido los que nos han ocupado la mayor parte del tiempo, ya que requerían mucho trabajo midiendo las intensidades wi-fi y buscando diferentes estrategias de organización de las plantas.

Por último, se abordó la tarea de renovar el generador de instrucciones, haciendo que sea más fácil de entender por el usuario y más eficaz. La duración aproximada de esta tarea fue de un mes y medio, ya en el periodo final del curso

CAPÍTULO VI

DISEÑO E IMPLEMENTACIÓN

6.1 INTRODUCCIÓN

En este capítulo se expone un diseño detallado de la funcionalidad principal del proyecto. Empezaremos hablando de las tecnologías utilizadas, como son Android y los Sistemas de voz.

Posteriormente explicaremos cómo están diseñadas e implementadas cada una de las partes que lo forman.

6.2 TECNOLOGÍAS

En este apartado hablaremos del sistema operativo para móviles Android, y el por qué se ha elegido para el desarrollo de este proyecto frente a otras alternativas.

Además explicaremos en qué consisten los sistemas de voz, empleados en este proyecto para la comunicación de la aplicación con el usuario.

6.2.1 Android

Android es un sistema operativo basado en el kernel de Linux, diseñado y desarrollado inicialmente por Android Inc. que en 2005 fue comprada por Google[6]. El sistema como tal, fue presentado públicamente en 2007, saliendo al mercado (es decir, en un dispositivo móvil) un año después, en 2008.

La filosofía tras el desarrollo de este sistema operativo radica en crear una plataforma de

[6] http://www.businessweek.com/stories/2005-08-16/google-buys-android-for-its-mobile-arsenal

desarrollo libre orientada a dispositivos móviles que posean una pantalla táctil, tales como los smartphones o las tablets. Siguiendo esta filosofía, Android no está pensado para ningún hardware de ningún fabricante en específico, lo que hace que su implementación por parte de los fabricantes, sea mucho más sencilla. Además, la fácil personalización de este sistema, ha hecho que sea elegido por la inmensa mayoría de fabricantes móviles, con Samsung a la cabeza de los mismos.

Dicha facilidad de implementación, hace que cuando surge una nueva empresa fabricante de dispositivos con las características adecuadas, Android sea el sistema operativo elegido en la inmensa mayoría de los casos.

El número de dispositivos que incorporan Android, ha crecido de forma imparable desde sus inicios, teniendo actualmente una cuota del 80% del mercado a nivel mundial (datos del IDC del último cuatrimestre de 2013). En la siguiente tabla se muestra el crecimiento de los diversos sistemas operativos móviles en los diferentes en el último periodo de 2012/2013.

Top Four Operating Systems, Shipments, and Market Share, Q3 2013 (Units in Millions)

Operating System	3Q13 Shipment Volumes	3Q13 Market Share	3Q12 Shipment Volumes	3Q12 Market Share	Year-Over-Year Change
Android	211.6	81.0%	139.9	74.9%	51.3%
iOS	33.8	12.9%	26.9	14.4%	25.6%
Windows Phone	9.5	3.6%	3.7	2.0%	156.0%
BlackBerry	4.5	1.7%	7.7	4.1%	-41.6%
Others	1.7	0.6%	8.4	4.5%	-80.1%
Total	**261.1**	**100.0%**	**186.7**	**100.0%**	**39.9%**

Tabla 3: SO móviles

El núcleo del sistema, en sus diferentes versiones, incorpora funcionalidades como el uso de GPS, Wi-Fi, síntesis de voz, reproducción multimedia… etc, lo que hace que el tiempo dedicado a desarrollar aplicaciones para el mismo no se pierda en búsqueda de APIs de terceros. La facilidad que provee la interfaz de Android ha hecho que surjan millones de aplicaciones en sus escasos siete años de vida. A esto también ha ayudado la existencia del mercado de aplicaciones para Android, Play Google, cuyo acceso para colocar aplicaciones es muy sencillo.

También es destacable la existencia de otros mercados de software libre y código abierto para Android, como F-Droid, algo inexistente en el resto de sistemas operativos de dispositivos móviles. De nuevo, siguiendo con la filosofía inicial, el propio sistema permite instalar aplicaciones (archivos con extensión .apk) directamente desde la memoria interna o una memoria flash, siempre aduciendo a la responsabilidad del usuario y al peligro que algo así puede conllevar ya que no dicha aplicación, no ha tenido por qué pasar los filtros de Google.

La arquitectura de Android se ordena en tres niveles, organizados en cuatro capas. Primero encontraríamos el nivel más bajo: el núcleo de Linux. En este nivel encontramos todos los controladores del dispositivo. El segundo nivel está distribuido en dos capas: librerías nativas y runtime de Android. Aquí es importante diferenciar entre las librerías básicas de Android, encontradas en el runtime, y las nativas. Las básicas, o core, hacen referencia a las funciones más básicas del sistema operativo, las que, habitualmente, se llaman a través de las nativas. Éstas últimas, por contra, son aquellas que dotan de utilidad adicional al sistema operativo para poder manejar gráficos o bases de datos, entre otros. El tercer nivel corresponde al entorno de aplicación, el cual controla el acceso de las aplicaciones a los diferentes recursos del dispositivo. Por último, encontramos el nivel de aplicación, en el que se encuentran los desarrollos finales. El siguiente gráfico ilustra la arquitectura descrita.

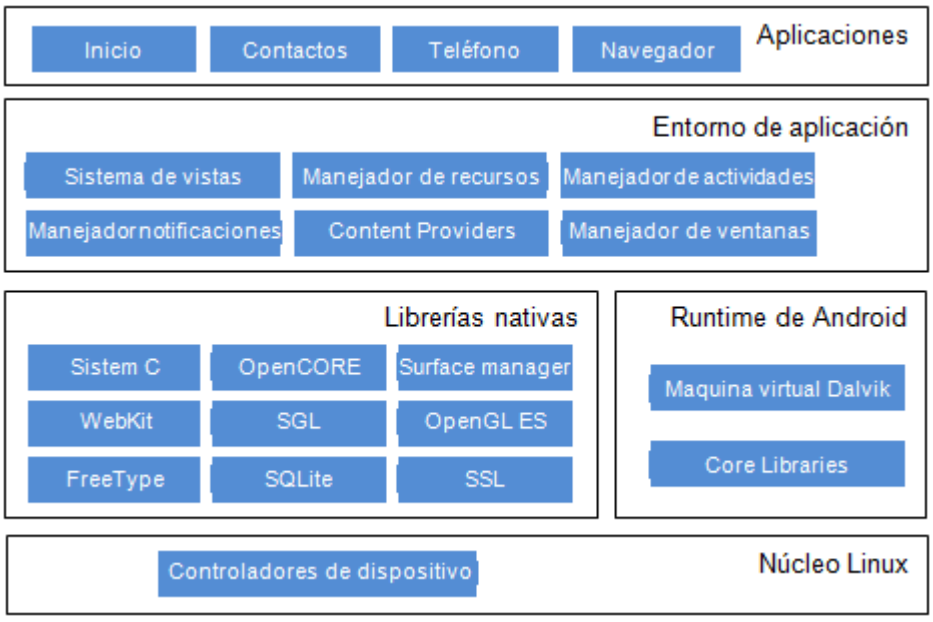

Figura 4: Capas de Andorid

El sistema operativo Android ha ido pasando por diferentes versiones, mejorando y actualizando su núcleo y sus funciones, haciendo posible el uso de las diferentes opciones hardware que los fabricantes han ido añadiendo a sus dispositivos móviles a lo largo de estos años. Así, se partió de la versión 1.0 en Septiembre de 2008 hasta la versión actual, la 4.4, o

KitKat. A mediados del presente año (Junio - Julio), aparecerá la nueva versión de Android (4.5 ó 4.6) con el sobrenombre de Lime Pie. La versión actual, destaca por su mejora en cuanto a rendimiento se refiere. El talón de Aquiles de los sistemas operativos móviles es el rendimiento frente al consumo. Aunque poco a poco se ha ido mejorando el problema con baterías con mayor capacidad y más eficientes, las posibilidades que el sistema operativo Android ha ido ofreciendo a las aplicaciones ha crecido mucho más rápido. La versión 4.4 solventa parte de este problema ofreciendo un rendimiento bastante atractivo sin llegar a consumir en exceso.

Por otra parte, otro gran atractivo de esta versión es la mejorada funcionalidad de la multitarea. Haciendo uso del hardware que la gran mayoría de dispositivos actuales dispone, se ha logrado realizar actividades en segundo plano sin una gran repercusión en el rendimiento del dispositivo.

Moviéndonos al plano de la usabilidad, KitKat añade varias funciones muy atractivas que harán que la experiencia del usuario sea mucho más confortable que anteriormente. Algunas de estas funciones son el modo envolvente, con el que se pueden eliminar de la pantalla cualquier tipo de barra de aviso y volver a ponerla con un simple movimiento de dedo; la identificación de llamadas de números que no estén en la agenda a través de las fichas locales de Google Maps; la encapsulación de todos los mensajes, ya sean SMS, MMS, videollamadas, etc, en una sola aplicación, Hangouts, y otras.

6.2.2 Sistemas de voz

Introducción

Cuando hablamos de “sistemas de voz”, nos referimos tanto al reconocimiento del habla como a la reproducción de un texto con voz. Actualmente, la tecnología de que disponemos hace que una tarea que hace años se pensaba como difícilmente alcanzable sea mucho más que accesible. En cuanto a dispositivos móviles se refiere, todos los sistemas operativos actuales disponen de alguna herramienta de reconocimiento de voz bastante funcional. Aunque se está lejos de la perfección, la captación de voz y su traslación a texto es algo muy conseguido hoy día.

Del mismo modo, la reproducción de un texto y su conversión a voz está muy avanzada y hay decenas de herramientas que actualmente permiten tal tarea. De nuevo, en cuanto a dispositivos móviles se refiere, todos los sistemas operativos contienen alguna herramienta para llevar a cabo este trabajo.
Aunque a día de hoy no es una tecnología que se use habitualmente, su existencia propicia muchas facilidades a la hora de realizar según qué tareas: comandos básicos para el uso de

manos libres, traducción, búsqueda web, etc. Mención especial merece la orientación de estas herramientas para personas con discapacidades, especialmente aquellos con visibilidad reducida.

Reconocimiento de voz

[19] [20]
El reconocimiento de voz es una de las formas de comunicación con los sistemas artificiales que se está sobreponiendo con más fuerza a otras formas de interacción más tradicionales, como los botones físicos. Aunque su uso sobre la funcionalidad táctil sigue siendo bastante inferior, también la está desplazando en algunos casos.

Desde hace unos años es posible interactuar con diferentes dispositivos móviles a través de órdenes básicas por voz. Algunas *smart TV's* son capaces de entender mensajes relativamente complejos (cambios de canal, encendido o apagado del televisor, etc). De hecho, cada vez es más habitual que los coches que salen nuevos al mercado incorporen algún sistema de reconocimiento de voz para, entre otras cosas, poder encender la radio, activar el GPS o calcular una ruta.

El reconocimiento de voz no sólo juega un papel importante, en ascenso, en los productos de consumo. En ciertos segmentos empresariales y profesionales, como el dictado médico o en los laboratorios, esta tecnología ahorra una gran cantidad de tiempo.

Pero estos sistemas no existen desde hace sólo unos años si no que ya se usaban con anterioridad en la atención telefónica, en la que te solicitaban un comando en concreto o un identificador para acceder a algún número o extensión en concreto. La tecnología actual no es más que una mejora de estos antiguos sistemas.

El funcionamiento de los sistemas de reconocimiento de voz se distribuyen en varias capas. El primero es el modelo acústico, que permite a la tecnología identificar la procedencia del sonido. Esta detección del canal de comunicación es vital para poder establecer el grado de distorsión que el mensaje puede experimentar.

La segunda capa trata el modelo lingüístico, es decir, el idioma. No es tan sencillo como preparar el dispositivo para que entienda algún idioma en particular. En ocasiones, hay que hacer pequeñas modificaciones a esta capa para que pueda entender un idioma con cierto acento, como puede ser el español hablado en Andalucía o en Galicia.
Por último, encontramos un motor que realiza un sondeo estadístico para encontrar la orden o

función más similar al audio recibido. Esta información se encuentra en una base de datos que, dependiendo de la aplicación y de las necesidades de los diseñadores, puede autoevaluarse y aprender con cada nueva consulta.

Reproducción de voz[7]

Los sistemas de reproducción de voz, o TTS por sus siglas en inglés (*Text To Speech*), son aquellos que permiten, por medios automáticos, la conversión de texto a voz artificial de forma que se asemeje al sonido que reproduciría una persona al leer dicho texto.

La historia de los sistemas de reproducción de voz, al igual que en el caso del reconocimiento de voz, no nació con los dispositivos móviles sino que ya en el siglo XIII, Alberto Magno y Roger Bacon, trataron de crear unas "cabezas parlantes" primitivas. Siguiendo esta idea de la creación de un dispositivo similar al humano, Wolfgang von Kempelen describió en una de sus obras una máquina accionada por un fuelle que sería capaz de reproducir vocales y consonantes [16]. Esta máquina dispondría de modelos de lengua y labios para tal fin. En 1837, Charles Wheatstone creó una máquina parlante basándose en el diseño de von Kempelen [17].

Ya en los años 30 del siglo XX, los laboratorios BELL desarrollaron el VOCODER [18], un analizador y sintetizador del habla que era operado por teclado. Este primitivo acercamiento a los sistemas actuales era claramente inteligible, aunque distaba mucho de una voz natural. El primer sistema de síntesis computarizado fue creado a finales de 1950, creándose el primer sistema completo texto-voz en 1968. Desde entonces, los acercamientos se han ido produciendo con mejores o peores resultados, pero avanzando en cuanto a la reproducción como tal se refiere.

Los sistemas conversores TTS deben ser capaces de reproducir una voz sintética lo más natural e inteligible posible, de forma que no haya problemas en la escucha. La razón es evidente, ya que de no cumplir este requisito, el sistema no podría alcanzar el fin con el que fue diseñado.

Otro importante requisito de estos sistemas es la síntesis de la voz artificial. Ésta ha de ser totalmente automática, sin necesidad de realizar ningún tipo de ajuste en ninguna parte del proceso. El campo de la síntesis de voz da para un trabajo mucho más largo y extenso que el apartado en el que nos encontramos, por lo que se procederá a enumerar las diferentes tecnologías de síntesis existentes:

[7] http://www.w3.org/TR/speech-synthesis/

- Síntesis concatenativa
- Síntesis por selección de unidades
- Síntesis de difonos
- Síntesis específica para un dominio
- Síntesis de formantes
- Síntesis híbrida
- Síntesis basada en modelos ocultos de Márkov

La conversión texto-voz se realiza a través de dos grandes fases. En la primera, se realiza una representación lingüística simbólica siguiendo los siguientes tres procesos de forma consecutiva:

1. **Normalización (*tokenización*) del texto**: la totalidad del texto se convierte a una forma textual convencional. Esto afecta principalmente a elementos como cifras, abreviaturas y otros.
2. **Conversión fonética**: con el texto normalizado, se asignan transcripciones fonéticas a cada palabra. Este proceso se denomina TTP (*Text To Phoneme*) o GTP (*Grapheme To Phoneme*).
3. **División prosódica**: el texto se divide en unidades prosódicas, esto es, unidades sintagmáticas, proposiciones y frases.

La segunda fase es la que forma el sintetizador de voz como tal. Se toma como entrada la representación lingüística simbólica que se ha obtenido en la fase anterior y se transforma en voz sintética.

Aplicaciones de los sistemas de voz

Como se ha mencionado en un párrafo anterior, las aplicaciones de los sistemas de voz son muchas y variadas, aunque en la práctica, aún están bastante fuera del uso habitual de cualquier entorno profesional.

Una aplicación directa y con gran repercusión es la redacción. Tomar notas o redactar un escrito a la velocidad del habla, supondría una enorme ganancia de tiempo para cualquier profesional que se dedique a realizar muchos escritos de cualquier tipo (abogados, científicos, etc). Para esta tarea, el sistema debe ser suficientemente bueno como para no cometer muchos errores ya que, en caso contrario, el tiempo ganado por un lado se perdería al tener que revisar el texto y corregir todos los errores.

Por otro lado encontramos que los sistemas TTS son de enorme utilidad en cuanto a

accesibilidad se refiere. Estos sistemas se pueden usar para que una persona con una visibilidad reducida pueda acceder sin grandes dificultades tanto a una página web como a un formulario digitalizado. Permite, igualmente, la interacción directa de dichas personas con ordenadores o sistemas móviles ya que se puede reproducir cada acción que el individuo haga, asegurándole un acceso sencillo.

6.3 PROYECTO

A continuación se detallarán cada una de las partes que han sido desarrolladas para implementar la funcionalidad necesaria para cumplir los objetivos establecidos.

En primer lugar hablaremos del **Mapeo del edificio**, o lo que es lo mismo, la organización y estructuración del edificio de la facultad para poder guiarnos por sus diferentes plantas. Además de ser un sistema genérico para facilitar su extensión a otros edificios.

Seguidamente se detallará la implementación de las diferentes partes: el **cliente** con el cual interactúa el usuario. El **servidor,** donde se calcula la ruta y se generan las instrucciones**. La conexión cliente-servidor** y la **base de datos**.

Figura 5: Esquema conexión de la aplicación

6.3.1 Mapeo del edificio

Una de las características más importantes de nuestro proyecto y que lo diferencia notablemente de su predecesor [2], es la completa reestructuración que hemos llevado a cabo a la hora de medir las señales Wi-Fi y organizar las diferentes plantas del edificio.

Al ponernos al frente del proyecto, nos encontramos con un sistema que consideramos poco eficiente, se utilizaban cuadrantes para la división de los pasillos y aulas, que venían representados en metros en la propia aplicación servidor, lo que hacía muy costosa la labor de medir las diferentes zonas e integrarlas posteriormente en la aplicación.

Hemos empleado un sistema de divisiones y estructuración que permite que se puedan incluir de manera sencilla las diferentes plantas del edificio. Además es un método genérico, esto hace que pueda ser utilizado en otros edificios.

Comenzamos por desechar toda la parte del código referente a la creación y organización de la estructura del edificio. Lo sustituimos por unos archivos .xml que sacamos fuera de la aplicación. La estructura y manejo de estos ficheros viene detallada en el apéndice C: Mapear un edificio.

Desaparecen las medidas en metros, las plantas se dividen en cuadrantes, que son únicos en todo el edificio, lo que facilita el determinar en qué planta nos encontramos. Estos cuadrantes vienen determinados por sus posiciones sureste y noroeste así como un identificador único.

En nuestro edificio un cuadrante corresponde a nueve baldosas aproximadamente, tanto el largo como en ancho. Se han tomado medidas cada tres baldosas, por lo tanto un cuadrante consta de nueve medidas. Aumentando el número de medidas no se consigue mayor precisión en el posicionamiento.

Partiendo de una posición inicial (x,y,z) = (0,0,0) determinada por:

- **x**: correspondiente al eje de coordenadas X.
- **y:** correspondiente al eje de coordenadas Y.
- **z:** indica la planta en la que nos encontramos.

En la siguiente imagen correspondiente a una sección de un pasillo de planta primera se puede apreciar dicho división.

Los puntos **azules** corresponden a las zonas donde se han tomado las medidas dentro de un cuadrante.

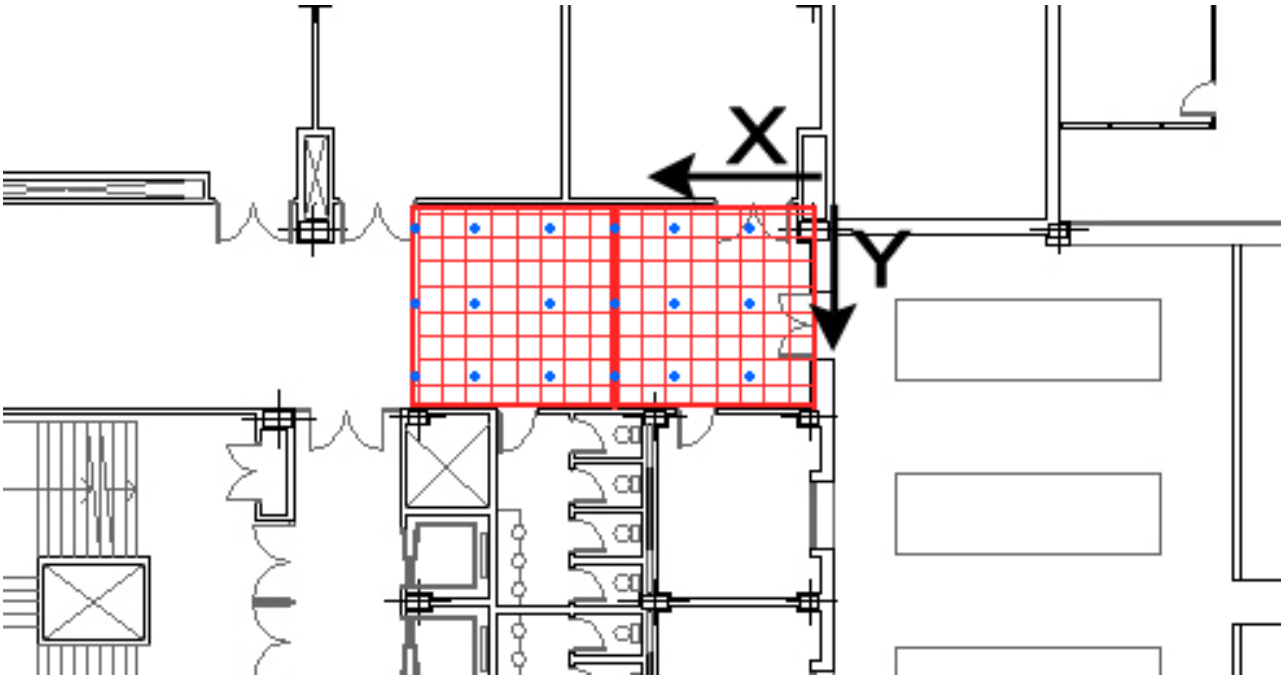

Figura 6: Medidas de un cuadrante

Vamos avanzando por el edificio midiendo las intensidades wi-fi con la aplicación cliente y almacenarlas en la base de datos con sus correspondientes coordenadas.

Los cuadrantes forman estancias, que pueden ser pasillos, aulas o laboratorios. De esta manera queda definida cada una de las plantas del edificio.

Uno de los aspectos por el cual tomamos la decisión de hacer esta división en cuadrantes únicos es para poder identificar fácilmente la planta en la que nos encontramos y así poder guiar al usuario por todo el edificio.

A continuación se muestra un mapa de la planta primera del edificio principal de la facultad:

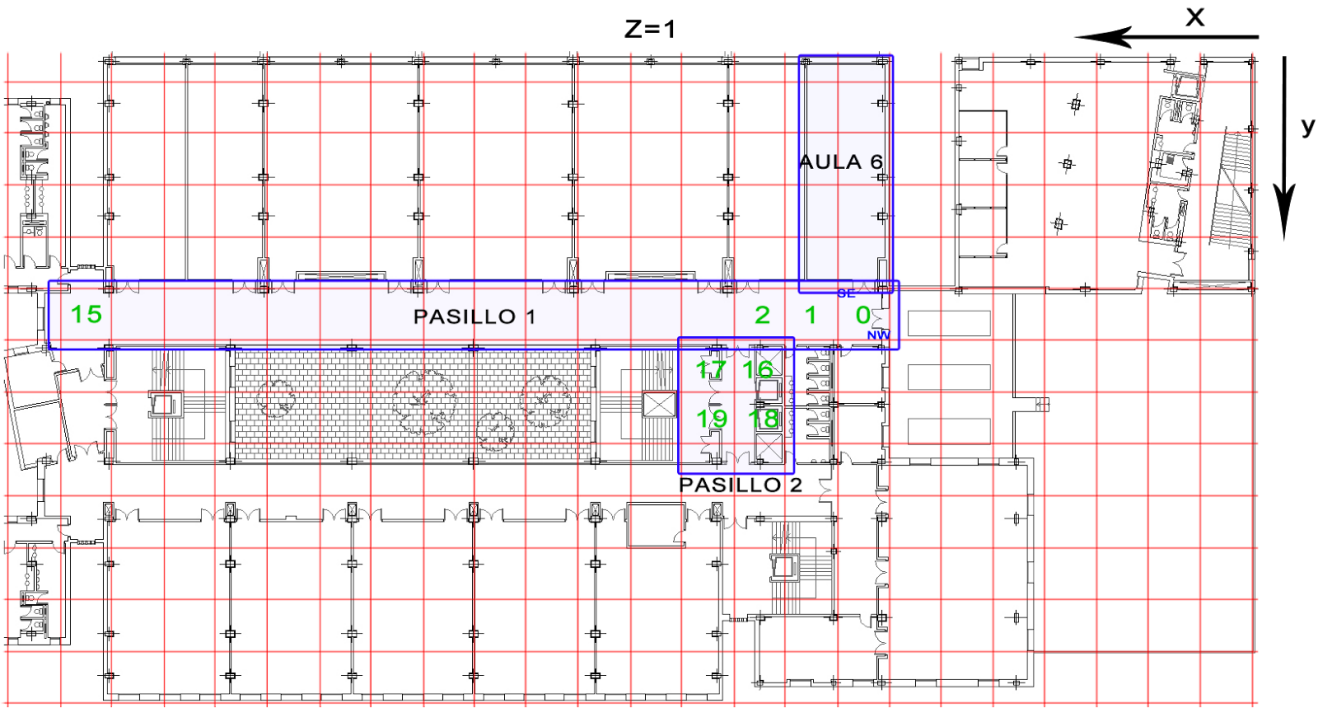

Figura 7: División en cuadrantes planta 1

En él se pueden apreciar:

- **cuadrantes**, en rojo.
- **identificador del cuadrante,** en verde.
- **algunas estancias**, en azul.
- **posición del cuadrante:** SE y NW, en azul.
- **eje de coordenadas**, (x,y,z), z=1 al tratarse de la primera planta.

Está estructura queda guardada en ficheros .xml almacenados en el servidor, en una carpeta de nombre 'xml' que se encuentra en el mismo directorio de la aplicación compilada del servidor .jar.
Estos ficheros son:

- **edificio.xml:** Archivo principal que indexa las diferentes plantas del edificio.
- **plantaX.xml:** Archivo que representa una planta del edificio, está dividido en estancias, que a su vez se dividen en cuadrantes con toda la información relativa a estos(posición SE y NW, conexiones con otros cuadrantes y objetos representativos).

Sin duda este ha sido uno de los aspectos más sofisticados, que más tiempo nos ha llevado y más problemas no ha causado. Estos problemas vienen dados principalmente por la inestabilidad de la señal wi-fi, que hacía que la labor de medir las intensidades fuera una tarea

muy costosa en tiempo, pese a ser tres personas con tres terminales móviles.

6.3.2 Cliente

El cliente es una aplicación móvil desarrollada para la API 19 de Android aunque compatible con versiones que implementen la API 7 o superior, esto es dispositivos que lleven como sistema operativo Android a partir de la versión 2.1.

Esta aplicación tiene una doble función: guiar al usuario hasta su destino y facilitar las mediciones de intensidades WiFi en el momento en el que se está preparando el sistema para su correcto funcionamiento. Por su parte la interfaz de guía requiere de dos funcionalidades más, obtener los datos relativos a la posición del usuario y la propia interacción aplicación-usuario que le llevará a este a su objetivo.

Como se ha mencionado previamente en esta memoria, la aplicación cliente extiende del proyecto AVANTI[1] A este se le ha añadido además del apartado de mediciones, un sistema de guía más sencillo e interactivo con el usuario (ya que ahora la indicación de la ruta a seguir depende en todo momento de la posición actual del dispositivo) y la posibilidad de funcionar en varias plantas, añadiendo una variable z que corresponde a la planta del edificio en el que se esté utilizando el sistema. Más adelante se expondrá más información sobre la implementación de estas mejoras en el capítulo actual.

A continuación se muestra un diagrama de clases de las actividades principales de esta aplicación cliente:

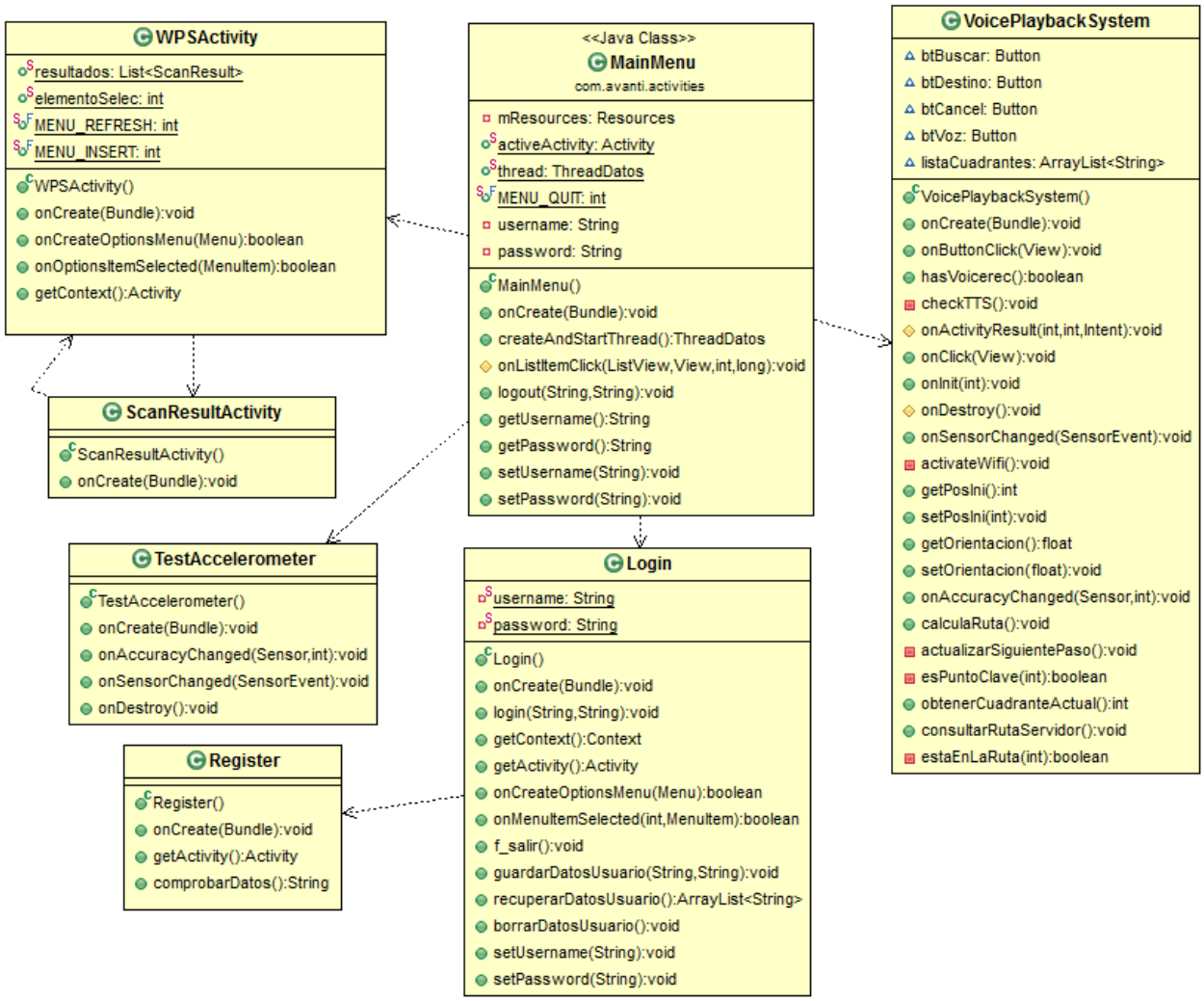

Figura 8: Diagrama de clases cliente

Estas activities de Android se suelen asociar a las pantallas principales de la interfaz de usuario, en nuestro caso se corresponde con las siguientes:

- **MainMenu:** es un ListActivity. En ella se muestra el acceso a todas las funcionalidades de la aplicación ordenadas en una lista.
- **Login:** es la pantalla de acceso a la aplicación, sirve para autentificar a los usuarios que estén registrados en el sistema, así como no permitir el acceso al resto. Para ello hace uso de la conexión con el servidor que contiene la base de datos, ya que en ella se halla la información relativa a estos.
- **Register:** esta clase está asociada a la pantalla de registro de un usuario. En caso de que un usuario no registrado en el sistema intente acceder a la aplicación, deberá introducir

sus datos necesariamente en está pantalla para que posteriormente se procese y guarde en el servidor que contiene la información de los usuarios.

- **TestAccelerometer:** pantalla en la que se muestran distintas variables que sirven de prueba del correcto funcionamiento del acelerómetro del dispositivo.
- **WPSActivity:** incorpora toda la funcionalidad relativa a la actividad WiFi alrededor del dispositivo. Permite actualizar la intensidad de la redes e insertarlas en el sistema a través de la conexión al servidor de la base de datos. Es útil para la medición y análisis en posiciones concretas del edificio que requieran un mayor control, debido por ejemplo a que sea un punto con bajas intensidades y que por lo tanto sería poco preciso para el posicionamiento.
- **ScanResultActivity:** a raíz de la pantalla correspondiente a WPSActivity muestra los datos de una red WiFi que se desee analizar.
- **Medir:** es la pantalla principal para el realizar el mapeo de edificio. Permite al usuario seguir posición a posición los puntos de los que debe medir e insertar los datos en el servidor con a lo sumo tres toques de pantalla.
- **VoicePlaybackSystem:** esta clase es la encargada principal de mostrar la interfaz de guía al usuario final. Permite la inserción de un destino (mediante voz o texto), que una vez enviado al servidor de cálculo de ruta mostrará las instrucciones paso a paso a paso según la ubicación actual del usuario. Para ello la aplicación consta de un método principal que se repetirá en bucle cada dos segundos. Este obtendrá la posición actual del dispositivo y comprobará si está en la ruta que ha recibido del servidor. En caso afirmativo esperará otros dos segundos para obtener una nueva ubicación, mostrando por pantalla el siguiente punto clave al que se debe dirigir el usuario. En caso negativo mandará al servidor la nueva ubicación con el objetivo de que recalcule la ruta ya que la actual es incorrecta.

VoicePlaybackSystem

Como se ha mostrado previamente esta activity será la principal en la tarea de guía de usuario. Como es usual en aplicaciones nativas para Android la implementación de pantallas consta dos partes, una que contendrá las funcionalidades (java) y otra con el diseño gráfico (xml). En este caso se corresponde con los archivos: VoicePlaybackSystem.java y playbackvoice.xml.

El fichero xml contiene la descripción de la pantalla, que está dividida en una parte para la interacción directa con el usuario (introducción del destino) y otra informativa (TextView para mostrar la ruta). En el apartado de la introducción del destino se posibilitan dos formas de interacción: mediante la escritura directa en un EditText a través del teclado o usando la propia voz del usuario. La opción seleccionada dependerá del botón que pulse el usuario, “Buscar destino” o “Búsqueda por voz”. Esta dualidad no complejiza en exceso la interfaz y amplia el

rango de posibilidades a la hora de utilizar la aplicación en distintos entornos, por ejemplo, facilitando el uso cotidiano mediante la introducción por voz o permitiendo su uso en espacios como bibliotecas o aulas en las que se requiere silencio. Por otra parte en la mitad inferior de la pantalla está ubicado un TextView que, tras los pertinentes cálculos que se ejecutan sin informar al usuario, mostrará una leyenda escrita con los movimientos que el usuario debe hacer para alcanzar su objetivo.

Esta interfaz está conectada, como hemos dicho, a una clase java que llevará el peso de asegurar que toda esta funcionalidad que estamos explicando se ejecute correctamente. Para ello su flujo de ejecución es el siguiente:

- Inicialización de todas la variables que van a ser usadas durante el proceso y enlace con la interfaz gráfica descrita en el xml.
- Puesta en espera del proceso a la escucha de dos listener, correspondientes al onClick los botones de “Buscar destino” y “Búsqueda por voz”.
- En el caso de que el usuario haga clic en “Búsqueda por voz” la aplicación llama a un nuevo activity de la librería TextToSpeech a la espera de un resultado, que será un string con el destino final dictado por el usuario, por ello se escribirá en el EditText que usamos a tal efecto. Además de opción cabe la posibilidad de que el usuario decida escribir la dirección a través del teclado directamente sobre el EditText y pulsar “Buscar Destino”. En ambos casos, si el destino ha sido insertado la aplicación continuará su normal funcionamiento.
- Una vez sabemos el destino se inicia el algoritmo principal que indica la ruta. Este es un bucle que se ejecuta cada 5 segundos (lanzado por un Timer), donde primeramente se obtiene el cuadrante en el que está posicionado el dispositivo, mediante el método de los k-vecinos. Esta posición se comparará con una serie de cuadrantes obtenidos del servidor para comprobar si el usuario está en la ruta correcta. En caso de que esta lista esté vacía o el cuadrante actual no esté en ella, se procede a consultar al servidor la ruta más corta indicándole la posición previamente calculada y la dirección destino. Una vez hecha la consulta se volverá a iniciar el bucle con los nuevos cuadrantes. Por el contrario si el usuario efectivamente está en la ruta se muestra por pantalla (y si está activado el volumen multimedia en el dispositivo también por voz) la instrucción de la siguiente posición a la que el usuario deberá ir. Al estar ejecutándose cíclicamente este proceso cuando el punto actual coincide con la posición indicada como siguiente objetivo en la ruta tanto el TextView como la voz sintetizada actualizan la siguiente referencia mostrándosela al usuario.

Medir

Esta pantalla ha sido implementada para facilitar la tarea de mapeo del edificio durante la configuración previa que se debe hacer del sistema. Para ello hacemos uso de la clase Medir.java y el documento xml medir.xml.

En lo que respecta a la estructura gráfica esta consta de tres bloques principales, uno por cada coordenada en el espacio que necesitamos medir. X e Y se corresponden con direcciones que forman el plano horizontal mientras Z será la altura de la planta en la que se esté midiendo (con posibilidad de fraccionarse para el caso de las escaleras entre una y otra). Por cada bloque mantenemos dos variables, la medición actual y la anterior con el objetivo de servir de recordatorio al administrador del sistema mientras está realizando la tarea. Además, como elemento interactivo, disponemos de dos botones señalados con las leyendas "+" y "-" cuya función es aumentar o disminuir el valor de la variable actual respectivamente. Esta variación la hemos fijado en cambios de +-0,5m ya que, según se expondrá en el apéndice B de mediciones en la guía de usuario, es la distancia óptima para que el sistema funcione correctamente. Finalmente disponemos de un botón general llamado "Refrescar y aceptar" que servirá para enviar los datos medidos al servidor.

En cuanto a la implementación java enlazamos todos los elementos gráficos con sus correspondientes variables y los hacemos corresponder con distintas subrutinas en los casos necesarios. Todas ellas conectadas mediante sus respectivos OnClickListener.

Para el caso de los botones de "+" y "-" estas rutinas son el incremento o decremento de la variable correspondiente a la coordenada actual para el bloque al que pertenezca el botón (X, Y o Z) y la actualización del TextView encargado de mostrárselo al usuario.

El comportamiento del botón de aceptación es más complejo. Primeramente se abre una conexión con el servidor en el que se almacenan los datos creando una instancia de la clase HttpServices. A través de ella se hace ping al servidor para comprobar si está operativo y en caso negativo devolver el error correspondiente. Si todo ha ido bien se hace una llamada al método refresh() que actualiza la lista de ScanResult llamada "resultados" con las mediciones de intensidades WiFi en la posición concreta en la que está el dispositivo en ese momento. Esta clase y los métodos necesarios para la actualización están incluidos en las librerías propias del sdk de android. Una vez hecho esto se manda al servidor la información indicada por el usuario con las mediciones efectuadas para que se almacenen relacionadas. Finalmente se asigna a las variables de posición anterior el valor de la posición actual, actualizando también el TextView que las muestra.

6.3.3 Servidor

Primeramente, se pondrán las características de la máquina usada como servidor:

Dirección: tot.fdi.ucm.es
Procesador: Intel Xeon CPU E5-2609 @ 2.40GHz
RAM: 4,00GB
Sistema operativo: Windows 8.1 Pro 64bits

La idea de la aplicación servidor es funcionar como un asistente a la aplicación cliente en el que se encontrarán los datos respecto a las diferentes plantas (a través de archivos *.xml*) y donde se realizarán los cálculos más pesados. Las instrucciones a seguir por el usuario, también se generarán en esta aplicación, quedando en manos del dispositivo móvil su reproducción.

A continuación se muestra un diagrama de clases así como una breve descripción de su funcionalidad.

- **Main**: método principal de la aplicación principal. Establece la conexión y espera a la petición de la aplicación cliente para realizar todos los cálculos.
- **Cuadrante**: clase que servirá para almacenar la información de los cuadrantes del mapeado.
- **Estancia**: clase que contendrá la información de pasillos, aulas y similares.
- **GenerarRuta**: clase que se utilizará para generar la ruta más corta entre dos cuadrantes dados.
- **LectorDestino**: clase utilizada para leer el JSON con los diferentes destinos a los que el usuario puede ir.
- **ListaCuadrantes**: lista completa de los cuadrantes del edificio.
- **Posición**: par de valores que indicarán la posición real de un punto en el mapa.
- **Persona**: clase que servirá de enlace entre la principal y la generación de la ruta a seguir.

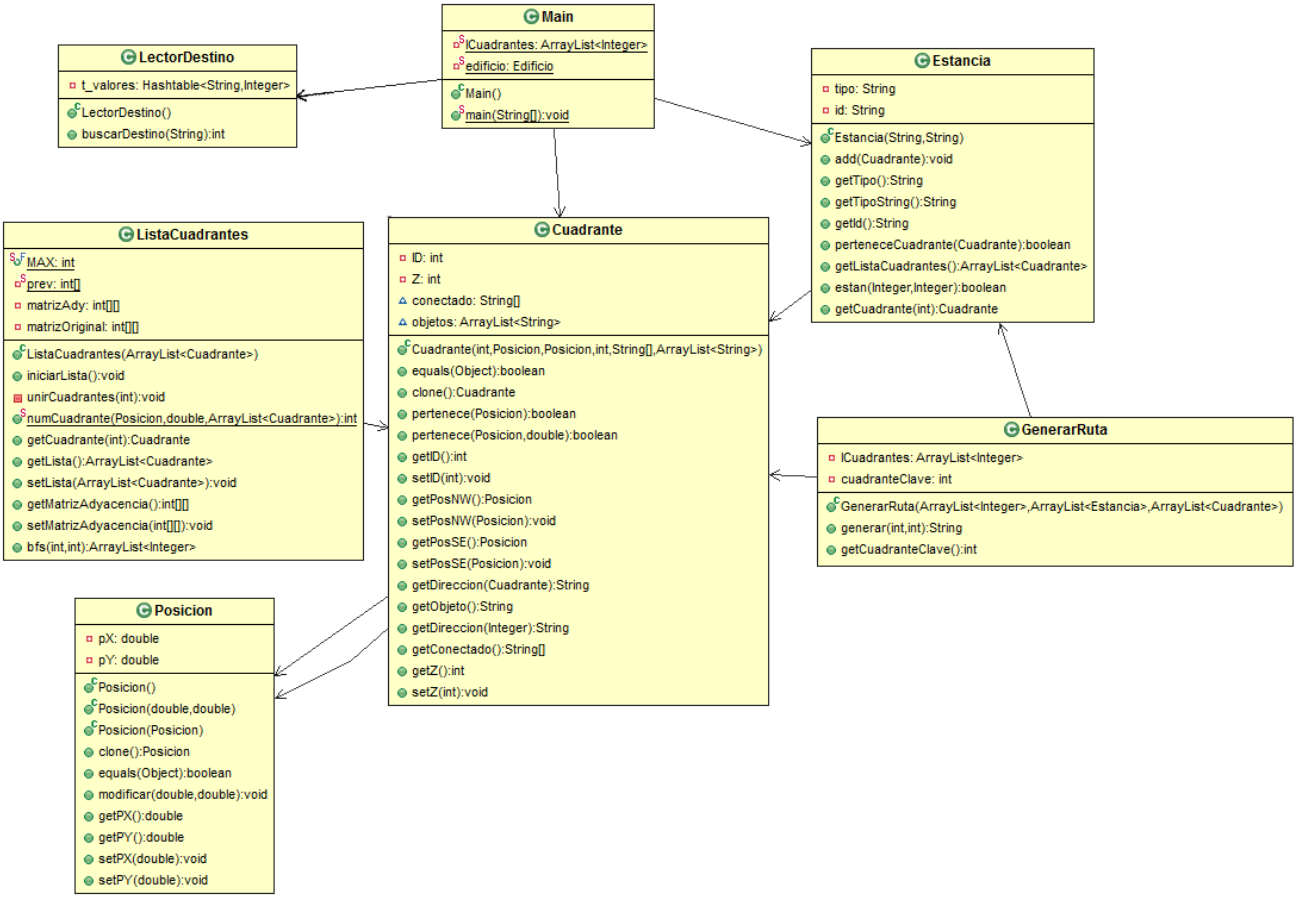

Figura 9: Diagrama de clases servidor

Main

Éste será el método principal que se ejecutará una vez lanzada la aplicación servidor. No contiene ninguna función auxiliar, aunque hace uso del resto de clases. Su flujo de funcionamiento será el siguiente:

- Cargamos todos los cuadrantes y estancias del edificio a través de la clase **Edificio**:
- Dentro de un bucle infinito, abrimos el puerto 22 y quedamos a la espera de la aplicación cliente.
- Una vez establecida la conexión, creamos el DataInputStream y el DataOutputStream, asociándolos al socket.
- Recuperamos la información pasada por la aplicación cliente. Esta información será, por orden:
    - **posOrigen**: coordenadas (x,y,z) de las que parte el dispositivo móvil.
    - **posDestino**: el índice del cuadrante al que quiere llegar el usuario.
    - **angOr**: el ángulo del dispositivo en el momento de la conexión.
    - **posAct**: coordenadas (x,y,z) en las que se encuentra el usuario en el momento de la conexión.

- Una vez comprobado que los valores tienen sentido, procedemos a calcular el camino mínimo desde el cuadrante **posOrigen** hasta el cuadrante **posDestino** a través de la clase **Persona**.
- Con este camino, generamos la ruta teniendo en cuenta todos los cuadrantes, las estancias, la posición actual **posActual** y la posición destino **posDestino**.
- Una vez obtenida la ruta y almacenada en un string, **ruta**, procedemos a enviarla al dispositivo.
- En este punto, ya habrá terminado el trabajo de la aplicación servidor por lo que cerramos el socket y volvemos al inicio del bucle.

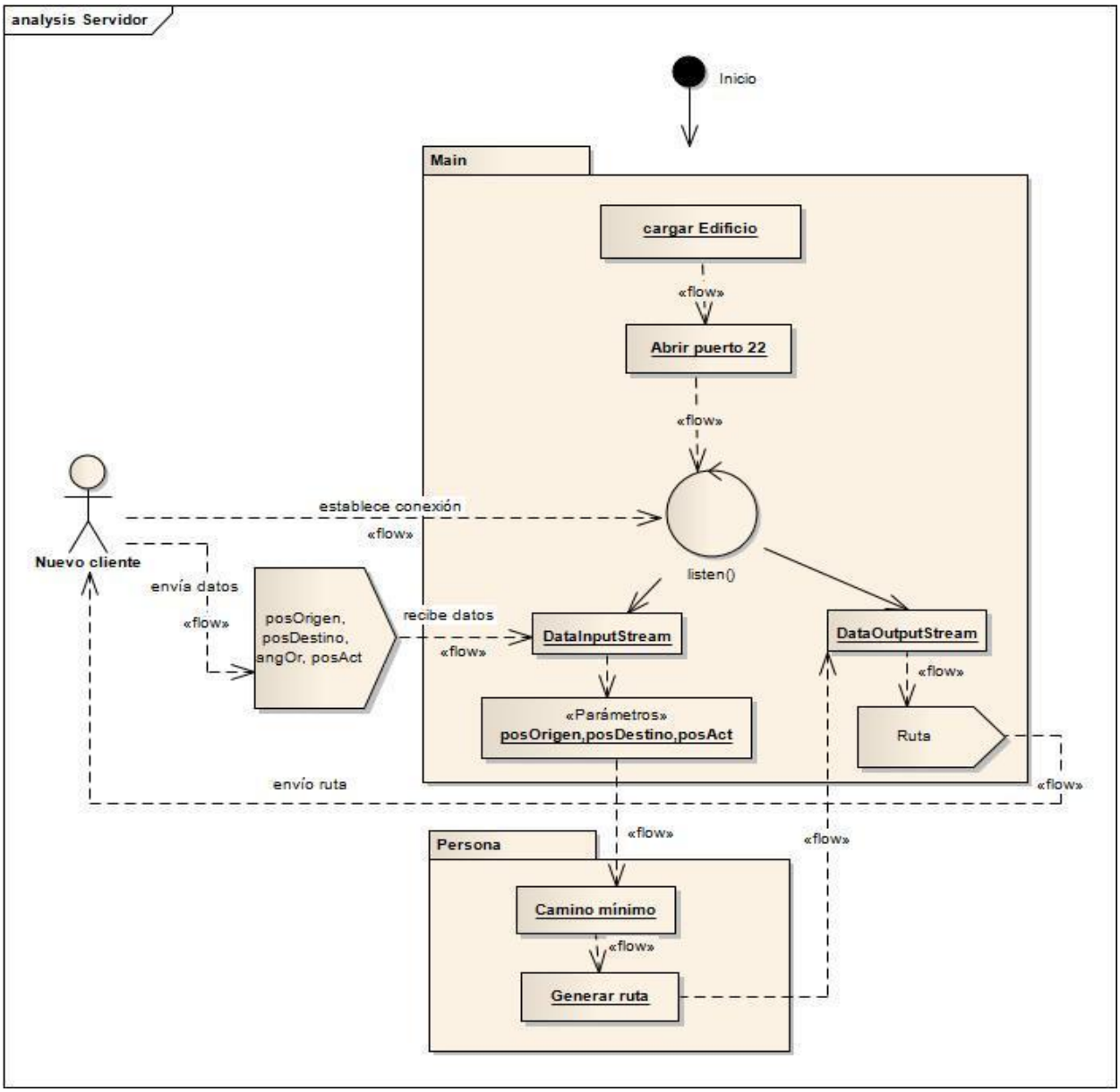

Figura 10: Diagrama de flujo servidor

Cuadrante

En esta clase almacenaremos la información referente a los cuadrantes que componen el mapeado del edificio. Cada cuadrante contendrá un identificador **ID**, un valor que indicará la planta en la que se encuentra **Z**, dos atributos de tipo **Posicion**, **posNW** y **posSE**, que indicarán la posición de las esquinas opuestas del cuadrante, un array de **String**, **conectado**, que indicará el **ID** del cuadrante con el que está conectado siguiendo el siguiente esquema:

- **conectado[0]**: norte
- **conectado[1]**: sur
- **conectado[2]**: este
- **conectado[3]**: oeste

En la siguiente imagen de una sección de un pasillo se pueden apreciar una par de cuadrantes y las direcciones de sus conexiones.

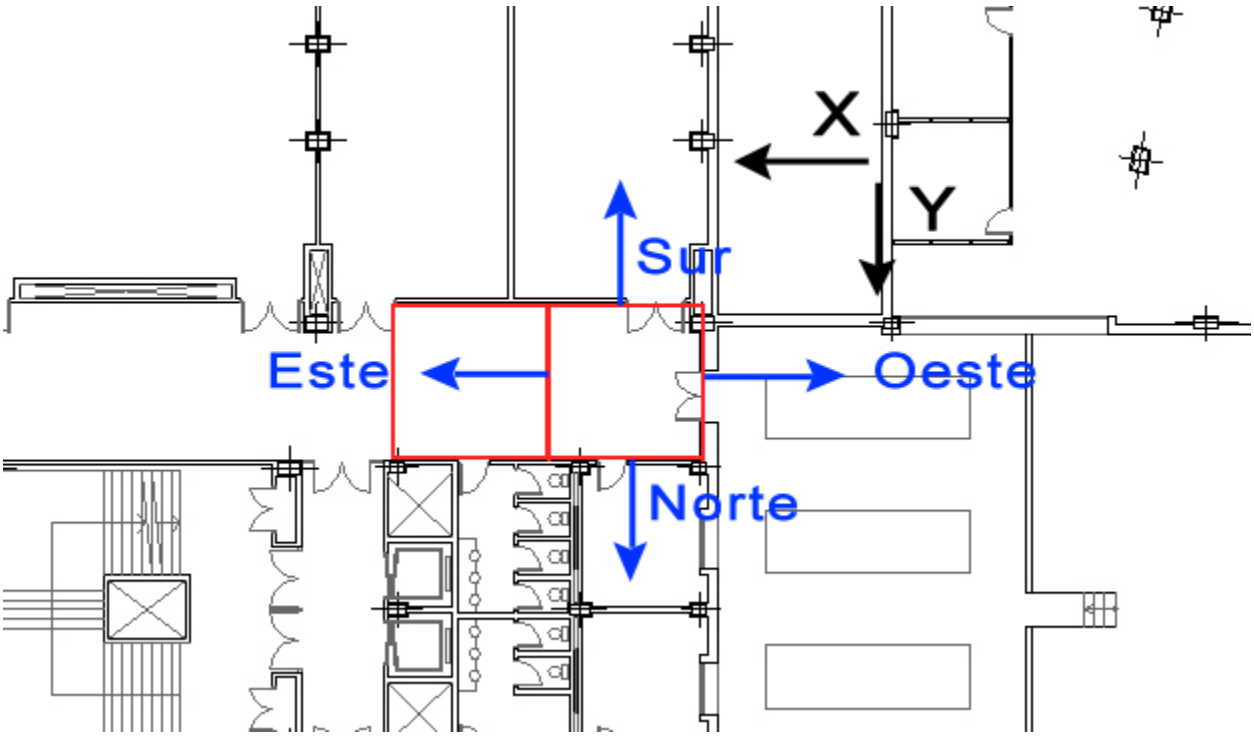

Figura 11: Conexiones de un cuadrante

Finalmente, tendremos un **ArrayList** de **String**, **objetos**, que contendrá los objetos más representativos del cuadrante como extintores, bancos, taquillas… etc.

A parte de los métodos típicos, cabe destacar dos: *pertenece* y *getDireccion*.

Pertenece

En este método se comprobará si dado una **Posicion p**, ésta cae dentro del cuadrante. El método consiste en comprobar si dicha posición **p** se encuentra dentro de los límites marcados por los atributos **posNW** y **posSE**.

getDireccion

En este caso, dado un cuadrante **c2**, obtendremos la dirección en la que está conectado con el actual, de existir tal conexión. Este método es especialmente útil a la hora de generar las instrucciones ya que nos servirá para conocer la dirección que tenemos que tomar.

Cargar edificio

Al iniciar la aplicación servidor, lo primero que se hace es cargar y guardar la estructura del edificio que viene dada en diferentes archivos xml. Las clases encargadas de ello son las siguientes.

- **Edificio**: método principal que carga el archivo edificio.xml
- **CargaXML**: clase que se encarga de cargar cada una de las plantas del edificio.

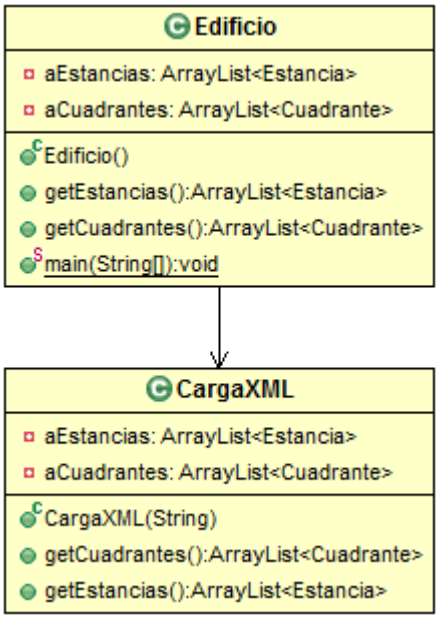

Figura 12: Diagrama de clases edificio

El contenido y estructura de los ficheros es el siguiente:

- **Edificio.xml:**

```
<edificio>
        <planta nombre="Planta1">
                <archivo>Planta1</archivo>
        </planta>
        <planta nombre="PlantaN">
                <archivo>PlantaN</archivo>
        </planta>

</edificio>
```

A continuación se detalla cada campo:

- **nombre:** Nombre representativo de la planta que estemos implementando.
- **archivo:** Nombre del fichero .xml de la planta correspondiente.

- **Planta1.xml:**

```
<planta>
   <Z> X <Z>
   <estancia id="pasillo1">
   <tipo>{Pasillo, aula, laboratorio, etc}</tipo>
   <cuadrantes>
          <cuadrante idc="0">
                 <SE>
                        <X>3</X>
                        <Y>13</Y>
                 </SE>
                 <NW>
                        <X>0</X>
                        <Y>16</Y>
                 </NW>
                 <conectado>
                        <norte>-1</norte>
                        <sur>-1</sur>
                        <este>1</este>
                        <oeste>-1</oeste>
                 </conectado>
```

```
                <objeto>extintor</objeto>
            </cuadrante>
        </cuadrantes>
    </estancia>
</planta>
```

A continuación se detalla cada campo:

- **Z:** Indica el número de la planta.
- **id:** El identificador que le queramos dar a la estancia.
- **tipo:** Corresponde a los tipos de estancias que tengamos en nuestro edificio. En nuestro caso: Pasillo, aula y laboratorio.
- **idc:** Identificador de los cuadrantes.
- **SE:** Posición sureste del cuadrante, con sus coordenadas X e Y
- **NW:** Posición noroeste del cuadrante, con sus coordenadas X e Y
- **conectado:** Indican el identificador de los cuadrantes colindantes. El -1 representa la pared.
- **objeto:** Objeto o elemento más representativo del cuadrante.

Obtención del destino

El usuario puede introducir el destino deseado tanto por teclado como por voz. Una vez hecho esto, es enviado al servidor en una variable *String*.

Los posibles destinos de un edificio están almacenados en un fichero de formato ligero **JSON**, alojado en el propio servidor. Este tipo de fichero nos permite un acceso más rápido a los datos, ya que en un único recorrido se almacenan todos los datos en un **HashTable.**

El contenido del fichero **destinos.json** es el siguiente:

```
[
{
"lugar": "estancia 1",
"cuadrante": "identificador cuadrante",
},
{
"lugar": "estancia N",
"cuadrante": "identificador cuadrante",
}
]
```

El **lugar** es el destino que recibiremos del usuario y el **cuadrante** es el identificador del cuadrante desde el que se accede a esa estancia.
Al recorrerlo, estos datos son almacenadas en la variable:

t_valores = new Hashtable<String,Integer>();

En caso de que el destino introducido por el usuario no esté almacenado, se le informará con un mensaje de error.

Cálculo de la ruta óptima

El algoritmo implementado para calcular el camino mínimo entre dos cuadrantes es el **algoritmo de Dijkstra** [11][12].

Para ello, comenzamos rellenando una matriz de adyacencia con todas las conexiones entre los cuadrantes, de forma que tenemos algo así:

Id. cuadrante	**0**	**1**	**2**	**3**	**...**	**n**
0	0	1	0	0	...	0
1	1	0	1	0	...	1
2	0	1	0	1	...	0
3	0	0	1	0	...	0
...	...	...	...	...	0	...
n	0	1	0	0	...	0

Tabla 4: Matriz de adyacencia

De tal manera que un **0** indica que los cuadrantes no están conectados entre sí y un **1** que sí lo están.

Además de la matriz de adyacencia, usaremos una cola para almacenar los cuadrantes no visitados (las expansiones de los nodos) y un array para marcar los cuadrantes ya visitados (que usaremos como poda). Para saber cuántos cuadrantes debemos visitar, usaremos una variable como contador. Para saber qué nodos debemos visitar, usaremos un array solución.

Comenzaremos añadiendo el cuadrante inicial a la cola antes de entrar en el bucle. Una vez

dentro, éste se ejecutará hasta que la cola esté vacía, momento al cual si llegamos indicará que no hay una camino posible. En cada vuelta, cogeremos el elemento que toque de la cola, eliminándolo de la misma y añadiendo uno al contador. Si este elemento es igual al cuadrante final, habremos terminado la búsqueda, por lo que saldremos del bucle. Si no, marcaremos ese cuadrante como visitado y procederemos a expandir todos los cuadrantes adyacentes a éste. Cuando encontremos un nodo que sí sea adyacente y no haya sido visitado, lo añadiremos a la cola para expandirlo en próximas vueltas del bucle.

6.3.4 Conexión cliente-servidor

Para conectar la aplicación Android con el servidor y crear un flujo constante de intercambio de datos utilizamos la clase java.net.Socket[10] y java.net.ServerSocket[10]. Esta conexión se lleva a cabo mediante un protocolo TCP [13].

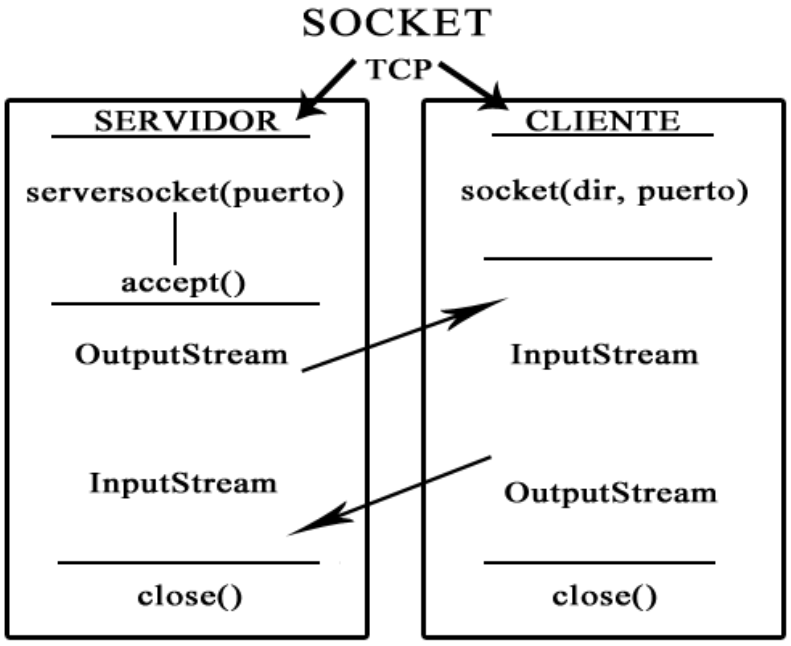

Figura 13: Diagrama conexión TCP cliente-servidor

En la aplicación cliente, disponemos de la clase Client, en ella se instancia un objeto de tipo **Socket** y se establecen la dirección y el puerto de entrada-salida. En nuestro caso:

Dirección servidor: tot.fdi.ucm.es

Puerto: 22

En la aplicación servidor se instancia un objeto de tipo **ServerSocket** asociado al puerto 22, que se mantiene en escucha, hasta que reciba y acepte una petición de conexión de un cliente.

Una vez establecida la conexión ya se puede proceder al intercambio de información mediante objetos **InputStream y OutputStream.**

En nuestra aplicación los datos que envía el cliente y recibe el servidor son:

- **origen:** coordenadas correspondientes a la posición inicial del usuario.
- **destino:** cuadrante correspondiente al destino que queremos ir.
- **ángulo orientación:** orientación del móvil en ese instante.
- **actual:** coordenadas en las que se encuentra el usuario en cada momento.

Por su parte, los datos que envía el servidor y recibe el cliente son:

- **lista:** contiene un listado de los cuadrantes que conforman la ruta óptima hasta el destino.
- **ruta:** cadena de texto con la siguiente indicación de movimiento.
- **cuadranteClave:** indica el próximo cuadrante hasta el que tiene que avanzar el usuario.

Para cerrar y dar por finalizada la comunicación basta con hacer un **close()** tanto de los flujos de datos como del socket.

6.3.5 Base de datos

Para almacenar los datos utilizados por la aplicación, tanto las medidas wi-fi como los usuarios, es necesaria una base de datos. La utilizada en el proyecto está realizada en MySQL, ya que es un sistema que destaca por su potencia de gestión a la vez que su sencillez.

A continuación una descripción detallada de cada una de las tablas que conforman la base de datos.

- Tabla **users**, que almacena la información personal del usuario al registrarse, su nombre y apellidos, el nombre de usuario y la contraseña, así como la última vez que se accedió al sistema y la última IP referente al posicionamiento que estuvo la última vez.

- Tabla **position**, guarda la información de las coordenadas en la que está el usuario, así como el usuario al que hace referencia y el tiempo en el que se produce.

- Tabla **repository**, almacena la información de todas las direcciones MAC correspondientes al mapeo del edificio. Se guarda la dirección MAC, así como la intensidad con la que se recibe y las coordenadas en la que se ha tomado.

En la siguiente imagen se presenta el Diagrama Entidad-Relación aplicado a la base de datos del sistema:

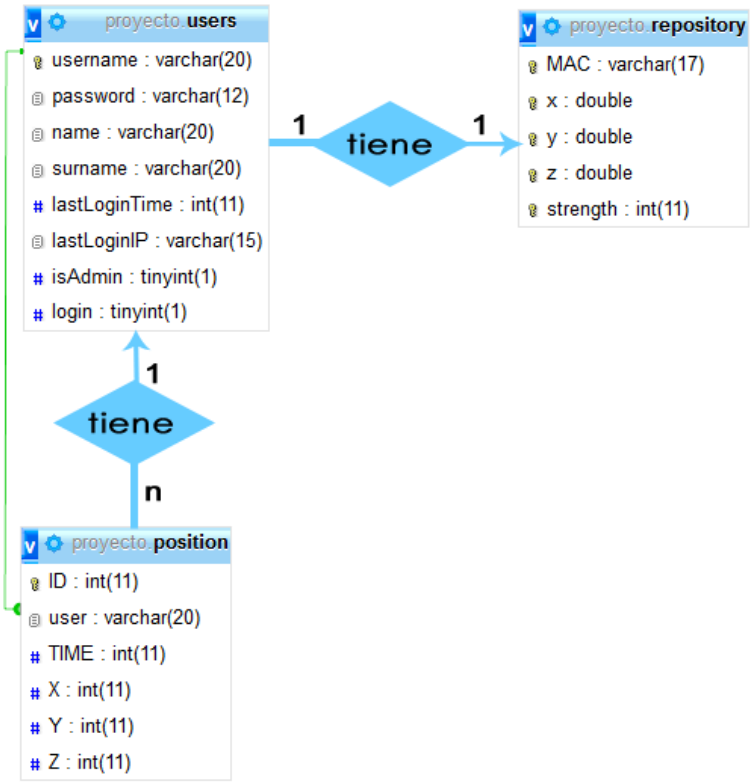

Figura 14: Modelo E-R de la base de datos

Conexión base de datos - cliente

La manera más sencilla de enlazar la aplicación cliente con la base de datos es realizando las consultas a través de ficheros PHP y HTML alojados en el servidor. Estos ficheros son los encargados de realizar las consultas directamente a la base de datos con las restricciones que pide la aplicación cliente.

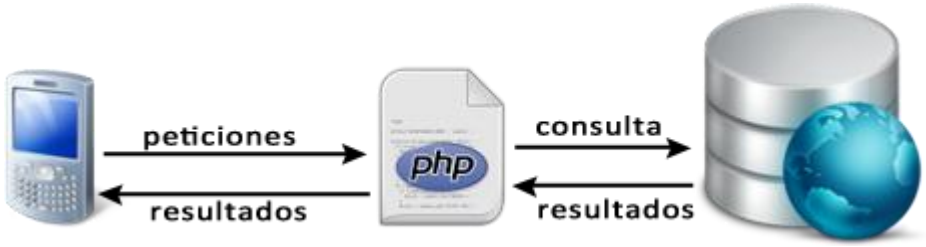

Figura 15: Conexión cliente-base de datos

A continuación un listado de estos ficheros y la funcionalidad que implementan:

- **index.html:** archivo principal mediante el cual se acceden a las consultas.
- **registrar.php:** se encarga de registrar un nuevo usuario en la base de datos.
- **consultar.php:** archivo que se encarga de dar acceso al sistema a un usuario. Es el "login".
- **getAllPositions.php:** obtiene todas las medidas wi-fi almacenadas.
- **insertNewPosition.php:** inserta una nueva medida wi-fi en la tabla **repository.**
- **logout.php:** finaliza la sesión del usuario en el sistema.

CAPÍTULO VII

RESULTADOS, CONCLUSIONES Y TRABAJO FUTURO

7.1 DESCRIPCIÓN DE LOS RESULTADOS OBTENIDOS Y ALCANZADOS

Una vez concluido el proyecto, podemos realizar un análisis de cada uno de los objetivos que se establecieron al principio, y determinar así en que medida se han cumplido.

- Establecimos como **primer objetivo** el poder detectar en que planta nos encontramos y guiarnos por todo el edificio. Podemos considerar que este objetivo se ha **cumplido en su totalidad**. Dentro de las limitaciones que ofrece el posicionamiento wi-fi, el sistema es capaz de estimar en que planta nos encontramos y guiarnos por todo el edificio generando las instrucciones apropiadas para cambiar de planta.

- Como **segundo objetivo** nos propusimos el crear un sistema genérico para la estructuración del edificio. Gracias a la división en cuadrantes que hemos hecho de las plantas y a su organización en ficheros .xml, no dejando nada metido en el propio código de la aplicación, ha sido posible completar este objetivo de **manera satisfactoria**. Hemos desarrollado un sistema que es independiente al edificio en el que nos encontramos, con lo que la aplicación podría ser utilizada en otros edificios mediante unas sencillas adaptaciones.

- El **tercer objetivo** era la renovación del sistema de generación de instrucciones para hacerlas interactivas. Hemos cumplido este objetivo **casi en su totalidad**, pues ahora el sistema da las instrucciones paso a paso para evitar que el usuario se pierda. Utilizamos elementos del mobiliario como puntos de referencia y un lenguaje más natural en las expresiones. Sin embargo, consideramos que aún se puede añadir más interactividad.

Este es un punto que ofrece muchas posibilidades y ha de estar en continuo desarrollo.

7.1.1 Resultados positivos

Tras finalizar el desarrollo de la aplicación, se pueden concluir una serie de resultados positivos.

- Se ha comprobado que las señales wi-fi resultan útiles para determinar la posición dentro de un edificio, incluso para detectar el cambio de planta. Si bien es cierto que serían necesarios más puntos de acceso para mejorar la eficacia.
- El manejo de ficheros .xml para gestionar la estructura del edificio ha resultado ser un gran acierto. Permite manejar una gran cantidad de información de manera muy sencilla y ágil. Además de permitir su modificación sin tener que acceder al código de la aplicación.
- La conexión entre el cliente y el servidor es muy fiable y consistente. A pesar del continuo intercambio de datos entre ambos, no se produce retraso alguno. Esto permite liberar de una gran carga de trabajo, como es el cálculo de la ruta mínima, al dispositivo móvil.
- Haber elegido el sistema operativo Android como base para el proyecto, en lugar de otros existentes en los actuales dispositivos móviles, ha sido un gran acierto debido al amplio mercado de éstos en comparación a los demás.
- La base de datos en la que se almacenan todas las medidas con sus respectivas potencias (de señales Wi-Fi) ha sido de gran utilidad en lugar de usar medidas introducidas directamente o usando ficheros externos.

7.1.2 Resultados negativos

Hay varios aspectos negativos que han limitado el desarrollo del proyecto:

- El posicionamiento wi-fi no resulta tan preciso como se esperaba. Ni aún ampliando notablemente el número de medidas dentro de un área, se consigue la precisión deseada. Probablemente se podrían obtener mejores resultados si se ampliase el número de puntos de acceso.
- El mapeado wi-fi del edificio es un trabajo muy costoso y está limitado a la calidad de la señal que en muchos casos es más baja de lo necesario.
- Distinguir de manera eficiente en que planta nos encontramos no siempre es posible debido a la inestabilidad de las señales recibidas.
- Trabajar sobre una aplicación preexistente ha supuesto algunos problemas debido a la falta de explicaciones en algunos casos y a la abundancia de partes de código cuyo uso era exclusivamente para pruebas, sin que estuviese etiquetado como tal.

7.2 CONCLUSIONES

Tras exponer todos los resultados, se puede concluir que el proyecto se ha desarrollado y se han cumplido los objetivos de manera satisfactoria.

Se ha establecido una línea de trabajo para el desarrollo de aplicaciones de guía en interiores que puede dar muy buenos resultados.

Ha sido gratificante y todo un reto el trabajar y conocer nuevas tecnologías como el wi-fi. Ofrece muchas posibilidades de desarrollo y sin duda la experiencia que hemos adquirido nos ayudará en un futuro.

Gestionar un proyecto ya existente y con tantas partes tan diferentes no ha resultado tarea nada fácil, pero finalmente hemos sabido enfrentarnos a él y cumplir con los objetivos previstos en un principio.

7.3 TRABAJO FUTURO

Sin duda alguna este proyecto tiene mucho potencial de cara al futuro, con una serie de ampliaciones y mejores, podría llegar a convertirse en una aplicación referente entre los sistemas de guía en interiores.

Para dejar la aplicación completamente funcional y que pudiera ser lanzada al mercado en su primera versión para ser utilizada por los alumnos, sería necesario completar el mapeado del edificio de la biblioteca, así como de las zonas del edificio principal que aún no han sido medidas. La integración de estas zonas no supondría un problema ya que el proyecto es fácilmente extensible.

Para darle más atractivo a la aplicación, sería interesante mejorar el sistema de generación de instrucciones. Crear expresiones más dinámicas y naturales, además de añadir diferentes objetos de referencia.

Esto incluye adaptaciones para generar instrucciones orientadas a usuarios con algún tipo de discapacidad, como pueda ser la falta de visión, motora o cerebral.
Un punto fuerte sería la generación de instrucciones visuales, ya dibujando una flecha en la pantalla del móvil que nos indique la dirección o mostrando imágenes de los diferentes puntos de referencia.

La Facultad de Informática tiene una estructura muy simple y bastante simétrica, por lo que habría que comprobar el correcto funcionamiento de la aplicación en cualquier otro edificio. Aunque esto tampoco no debería suponer ningún problema ya que es un sistema genérico, independiente de la estructura.

Al igual que tenemos la posibilidad de compartir nuestra ubicación en exteriores, una pequeña pero interesante funcionalidad sería el poder hacer lo mismo dentro de un edificio, para así facilitar el poder encontrarse con otra persona.

Por último, para hacer la aplicación más eficiente, sería conveniente estudiar otros posibles métodos de posicionamiento que dieran resultados más precisos que los utilizados actualmente.

APÉNDICE A

MANUAL DE INSTALACIÓN

En este apéndice se detallarán los pasos necesarios a seguir si se desea probar la aplicación o incluso instalar todas las partes del proyecto para su modificación.

A.1 REQUISITOS

La aplicación ha sido probada en diversos terminales de distintas características. Desde dispositivos de gama media como es un Motorola Moto G, hasta otros de gama alta como el Galaxy Note 3.

Se ha probado además en diferentes versiones Android, a partir de la 2.3 hasta la 4.4.2.

A.2 INSTALACIÓN EN ECLIPSE

Para instalar los archivos que conforman la aplicación cliente, se requiere de un entorno de programación Android. En nuestro caso hemos utilizado **Eclipse**[8], en el cual es necesario tener instalado el SDK de Android correspondiente[9]. El proyecto soporta las versiones 2.3 hasta 4.4.2, por lo que cualquier versión dentro de este rango será apta para ejecutar el proyecto.

Se proporciona la carpeta **Login** donde se encuentra el proyecto listo para ser importado en Eclipse. Para ello, haremos click en *File* y seleccionaremos *Import*. En la ventana emergente que aparece, pincharemos en *Android -> Existing Android Code into Workspace.* Seleccionaremos la ruta en la que está alojada nuestro proyecto y daremos aceptar.

[8] http://www.eclipse.org/

[9] http://developer.android.com/sdk/index.html

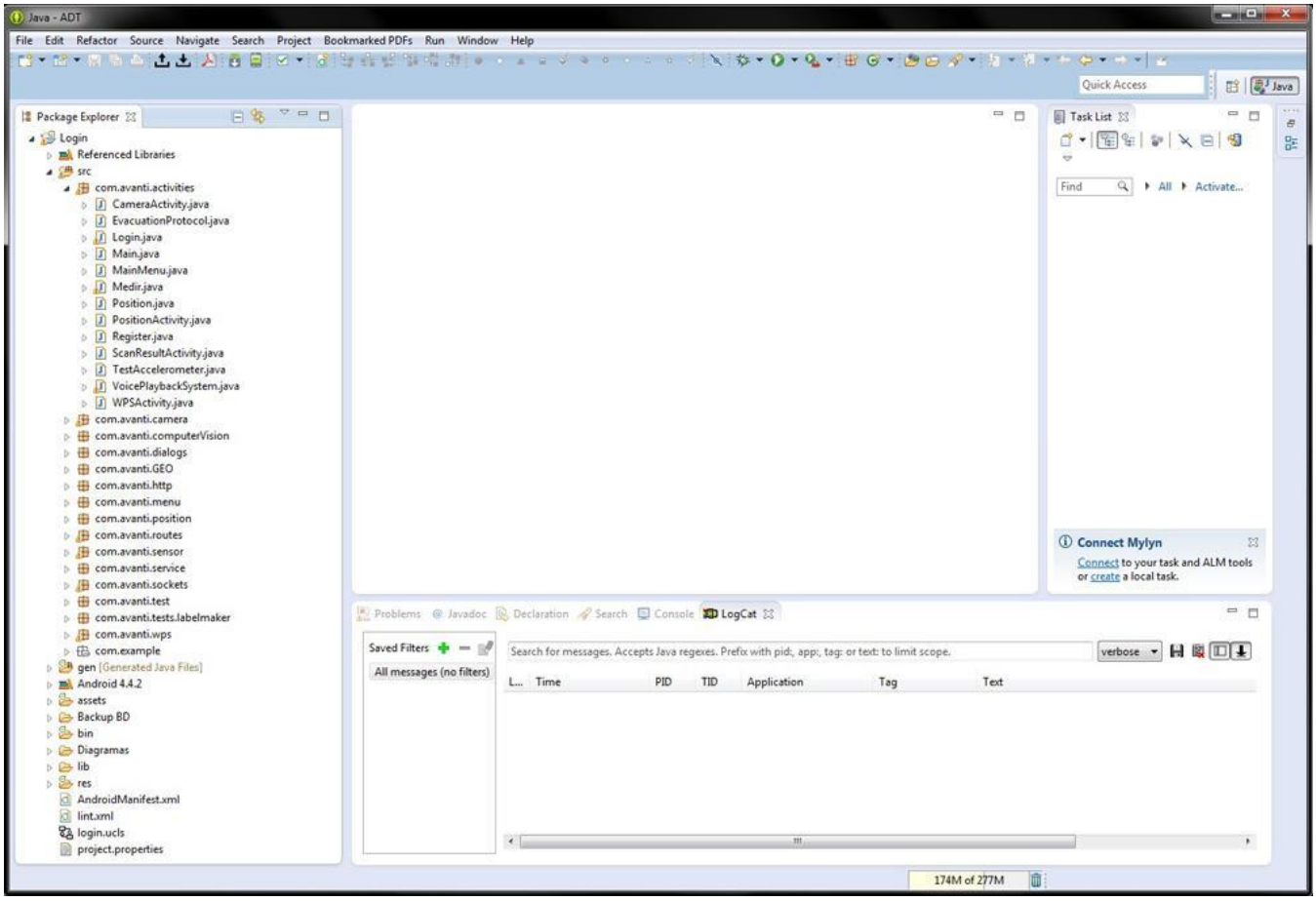

Figura 16: Cliente en Eclipse

A.3 INSTALACIÓN EN UN TERMINAL MÓVIL

Dentro de la carpeta bin del proyecto, encontraremos un ejecutable **Login.apk**, que podremos almacenar en nuestro dispositivo proceder a instalarlo. Para ello necesitaremos tener activada la opción de instalar aplicaciones de orígenes desconocidos en nuestro dispositivo, o bien hacerlo mediante un instalador de aplicaciones como los que hay disponibles en la *Play Store*.

Si se realizan cambios en el código, se tendrá que volver a recompilar el proyecto para obtener el .apk actualizado.

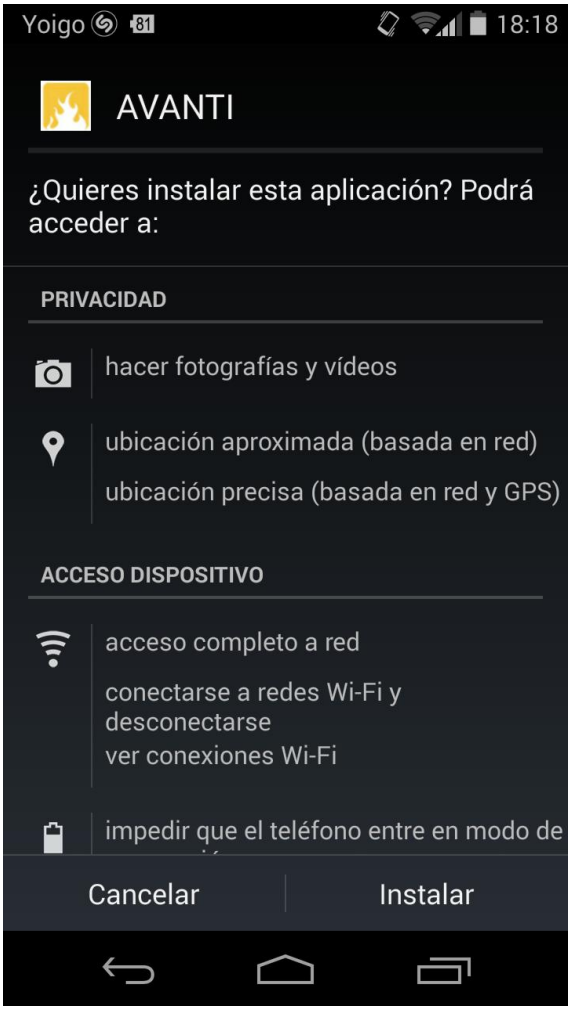

Figura 17: Permisos aplicación móvil

A.4 INSTALACIÓN DEL SERVIDOR

Si únicamente se desea probar la aplicación no es necesario realizar cambios en la parte del servidor, ya que todo lo necesario para su funcionamiento se encuentra convenientemente instalado y funcionando en los servidores de la Facultad de Informática.

En el caso de que queramos hacer modificaciones en las aplicaciones del servidor, **CalcularRuta** o en la **Base de datos**, nos remitimos al Apéndice C: Extensión de la aplicación, donde se explica todo lo necesario para establecer un nuevo servidor.

APÉNDICE B

MANUAL DE USUARIO

Este apéndice tiene como fin recoger el funcionamiento principal de la aplicación y ayudar al usuario a su manejo mediante explicaciones detalladas y capturas de pantalla

B.1 PANTALLA DE INICIO DE LA APLICACIÓN

Una vez instalada la aplicación en nuestro dispositivo móvil, lo primero que nos aparece cuando la ejecutamos es una pantalla con un sencillo formulario. Si el usuario ya ha utilizado previamente la aplicación, deberá introducir su usuario y contraseña y pulsar **Entrar** para quedar logueado en el sistema.

En caso de introducir unos datos erróneos, se le avisará con un mensaje de error, dándole la opción de volver a introducir los datos.

Si el usuario no está registrado, mediante el botón **Registrarse** podrá acceder al formulario de registro de la aplicación.

B.2 PANTALLA DE REGISTRO DE LA APLICACIÓN

Si es la primera vez que un usuario accede al sistema, deberá registrarse a través del formulario de registro. A esta pantalla se accede desde la ventana principal tras pulsar el botón Registrarse**.**

En esta pantalla se requerirán el **nombre, apellidos, usuario y contraseña**.

Una vez hecho esto, pulsará el botón **Registrarse,** y el sistema tras comprobar que no existe ya ese usuario en la base de datos, dejará registrado los datos del nuevo usuario.

El usuario ya podrá acceder a las funcionalidades principales de la aplicación introduciendo estos datos en la pantalla principal.

Figura 18: Pantalla de inicio aplicación

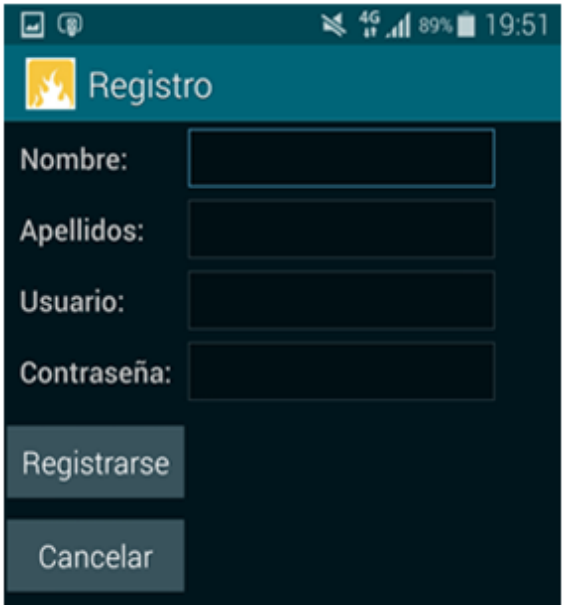

Figura 19: Pantalla de registro aplicación

B.3 Menú principal de la Aplicación

Una vez hemos accedido al sistema, aparecerá el menú principal de la aplicación en el que encontramos las siguientes opciones:

- Ver actividad Wifi
- Posiciónname en el mapa
- Aplicación real
- Prueba del acelerómetro
- Medir intensidades
- Indicar ruta hacia el destino
- Cerrar sesión

Figura 20: Pantalla menú principal

B.3.1 Ver actividad Wifi

En esta pantalla se muestra un listado con todas las redes inalámbricas detectadas por el dispositivo desde su posición. Aparece el nombre de la red y su dirección MAC.

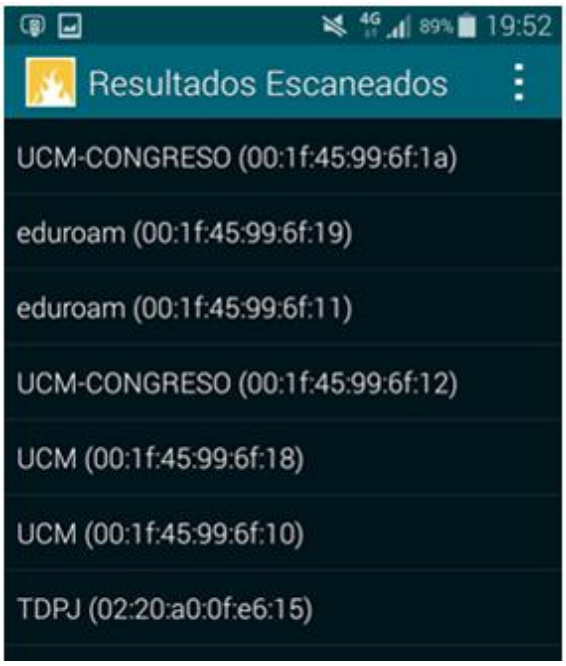

Figura 21: Pantalla actividad wi-fi

Si seleccionamos cualquiera de ellas, aparecerá una nueva pantalla donde se detallan las características de la red.

- Nombre de la red (SSID)
- Dirección MAC (BSSID)
- Frecuencia
- Potencia

Figura 22: Pantalla de datos de una red

De vuelta en la pantalla del listado de redes, si pulsamos el botón de opciones de nuestro terminal, nos aparece un pequeño menú con las siguientes opciones:
La opción **salir,** nos regresa al menú principal de la aplicación.
La opción **insertar (Server),** abrirá un 'dialog' en el que podemos introducir las coordenadas x,y,z de la posición en la que nos encontramos. Si pulsamos Aceptar, se introducirán en la base de datos.
La opción **refrescar lista**, realiza un nuevo escaneo de las redes.

B.3.2 Posicióname en el mapa

Se muestra un mapa de una planta de la Facultad, con las coordenadas de nuestra posición y el identificador del cuadrante en el que nos encontramos.

B.3.3 Aplicación real

En esta actividad, se muestra la vista de la cámara en la pantalla del dispositivo.

B.3.4 Prueba del acelerómetro

En esta pantalla, se muestran los valores del acelerómetro para los ejes X, Y, Z, así como el giro sobre dichos ejes. También se indicará si el giro es válido o no. El giro será válido si el teléfono se encuentra de forma horizontal con la cámara apuntando hacia el frente, en ese caso, se mostrarán los valores normalizados del acelerómetro en cada uno de los tres ejes.

B.3.5 Medir intensidades

Esta pantalla consta de un pequeño formulario para facilitar las mediciones de la actividad wi-fi. Aparecen tres etiquetas (x,y,z) que se corresponden con los valores de las coordenadas que estamos midiendo. A su vez estas etiquetas tienen dos valores: actual y última.
"**Actual**" representará la posición de la coordenada que queremos medir, modificando su valor con los botones "+" y "-" según queramos aumentarlo o disminuirlo. El incremento o decremento de este valor será siempre de 0.5 (usualmente su unidad será el metro). "**Última**" muestra el valor de la anterior coordenada medida y añadida a la base de datos, facilitando el seguimiento de un orden en la medición con el objetivo de evitar confusiones en la persona que esté realizando la tarea.

Por último, esta pantalla dispone de un botón con el título "**Refrescar y acept**ar". Su función es actualizar los datos de las intensidades wi-fi en la posición en la que se encuentre el dispositivo en ese momento y mandarlos al servidor para que los almacene con los valores de la coordenada indicados por el usuario. Una vez se ha realizado todo este proceso se actualizará el valor de última para que coincida con la coordenada recién medida.

Figura 23: Pantalla escaneo wi-fi

B.3.6 Indicar ruta hacia el destino

Esta es la pantalla que contiene la principal funcionalidad de este proyecto.

Figura 24: Pantalla insertar destino

En ella aparece un cuadro de texto donde el usuario podrá escribir el destino deseado para posteriormente pulsar el botón **Buscar destino**.

Si en lugar de escribir el destino, queremos indicarlo por voz, deberá pulsar el botón **Búsqueda por voz**.

Si el destino ha sido insertado correctamente, el sistema comprueba que esté activado el Wi-Fi en el terminal. Es necesario tenerlo activado, pues necesitamos saber la posición actual en la que se encuentra el usuario. Para ello, si el Wi-Fi no está activado, se abrirá un cuadro de diálogo, dónde solicitará activar el Wi-Fi, el usuario puede Aceptar, caso en el que se activará, o puede Cancelar, lo que provocará volver al menú principal.

Tras esto, aparecerá en pantalla la siguiente instrucción de guía que tiene que seguir el usuario. Además será reproducida por voz para facilitar su entendimiento.

Figura 25: Pantalla instrucciones

B.3.7 Cerrar sesión

Esta opción desconecta al usuario del sistema y le regresa a la pantalla principal de la aplicación.

APÉNDICE C

EXTENSIÓN DE LA APLICACIÓN

Uno de los grandes puntos fuerte de esta aplicación es que resulta tremendamente sencilla adaptarla para hacerla funcionar en otros edificios diferentes a la Facultad de Informática.
A continuación detallaremos paso a paso todos los pequeños cambios necesarios para adaptarla y que sea utilizable en cualquier otro edificio que se desee

C.1 CONEXIÓN CLIENTE-SERVIDOR.

Se requiere de un servidor en el que esté alojada y corriendo la aplicación del cálculo de rutas.
En primer lugar necesitaremos modificar esta aplicación para establecer el puerto de escucha que permitirá la conexión TCP entre la aplicación cliente y el servidor (Véase capítulo VI: Conexión cliente-servidor).

Para ello importaremos en **Eclipse** el proyecto **CalcularRuta.** Dentro del paquete **principal** en la clase **Main**, modificaremos la declaración de la variable:

- int PORT = **XX**;

Donde **XX** corresponde al puerto que utilizaremos para la conexión.

Una vez establecido, exportaremos el proyecto a un fichero **.jar** ejecutable, que tendremos que alojar en un directorio de nuestro servidor.

Por último, para ejecutar nuestra aplicación y que se mantenga a la escucha de clientes, hemos de seguir los siguientes pasos:

1. Instalar la máquina virtual de JAVA[X] en nuestro servidor
2. En Windows:
    a. Ir a Inicio->Ejecutar
    b. Escribir **cmd** y pulsar ENTER

3. Navegar hasta la carpeta en la que se encuentra nuestro **.jar**
4. Escribir: java -jar **NombreAplicacion.jar** y pulsar ENTER

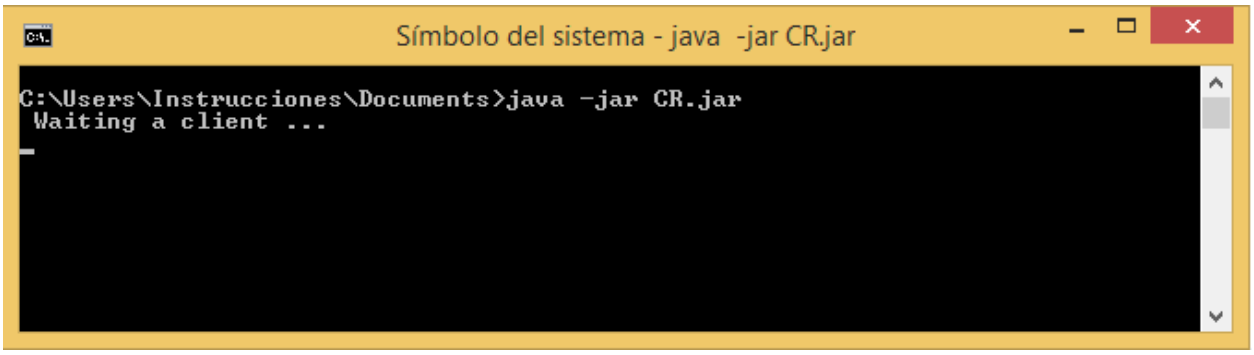

Figura 26: Ejecución aplicación servidor

De esta forma ya tendremos nuestra aplicación servidor escuchando, lista para recibir clientes.

A continuación tendremos que modificar nuestra aplicación servidor para establecer tanto la dirección en la que se encuentra alojada nuestro servidor, como el puerto de escucha.
Para ello, importamos el proyecto en Eclipse y nos vamos al paquete **com.avanti.sockets** y en la clase **Client.java** modificaremos las siguientes variables:

- String adresaServer ="***dirección servidor***";
- int PORT = **XX**; Ha de ser el mismo que el establecido en la aplicación servidor

C.2 CREAR BASE DE DATOS

Es necesaria una base de datos donde se almacenará la información relativa a las mediciones de la señal wifi así como de los usuarios de la aplicación. Las características y el diseño de las tablas ha de ser el mismo que los descritos en el capítulo VI de esta memoria.

Se requiere además de un *hosting* externo que nos permita alojar bases de datos MySQL así como los ficheros PHP necesarios para la conexión de la aplicación cliente con la base de datos.

C.3 ENLACE CLIENTE-BASE DE DATOS

En la aplicación cliente tenemos que modificar los datos relativos a la conexión con la base de datos. Con el proyecto abierto en Eclipse, iremos a la clase **HttpServices.java** que se encuentra en el paquete **com.avanti.http**, allí encontraremos las siguientes variables:

- public static String **host** = "**dir IP del hosting**";
- public static String **port** = "**puerto del hosting**";
- public static String **urlServer** = "http://direccionWEB";

Se recomienda utilizar un hosting de calidad debido al gran número de conexiones y la cantidad de datos que se transfieren entre el cliente y la base de datos.

En nuestro hosting web estarán alojados los ficheros PHP descritos en el capítulo VI: Base de datos.

En cada uno de estos ficheros, será necesario cambiar los datos de la conexión, en la siguiente línea:

- $conexion = mysql_connect("**dir IP**","**usuario BBDD**","**contraseña BBDD**");

Tanto la dirección IP, como el puerto son datos proporcionados por el proveedor de nuestro hosting. El usuario y contraseña de la base de datos los establecemos nosotros cuando la creamos.

C.4 MEDIR INTENSIDADES WIFI

Cuando tengamos la base de datos operativa, procederemos a mapear nuestro edificio, midiendo las intensidades wi-fi de los diferentes pasillos y salas o estancias.

Para tomar medidas, utilizaremos la aplicación cliente. Para ello, hemos de compilarla para crear el **.apk**, que instalaremos en nuestro terminal Android.

Utilizaremos la herramienta de tomar mediciones de nuestra aplicación, tal y como se detalla en el Apéndice B: medir intensidades.

En el capítulo VI: Mapeo del edificio se describe detalladamente el método utilizado para medir la Facultad de Informática.

Estableceremos un eje de coordenadas (X,Y,Z) en un punto del edificio. A partir de ahí iremos avanzando guardando las posiciones. Se recomienda tomar el mayor número posible de medidas para mejorar la precisión. Tomaremos como referencia dos o tres baldosas en caso de que las haya, si no, cada dos pasos.

C.5 Creación de los XML relativos a la estructura del edificio.

Los ficheros **xml** en los que se representa el mapeo de nuestro edificio.
La estructura estará compuesta por un primer fichero **edificio.xml** y tantos ficheros como plantas tengamos.

Edificio.xml: Tiene la función de índice, basta con añadir la planta que queramos implementar.

```
<edificio>
        <planta nombre="nombrePlanta">
                <archivo>nombreFichero</archivo>
        </planta>
</edificio>
```

A continuación se detalla cada campo:

- **nombre:** Nombre representativo de la planta que estemos implementando.
- **archivo:** Nombre del fichero .xml de la planta. No es necesario escribir la extensión .xml

NombreFichero.xml: Corresponde a la nueva planta que hayamos añadido. Estará compuesto por estancias, con la siguiente forma:

```
<planta>
   <Z> X <Z>
   <estancia id="estancia1">
   <tipo>{Pasillo, aula, laboratorio, etc}</tipo>
   <cuadrantes>
          <cuadrante idc="0">
                 <SE>
                        <X>3</X>
                        <Y>13</Y>
                 </SE>
                 <NW>
                        <X>0</X>
                        <Y>16</Y>
                 </NW>
                 <conectado>
```

```
                    <norte>-1</norte>
                    <sur>-1</sur>
                    <este>1</este>
                    <oeste>-1</oeste>
                </conectado>
                <objeto>dato0</objeto>
            </cuadrante>
        </cuadrantes>
        </estancia>
    </planta>
```

A continuación se detalla cada campo:

- **Z:** Indica el número de la planta.
- **id:** El identificador que le queramos dar a la estancia.
- **tipo:** Corresponde a los tipos de estancias que tengamos en nuestro edificio. En nuestro caso: Pasillo, aula y laboratorio. Si se desean añadir tipos distintos habría que modificar el fichero Tipos.java que se encuentra dentro de la aplicación servidor.
- **idc:** Identificador de los cuadrantes.
- **SE:** Posición sureste del cuadrante, con sus coordenadas X e Y
- **NW:** Posición noroeste del cuadrante, con sus coordenadas X e Y
- **conectado:** Indican el identificador de los cuadrantes colindantes. El -1 representa la pared.
- **objeto:** Objeto o elemento más representativo del cuadrante.

Estos ficheros deberán estar alojados en el servidor, dentro de una carpeta llamada **'xml'**, en el mismo directorio en el que se encuentre la aplicación .jar.

C.6 ESTABLECER DESTINOS

Todos los posibles destinos dentro de un edificio son almacenados en un fichero JSON. La estructura es la siguiente:

```
[{
"lugar": "sala de juntas",
"cuadrante": "14",
},{
"lugar": "aula 13",
"cuadrante": "15",
}]
```

Donde **lugar** indica la estancia a la que nos referimos, mientras que **cuadrante** indica el identificador del cuadrante que se encuentra a la entrada de esa estancia. Según la numeración que le hayamos dado a nuestros cuadrantes.

El fichero ha de llamarse **destinos.json** y estará alojado en el servidor, en el mismo directorio que la aplicación .jar.

C.7 COMPILAR Y EJECUTAR APLICACIÓN

Por último, con la aplicación servidor ya compilada y corriendo en un .jar, nos falta hacer lo propio con la aplicación cliente.

Con el Eclipse crearemos el **.apk** que enviaremos a nuestro dispositivo Android donde lo instalaremos y ejecutaremos.

BIBLIOGRAFÍA Y REFERENCIAS

[1] E. López Mañas, F. J. Moreno y J. Plá Herrero, AVANTI: Sistema de Asistencia a la Evacuación de Incendios. Proyecto SSII, Madrid: Facultad de Informática, Universidad Complutense de Madrid, 2010.

[2] Mariana Martín-Calderín de la Villa, Sistema de guía por voz en interiores. TFG, Madrid: Facultad de Informática, Universidad Complutense de Madrid, 2013.

[3] NAVSTAR, 1993. Global Positioning System Standard Positioning Service signal specification. Dept. of Defense rept., November, 1993, 46pp + appendices.

[4] Luis Fernández, Daniel Hernández.Revista de la Facultad de Ingeniería Universidad Central de Venezuela. v20 n.3 Caracas julio 2005

[5] Frédéric Evennou, François Marx. Advanced integration of WIFI and inertial navigation systems for indoor mobile positioning. EURASIP Journal on Applied Signal Processing volume 2006, 01 January. P164-164. 2006.

[6] Ramsey, B.W.; Mullins, B.E.; White, E.D. Improved tools for indoor ZigBee warwalking. Local Computer Networks Workshops (LCN Workshops), 22-25 Oct 2012 IEEE 37th Conference on, 2012. P 921-924

[7] Enge, P. Walter, T. ; Pullen, S. ; Changdon Kee ; Yi-Chung Chao ; Yeou-Jyh Tsai. Wide area augmentation of the Global Positioning System. Proceedings of the IEEE (Volume:84 , Issue: 8). Aug 1996. P 1063-1088

[8] L. Gauthier, P. Michel & J. Ventura-Traveset, J. Benedicto. EGNOS: The First Step in Europe's Contribution to the Global Navigation Satellite System. Bulletin 105 – february 2001

[9] C. Rizos, "Network RTK Research and Implementation - A Geodetic Perspecti," *Positioning*, Vol. 1 No. 2, 2002, pp. -.

[10] Manual Android: Server: http://developer.android.com/reference/java/net/Socket.html ServerSocket: http://developer.android.com/reference/java/net/ServerSocket.html

[11] E. W. Dijstra. A Note on Two Problems in Connexion with Graphs. Numerische Mathematik 1, 269-271 (1959)

[12] Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2001). "Section 24.3: Dijkstra's algorithm". *Introduction to Algorithms* (Second ed.). MIT Press and McGraw–Hill. pp. 595–601. ISBN 0-262-03293-7.

[13] Cerf, V.; Kahn, R. (1974). «A Protocol for Packet Network Intercommunication». *IEEE Transactions on Communications* COM-22 (5): pp. 637-648.

[14] Software Quality Assurance Subcommittee of the Nuclear Weapons Complex Quality Managers under the United States Department of Energy, SQAS21.01.00-1999.

[15] David W. Aha. Editorial. Lazy Learning. pp 7-10. 1997. Artificial Intelligence.

[16] Homer Dudley and T.H. Tarnoczy. The Speaking Machine of Wolfgang von Kempelen. *Journal of the Acoustical Society of America*, March 1950, Volume 22, Issue 2, pp. 151–166.

[17] Standage, Tom, *The Turk: The Life and Times of the Famous Eighteenth-Century Chess-Playing Machine*, New York: Walker & Company, 2002: pp 76-81

[18] H. Dudley. *Bell Labs Record 17(2):122--126* (*1939*)

[19] HATON, J. P. (1985): «Knowledge-Based and Expert Systems in Automatic Speech Recognition». En New Systems and Architectures for Automatic Speech Recognition and Synthesis. R. DeMori y Ch. Y. Suen (eds.) NATO-ASI Series. Springer Verlag, págs. 249-270.

[20] MARIANI, J. (1989): «Recent Advances in Speech Processing». Proc. IEEE Int. Conf. on Acoustic, Speech and Signal Processing - 1989, págs. 429-440.