Departamento de Informática

PROYECTO FIN DE CARRERA
Ingeniería en Informática

GEOLOCALIZACIÓN EN INTERIORES

Autor: Iñigo Mariño Lizuain

Tutor: Ángel García Crespo

Leganés, Julio de 2013

Agradecimientos

Primero de todo, he de dar las gracias a mis compañeros de universidad durante todos estos años y en especial a Peri. Sin ellos la carrera hubiera sido mucho más dura y no hubiera disfrutado tanto de los largos días de estudio y práctica.

Gracias a mis amigos fieles e incondicionales Ager, Chang, Edu, Lammers, Portilla y Raúl, pues me han prestado su ayuda y apoyo y han conseguido hacer de la vida universitaria la mejor de mis etapas en la vida.

Es necesario mencionar a mi familia, pues sin su ánimo y supervisión me hubiera relajado más de lo necesario. Es vital tener una base sólida y estricta para acometer los objetivos más importantes.

Por último, quiero dar gracias a la persona más importante durante mi etapa universitaria. Gracias Alexandra por estar siempre ahí, por alentarme, ser mi sombra y hacerme sentir capaz de todo.

Gracias a todos.

Resumen

El principal propósito de este proyecto es realizar una aplicación mediante la cual podamos guiarnos en el interior de todo tipo de edificios. De por sí, este ya supone un gran reto a cumplir, pero se complica un poco más al añadir las restricciones que impiden el uso de tecnologías *GPS, Wi-Fi* y datos móviles.

Este desarrollo pretende ayudar a evolucionar las actuales vías de investigación y ofrecer una alternativa útil a los usuarios que desean conocer su ubicación en determinadas ubicaciones interiores con una gran precisión.

Hoy en día las tecnologías aplicadas pasan por la utilización de la ubicación mediante satélites e internet, aspecto limitador en interiores, pues la precisión disminuye en el mejor de los casos y es inexistente en el peor (*GPS*).

Al final del documento queda plasmado que se ha encontrado una solución que da buenos resultados y está al alcance de cualquier usuario de Android de forma gratuita. La única restricción que existe está en el abanico de rutas y edificios proporcionados, aspecto que queda en manos del desarrollador de la aplicación.

Es posible cambiar este último aspecto en futuras mejoras de la aplicación.

Palabras clave: Android, aplicación móvil, ubicación, edificio, posición, campo magnético, intensidad de señal móvil.

Abstract

The main purpose of this final year project is to develop an application through which we can be guided inside all types of buildings. In itself, this already represents a major challenge to meet, but it is a bit more complicated to add restrictions that prevent the use of *GPS* technology, *Wi-Fi* and mobile data.

This development aims to help evolve the current lines of research and provide a useful alternative for users who want to know their location in certain properties with great precision.

Nowadays, used technologies pass through location using satellite and the Internet, both ways limited on indoors. The accuracy decreases on the best case and is nonexistent at worst (*GPS*).

At the end of the document is reflected it has been found a solution that works well and can be used by any user of Android free of charge. The only restriction is that the range of routes and buildings provided is up to the application developer.

This last point can be changed in future application enhancements.

Keywords: Android, mobile application, location, building, position, magnetic field, signal strenght.

Índice General

Índice de Ilustraciones

Índice de Tablas

1. Introducción y Objetivos

1.1 Introducción

El objetivo de este apartado consiste en exponer el contenido de esta memoria para que constituya una referencia inicial en el contenido total de la misma. Esta sección constituye una breve introducción de los conceptos que se explican en las sucesivas secciones del documento.

El objetivo del proyecto consiste en resolver la dificultad actual en la localización en el interior de edificios. La meta del proyecto es utilizar un nuevo método de ubicación, mediante la utilización de nuevas herramientas que no se ven alteradas por la ausencia de señales *WI-FI* o *GPS*. Por supuesto, cualquier usuario estará en disponibilidad de usar la aplicación.

A lo largo de los años, la demanda de nuevos servicios por parte de los usuarios ha aumentado debido a los avances realizados. Entre éstos podemos añadir la disponibilidad de internet en el teléfono móvil, un *GPS*, así como diversos sensores que han sido añadidos a los dispositivos de última generación. De este modo, gracias a éstas herramientas han surgido nuevas aplicaciones que nos han permitido realizar grandes avances en la comunicación social.

Hoy en día ésta comunicación social hace casi indispensable poseer una aplicación para enviar mensajes en tiempo real, leer el correo electrónico desde cualquier sitio y conocer nuestra ubicación actual en un determinado momento. El problema de la ubicación parece haber llegado a un punto óptimo en lo concerniente a espacios exteriores. Sin embargo, en lo referente a la localización en interiores aún no se ha llegado a una solución tan conveniente.

Este proyecto pretende ofrecer una solución a este problema mediante una aplicación *Android* que sea capaz de localizarnos dentro de un edificio, sea cual sea nuestra ubicación.

1.2 Motivaciones

Los cambios sufridos en la evolución de la sociedad han propiciado la creación de nuevos terminales móviles que poseen una tecnología potente al alcance del usuario en todo momento. Este proyecto se presenta como solución a uno de los muchos problemas derivados de las necesidades sociales: la localización en interiores.

Como ejemplos de las deficiencias que elimina este proyecto, tenemos las siguientes situaciones comunes:

- Los actuales sistemas de localización no tienen casi efecto dentro de los edificios, debido a que sus instrumentos no son válidos en interiores. Es por ello que este proyecto utiliza nuevas medidas que ayudan a subsanar el problema.

- Además de no ser eficientes, lo sistemas de localización actuales consumen una gran cantidad de recursos (véase consumo de datos móviles, batería, etc.), problema que no afectaría a este proyecto, basado en nuevos recursos.

Las deficiencias existentes son numerosas y gracias al avance de los nuevos terminales, podemos ofrecer una solución satisfactoria y que optimice recursos.

La motivación principal del proyecto consiste en solventar el mayor número de errores existentes en los anteriores sistemas encargados de la localización en interiores.

1.3 Objetivos

El objetivo del proyecto de presentado es el siguiente:

- Desarrollar una aplicación que nos muestre en todo momento nuestra ubicación dentro de un edificio.

A este objetivo principal se unen otros secundarios que nos ayudan a dividirlo en secciones:

- Especializarse en el dominio de *Android*.
- Crear una interfaz cómoda y sencilla para el usuario.
- Diseñar una base de datos capaz de almacenar la información necesaria para que la aplicación funcione de modo correcto.
- Diseñar un prototipo funcional capaz de mostrar los objetivos y las pretensiones del proyecto.

1.4 Posibles aplicaciones del sistema

Cualquier usuario de un terminal con el sistema operativo de *Android* puede usar el sistema siempre y cuando lo desee.

Para empezar a utilizar la aplicación, el usuario solamente debe instalarla en el teléfono y podrá disponer de ella.

En algunas ocasiones es posible que el usuario deba calibrar la brújula del móvil, debiendo seguir los pasos que se especifican más adelante.

Los mapas que podrán ser usados por el usuario para la localización interior serán proporcionados por el desarrollador y por tanto dependen de él las ubicaciones accesibles.

1.5 Estructura de la memoria

La memoria del proyecto seguirá un orden riguroso mediante una serie de secciones o capítulos. Las secciones, junto con sus correspondientes explicaciones, son las siguientes:

1. **Introducción y objetivos:** en esta sección se muestra el problema que se quiere resolver, las motivaciones del proyecto y los objetivos a cumplir.

2. **Estado del arte:** este capítulo expone las tecnologías empleadas con anterioridad y aplicaciones similares. Se centra en el entorno *Android*, usado en este proyecto.

3. **Análisis del sistema:** esta sección presenta las necesidades del sistema para cumplir los objetivos. Para ello se especifican los requisitos de usuarios, requisitos del software y los diferentes casos de uso concebidos.

4. **Diseño del sistema:** a partir del apartado 3, se define la arquitectura que mejor conviene al sistema, así como los componentes que lo componen.

5. **Implementación:** en este apartado se analiza de forma más detallada cada módulo que contiene el proyecto, detallado en el punto 4. La forma de abordar la implementación y las alternativas junto con sus razonamientos, se exponen aquí.

6. **Plan de Pruebas:** en esta sección se recogen las pruebas de aceptación que se han aplicado al proyecto para confirmar su óptima implementación de acuerdo con los casos de uso.

7. **Planificación:** contiene la planificación del proyecto, realizada mediante la herramienta *Microsoft Project.*

8. **Presupuesto final:** presenta el presupuesto personal asumido para la realización del proyecto.

9. **Conclusiones:** esta sección recopila el grado de satisfacción obtenido al finalizar el proyecto, los problemas encontrados (resueltos o sin resolver), las futuras líneas de investigación y las posibles mejoras del proyecto.

2.Estado del arte

2.1 Introducción

Este apartado muestra un estudio de las distintas tecnologías que se han desarrollado hasta ahora relacionadas con el producto que se desea implementar. Este estudio sirve como base para conseguir un producto final de calidad contrastada.

Lo siguiente que se propone en esta fase del proyecto es una comparativa de las diferentes alternativas existentes en el mercado actual que cubren una funcionalidad similar a la propuesta en este proyecto. En esta comparativa se destacan las ventajas e inconvenientes de cada producto para mostrar los puntos fuertes en los que podemos apoyarnos y las debilidades que debemos evitar al abordar un proyecto de estas características.

Por último, después de haber estudiado las tecnologías actuales y los productos existentes en el mercado, seremos capaces de determinar si el desarrollo propuesto es factible en la actualidad.

2.2 Android

Android es un sistema operativo diseñado para dispositivos móviles. Está basado en *GNU/Linux* e inicialmente fue desarrollado por *Open Handset Alliance*, liderada por *Google*. La presentación de la plataforma *Android* se realizó el 5 de Noviembre de 2007 junto con la fundación de la propia *Open Handset Alliance*, que constituye un consorcio de 48 compañías de hardware, software y telecomunicaciones comprometidas con la promoción de estándares abiertos para dispositivos móviles.

Esta plataforma permite el desarrollo de aplicaciones por particulares ajenos a *Google*, para lo cual es necesario el empleo del lenguaje *Java* y de bibliotecas específicas proporcionadas o adaptadas por *Google*. Está plataforma se puede considerar a de código abierto, lo que significa que cualquier desarrollador puede crear y desarrollar aplicaciones escritas con lenguaje *C* u otros lenguajes y compilarlas a código nativo de *ARM*.

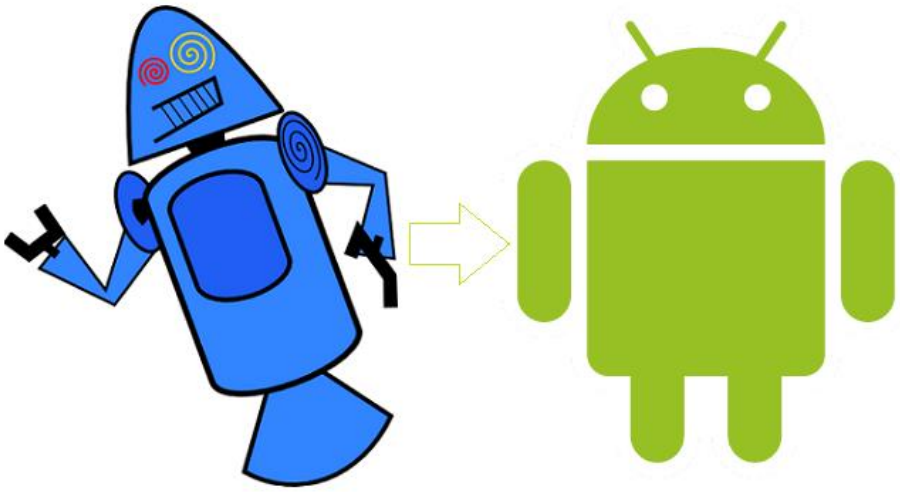

Ilustración 1: Primer logo de *Android* y su evolución actual.

2.2.1 Plataformas de desarrollo

A la hora de realizar el proyecto se han tenido en cuenta las numerosas plataformas en las cuales se puede desarrollar. El mercado cuenta con muchas por lo que esta elección ha de realizarse de acuerdo a las necesidades Hardware y Software que tenga nuestra aplicación.

Las mayores plataformas son *Android* e *iOS* pues son las que ocupan mayor mercado actualmente, las más desarrolladas y las que poseen un mayor grado de progresión en el presente. A esto hay que añadir que ambas colaboran de forma impecable con cualquier servicio de telecomunicación y base de datos y poseen una sencillez y una interfaz preparadas para cualquier sector del público.

Por todo ello, dentro de las opciones a tener en cuenta para el desarrollo del proyecto hemos descartado el resto de sistemas operativos para móvil como *Windows Phone 7, Symbian, RIM* o *Webos*.

Una vez escogidos estos dos sistemas, hay que sopesar las ventajas e inconvenientes de cada uno.

En primer lugar, los principales inconvenientes de *Android* consisten en el gran número de dispositivos que existen con este sistema operativo (*Samsung, HTC, LG,* etc.) y las numerosas versiones de éste que existen. La existencia de un gran número de dispositivos móviles es perjudicial para el desarrollador pues existen algunos que poseen más sensores que otros o una pantalla más grande y con mayor resolución, lo que puede dar lugar a problemas de adaptación de la aplicación.
Por otro lado, *Android* supone desarrollar aplicaciones en lenguaje *Java* y algo de *XML*, ambos estudiados de forma exhaustiva durante la carrera, lo que ha hecho que sea bastante más sencillo el proceso de implementación.

En segundo lugarel principal inconveniente de *iOS* es que el lenguaje necesario para desarrollar aplicaciones es *Objective-C*. Éste me resulta completamente desconocido, por lo que a la hora de comenzar necesitaría de un tiempo previo de estudio y aprendizaje muy costoso. Además, *Objective-C* resulta bastante engorroso y difícil a la hora de modificar parámetros relacionados con el *Wi-Fi,* el *GPS* y otros sensores.
La principal ventaja de *iOS* consiste en la existencia de un terminal único y con unas dimensiones únicas, de forma que el tema de compatibilidad de interfaz no sería un problema.

Una vez analizadas las dos plataformas en detalle, he considerado que la elección que se ajusta más a las necesidades del proyecto y a mis conocimientos es *Android.*

El terminal de prueba empleado para el test de la aplicación puede ser casi cualquiera, pues las necesidades no son muy acotadas. Debido a la disponibilidad inmediata, el dispositivo usado es el *Samsung Galaxy Ace*, con versión del sistema operativo *Android 2.3.*

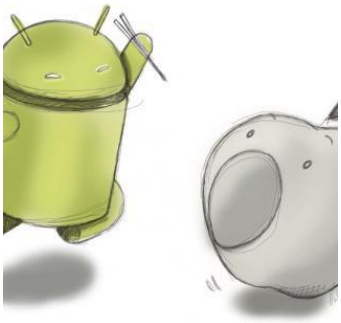

Ilustración 2: *Android* vs. *IOS.*

2.2.2 Introducción a Android

Android es un Sistema Operativo, además de una plataforma de Software, basado en el núcleo de Linux. Su diseño inicial iba dirigido al uso en dispositivos móviles y tabletas, ampliándose posteriormente a *Google TV*, reproductores de música, cámaras de fotos, etc. *Android* permite controlar dispositivos por medio de bibliotecas desarrolladas o adaptadas por *Google* mediante el lenguaje de programación *Java.*

Su estructura está compuesta por una serie de aplicaciones que se ejecutan en un *framework Java* de aplicaciones orientadas a objetos sobre el núcleo de las bibliotecas de *Java*, en una máquina virtual *Dalvik* con una compilación en tiempo de ejecución. Las bibliotecas (escritas en lenguaje *C*) se compilan a código nativo de *ARM*. Éstas poseen un administrador de interfaz gráfica, un *framework OpenCore*, una base de datos *SQLite* relacional, una interfaz de programación de API gráfica *OpenGL*, un motor de renderizado *WebKit*, un motor gráfico *SGL*, *SSL* y una biblioteca estándar de *C Bionic*.

2.2.3 Arquitectura

Android posee una arquitectura muy característica, compuesta de cinco componentes:

- **Aplicaciones:** las aplicaciones base incluyen un cliente de correo electrónico, sistema de *SMS*, calendario, mapas, navegador, contactos y muchas más. Todas ellas han sido escritas en lenguaje *Java.*

- ***Framework* de aplicaciones:** los desarrolladores tienen acceso a todos los Apis del *framework* utilizados por las aplicaciones base. La arquitectura posee un diseño mediante el cual se simplifica la reutilización de los componentes. Mediante una serie de reglas de seguridad, toda aplicación puede publicar sus capacidades y puede hacer uso de las capacidades de otras. Una capa de servicios disponible para las aplicaciones incluye lo siguiente:
  - Un conjunto de completo de vistas que pueden ser utilizadas para desarrollar una aplicación: listas, cajas de texto, botones *radio buttons*….
  - Proveedores de contenidos que permiten acceder a datos que pertenecen a otras aplicaciones y compartir los suyos.
  - Un administrador de recursos que permite el acceso a éstos.
  - Un administrador de notificaciones que provee a las aplicaciones de la capacidad de notificar alertar en la barra de notificaciones.
  - Un administrador de actividades que se encarga de gestionar el ciclo de vida de las aplicaciones.

- **Librerías:** *Android* incluye una serie de librerías escritas en *C* y *C++* usadas por numerosos componentes del sistema. Son accesibles a los desarrolladores mediante el *framework* de aplicaciones de *Android*. Algunas de ellas son: SQLite, librerías de gráficos, librerías de medios y librería de *C* estándar entre otras.

- **Android Runtime:** *Android* incluye una serie de librerías base que ofrecen la mayoría de las funcionalidades disponibles en las de *Java*. Cada aplicación corresponde a un proceso con su propia instancia de la máquina virtual *Dalvik.* Ésta ha sido diseñada de forma que un dispositivo puede ejecutar varias instancias de la misma de un modo eficiente. *Dalvik* ejecuta archivos en el formato propio *.dex*, optimizado para memoria mínima. La máquina virtual ejecuta clases compiladas en este formato por la herramienta *dx.*

- **Kernel de Linux:** la versión correspondiente a esta capa es la 2.6, similar a la incluida en cualquier distribución de Linux, con algunos cambios adaptados a las características de los dispositivos móviles. El kernel se encarga de gestionar los recursos del dispositivo y el sistema operativo, actuando como una capa de abstracción entre el hardware y el dispositivo. Con ello el desarrollador no accede directamente a esta capa, si no que usa una serie de librerías disponibles en capas superiores. Es una forma útil de evitar tener que conocer todas las características hardware.

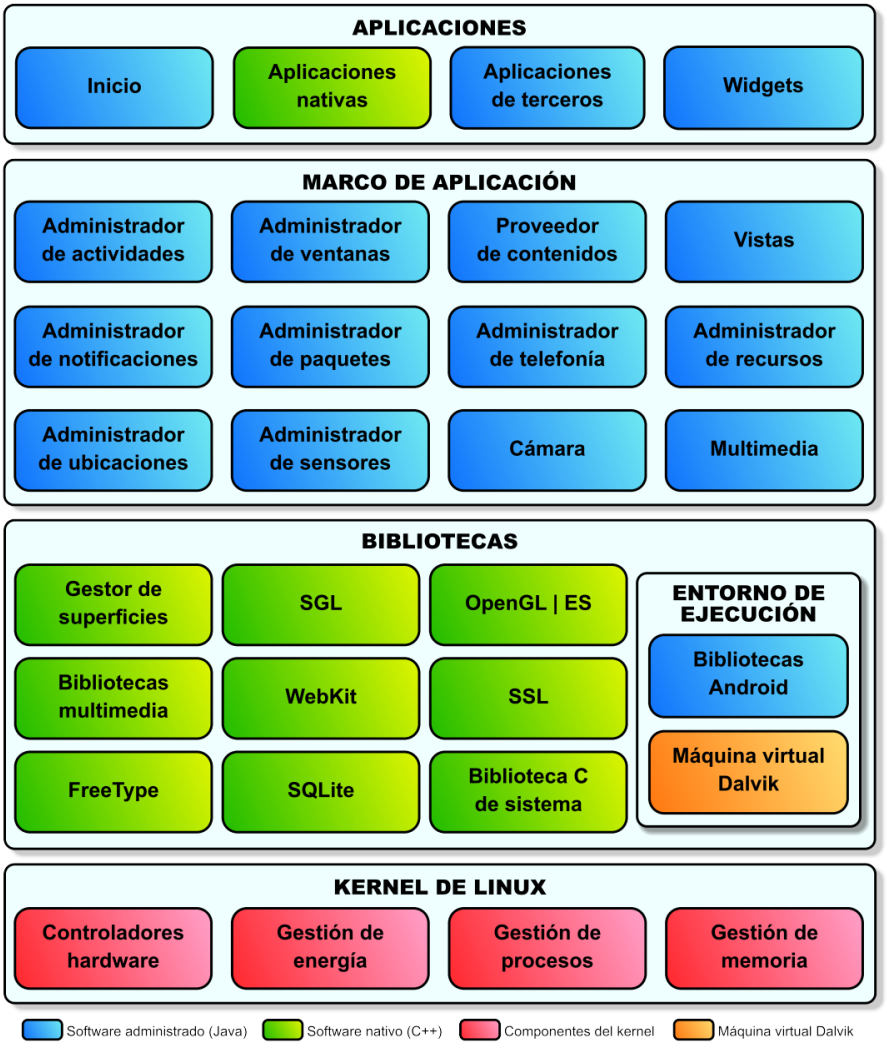

Ilustración 3: Arquitectura de Android.

2.2.4 Versiones de Android

Desde que se realizó su liberación, *Android* ha tenido una gran cantidad de actualizaciones que han provisto de mejoras y corrección de errores al sistema. Las versiones de *Android*, desde su inicio, se desarrollan con el nombre de un postre en inglés. En cada versión el postre elegido empieza por una letra distinta siguiendo un orden alfabético:

Android 1.0: Apple pie 	- Liberada el 23 de Septiembre de 2008. - Características: o Aplicaciones: Gmail, Google Calendar, Google Maps, Google Talk, etc. o Navegador de internet con zoom. o Android Market. o Soporte para cámara. o Acceso a email. o Soporte para WI-FI y Bluetooth. o Soporte para SMS y MMS. o Reproductor multimedia. o Soporte para notificaciones.
Android 1.1: Banana Bread 	- Liberada el 9 de Febrero de 2009. - Características: o Nuevos detalles de Google Maps. o Archivos adjuntos en mensajes. o Visibilidad opcional del teclado durante las llamadas.
Android 1.5: Cupcake 	- Liberada el 30 de Abril de 2009. - Características: o Posibilidad de grabar y reproducir videos a través del modo camcorder. o Capacidad de subir videos a YouTube e imágenes a Picasa. o Un nuevo teclado con predicción de texto. o Soporte para Bluetooth A2DP y AVRCP. o Capacidad de conexión automática para conectar auricular Bluetooth. o Nuevos widgets y carpetas que se pueden colocar en las pantallas de inicio. o Transiciones de pantalla animadas.

Tabla 1: Versiones de Android (1).

Android 1.6: Donut 	- Liberada el 15 de Septiembre de 2009. - Características: o Renovación de Android Market o Una interfaz integrada de cámara y galería. o Selección múltiple de fotos para eliminarlas. o Búsqueda por voz actualizada, con respuesta más rápida y mayor integración con aplicaciones nativas. o Opción de buscar marcadores, historiales, contactos y páginas web desde la pantalla de inicio. o Actualización de soporte para CDMA/EVDO, 802.1x, VPN y text-to-speech. o Soporte para resolución de pantalla WVGA. o Mejoras de velocidad en las aplicaciones de búsqueda y cámara. o Framework de gestos y herramienta de desarrollo GestureBuilder. o Navegación gratuita turn-by-turn de Google.
Android 2.0/2.1: Éclair 	- Liberada el 26 de Octubre de 2009. - Características: o Velocidad de hardware optimizada. o Soporte para más tamaños de pantalla y resoluciones. o Interfaz de usuario renovada. o Nuevo interfaz de usuario en el navegador y soporte para HTML5. o Nuevas listas de contactos. o Mejor relación de contraste para los fondos. o Mejoras en Google Maps 3.1.2. o Soporte para Microsoft Exchange. o Soporte integrado de flash para la cámara. o Zoom digital. o MotionEvent mejorado para captura de eventos multi-touch. o Teclado virtual mejorado. o Bluetooth 2.1. o Fondos de pantalla animados. - El SDK 2.0.1 fue liberado el 3 de diciembre de 2009. - El SDK 2.1 fue liberado el 12 de enero de 2010.

Tabla 2: Versiones de Android (2).

Android 2.2: Froyo 	- Liberada el 20 de Mayo de 2010. - Características: o Optimización del sistema Android, la memoria y el rendimiento. o Mejoras en la velocidad de las aplicaciones, gracias a la implementación de JIT. o Integración del motor JavaScript V8 del Google Chrome. o Soporte mejorado de Microsoft Exchange (reglas de seguridad, reconocimiento automático, GAL look-up, sincronización de calendario, limpieza remota). o Lanzador de aplicaciones mejorado con accesos directos a las aplicaciones de teléfono y Browser. o Funcionalidad de Wi-Fi hotspot y tethering por USB. o Permite desactivar el tráfico de datos a través de la red del operador. o Actualizaciones automáticas del Market. o Cambio rápido entre múltiples idiomas de teclado y sus diccionarios. o Marcación por voz y compartir contactos por Bluetooth. o Soporte para contraseñas numéricas y alfanuméricas. o Soporte para campos de carga de archivos en el navegador. o Soporte para la instalación de aplicación en la memoria SD. o Soporte para Adobe Flash 10.1. o Soporte para pantallas de alto número de Puntos por pulgada.

Tabla 3: Versiones de Android (3).

Android 2.3: Gingerbread 	- Liberada el 6 de Diciembre de 2010. - Características: o Soporte para dispositivos móviles. o Actualización del diseño de la interfaz de usuario. o Soporte para pantallas extra grandes y resoluciones WXGA y mayores. o Soporte nativo para telefonía VoIP SIP. o Soporte para reproducción de videos WebM/VP8 y decodificación de audio AAC. o Nuevos efectos de audio. o Soporte para Near Field Communication. o Funcionalidades de cortar, copiar y pegar. o Teclado multi-táctil rediseñado. o Soporte mejorado para desarrollo de código nativo. o Mejoras en la entrada de datos, audio y gráficos. o Recolección de elementos concurrentes para un mayor rendimiento. o Soporte nativo para más sensores. o Un administrador de descargas para archivos grandes. o Administración de la energía mejorada y administrador de tareas. o Soporte nativo para múltiples cámaras. o Cambio de sistema de archivos de YAFFS a ext4.
Android 3.0/3.1/3.2: Honeycomb 	- Liberada el 22 de Febrero de 2011. - Características: o Soporte para tablets. o Escritorio 3D con widgets rediseñados. o Sistema multitarea mejorado o Mejoras en el navegador web. o Soporte para videochat mediante Google Talk. o Mejor soporte para redes WI-FI. o Añade soporte para una gran variedad de periféricos y accesorios con conexión USB. o Los widgets pueden redimensionarse de forma manual sin límite. o Se añade soporte opcional para redimensionar correctamente las aplicaciones inicialmente creadas para móvil para que se vean bien en Tablets.

Tabla 4: Versiones de Android (4).

Android 4.0: Ice Cream Sandwich 	- Liberada el 19 de Octubre de 2011. - Características: o Versión que unifica el uso en cualquier dispositivo, tanto en teléfonos, tablets, televisores, netbooks, etc. o Opción de utilizar los botones virtuales en la interfaz de usuario. o Aceleración por hardware. o Multitarea mejorada. o Gestor del tráfico de datos de internet. El entorno le permite establecer alertas y desactivar los datos cuando se pasa de su límite. o Los widgets están en una nueva pestaña. o El corrector de texto ha sido rediseñado y mejorado. o Se pueden descartar las notificaciones no importantes y desplegar la barra de notificaciones con el dispositivo bloqueado. o La captura de pantalla, con solo pulsando el botón de bajar volumen y el botón de encendido. o Posibilidad de hacer fotografías panorámicas de forma automática. o Android Beam es la nueva característica que nos permitirá compartir contenido entre teléfonos. Vía NFC. o Reconocimiento de voz del usuario. o Buzón de voz visual que le permite adelantarlo o retroceder los mensajes de voz. o Reconocimiento facial. o Las carpetas son mucho más fáciles de crear, con un estilo de arrastrar y soltar. o Un único y nuevo framework para las aplicaciones. o Soporte nativo del contenedor MKV. o Soporte nativo para el uso de Stylus (lápiz táctil).
Android 4.1: Jelly Bean 	- Liberada el 9 de Julio de 2012. - Características: o Mejora de la interfaz de usuario. o Posibilidad de apagar las notificaciones de una aplicación. o Compartir archivos por bluetooth (Android Beam). o Dictado por voz sin conexión a internet. o Mejora en la cámara. o Mejora en el dictado por voz. o Google Wallet. o Fotos de contactos de alta resolución.

Tabla 5: Versiones de Android (5).

2.3 Aplicaciones similares

En la actualidad existen dos alternativas similares al proyecto que se está realizando. La primera de ellas viene de la mano de *Google* con su actualización de *Google Maps* a la versión 6.0, y la segunda corresponde a un proyecto llevado a cabo por la empresa finlandesa *IndoorAtlas*.

Maps con su nueva versión nos ofrece una serie de mapas correspondientes a centros comerciales y aeropuertos. Por ahora esta versión sólo se encuentra disponible en los Estados Unidos y en Japón. Por supuesto, para el uso de esta tecnología es necesario poseer datos móviles, *WI-FI* o *GPS*.
Una de las ventajas que posee esta alternativa es que si subimos o bajamos a una planta diferente, *Google* se encarga de actualizar automáticamente la pantalla y situarnos en el nuevo mapa correspondiente.

La empresa *IndoorAtlas* ha querido ir más allá a la hora de posicionarnos en el interior de edificios. El sistema empleado para el funcionamiento de la aplicación consiste en el campo magnético propio de los interiores. Cada habitación, pasillo o tramo de escaleras posee una variación particular del campo magnético, lo que permite posicionarnos con una precisión mucho mayor que los sistemas basados en internet o satélite. La parte mala de esta gran novedad es que hay que recorrer cada pasillo del edificio realizando las medidas del campo magnético.

La parte buena es que la aplicación está diseñada para que cualquier usuario pueda crear un plano del interior de un edificio y realizar las medidas pertinentes para que cualquier otra persona pueda utilizarlo. Un API dedicado nos ayuda a la hora de diseñar los planos, subirlos al servidor y aplicar las medidas seleccionadas. De este modo, no sólo ganamos en precisión, sino que además poseemos una aplicación completamente accesible y en la que podemos compartir nuestros propios edificios.

Si hay una pega que poner a este último proyecto, es que aún no se encuentra disponible en el mercado y no tiene una fecha concreta de lanzamiento.

La conclusión de estas aplicaciones es que, la primera de ellas, *Google Maps*, no posee la capacidad de colaboración por parte del resto de los usuarios, que por otro lado, no poseen la infraestructura de la empresa para realizar dichas aplicaciones.

En cuanto a *IndoorAtlas*, su proyecto resulta innovador e inspirador, pues sí parece estar al alcance de cualquier persona realizar una aplicación de estas características. De hecho, el proyecto presente se inspira completamente en esta última opción.

3. Análisis del sistema

3.1 Introducción

El análisis del sistema que se define en esta sección está basado en uno de los estándares recogidos por la *European Space Agency* o *ESA:* su versión *Lite* para proyectos de Ingeniería del Software. Esta opción es la más correcta y adecuada pues éste es un proyecto poco denso y no requiere de estándares extensos como la versión completa de la *ESA* o *IEEE*. Por supuesto, el estándar no se ha seguido en su totalidad, pues algunas secciones han quedado fuera del proyecto y otras han sido modificadas de acuerdo a la armonía del trabajo.

De acuerdo al seguimiento propuesto, se establecerán las funcionalidades requeridas por el sistema y junto con ellas se analizarán las consecuentes capacidades y restricciones que poseerá el sistema.

Por último, en un análisis riguroso se expondrán los requisitos de usuario, los cuáles recogen todo el funcionamiento del sistema y que, junto con los casos de uso que reflejarán también, constituyen la base de los requisitos de software.

3.2 Descripción general

Esta sección pretende dejar establecidas cada una de las características del sistema, de manera que en el punto 4.0 se especifique su funcionamiento de forma plena. Se definirán las capacidades del sistema, sus restricciones, los tipos de usuarios que el sistema incluye y el entorno en el que se ejecutará.

3.2.1 Capacidades del sistema

Las capacidades que debe cumplir el sistema respecto a los usuarios son:

- Cualquier usuario puede utilizar el sistema, sin necesidad de login.
- El usuario puede consultar la información y el correo de contacto.
- Se puede seleccionar cualquiera de los edificios que se encuentren en la lista de localizaciones disponibles.
- El usuario puede conocer su posición actual aproximada en el mapa, siempre que se encuentre en la localización correspondiente a ese mapa.

3.2.2 Restricciones del sistema

Las restricciones que posee la implementación del sistema son:

- Los mapas disponibles depende únicamente del desarrollador de la aplicación y ningún usuario puede añadir nuevos.
- La base de datos está hecha en *XML* debida a la poca extensión, pudiendo ser cambiada en un futuro.
- Es necesario que el usuario tenga la brújula bien calibrada, pues si no lo está, las medidas tomadas por su dispositivo no devolverían una ubicación adecuada.
- Cada vez que el usuario se encuentre en un edificio con un mapa disponible, debe seleccionarlo en la lista de ubicaciones disponibles.
- El usuario debe mantenerse en movimiento para obtener la posición exacta.
- En caso de edificios con varias plantas, existe un mapa por cada una de ellas.
- La aplicación se encuentra íntegramente en castellano.

3.2.3 Usuarios del sistema

En el sistema existe un único tipo de usuario, que podrá acceder a las capacidades anteriormente descritas. Para ello no es necesario que posea un conocimiento adicional aparte del manejo de su dispositivo *Android*.

3.2.4 Entorno operacional

La tecnología que requiere el sistema es:

- Un dispositivo *Android* con la versión 2.3 *Gingerbread* como mínimo.
- Es necesario que la aplicación esté instalada en el dispositivo.
- El dispositivo debe tener brújula digital y estar calibrada.
- La base de datos estará en formato *XML*.

3.3 Requisitos de usuario

Los requisitos de usuarios se especifican con el objetivo de cubrir toda la funcionalidad que requiere el sistema. Los requisitos de usuarios son de dos tipos: de capacidad y de restricción.

Los requisitos de capacidad nos indican qué es lo que debe hacer el sistema y los requisitos de restricción explican cómo debe hacerlo.

El formato seguido para los requisitos de usuario se muestra en la Tabla 6: Formato de Requisito de Usuario. :

Identificador	
Descripción	
Necesidad	☐ Alta ☐ Media ☐ Baja
Prioridad	☐ Alta ☐ Media ☐ Baja
Origen	
Grado de Verificabilidad	

Tabla 6: Formato de Requisito de Usuario.

Cada campo tiene el siguiente significado:

- **Identificador:** identificador único para cada requisito. Está formado por las siglas RU seguidas de una C, si es un requisito de capacidad, o de una R, si es un requisito de restricción. Después de las siglas, se añade un número que establece el orden.

- **Descripción:** explica de forma precisa el significado del requisito.

- **Necesidad:** representa la importancia del requisito según las especificaciones del cliente.

- **Prioridad:** establece un orden prioritario para el requisito respecto al resto.

- **Origen:** hace referencia al origen del requisito.

- **Grado de Verificabilidad:** indica la medida en que puede comprobarse que el requisito está cumplido.

3.3.1 Requisitos de capacidad

Los requisitos de capacidad establecen las funciones que debe realizar el sistema para satisfacer una serie de objetivos. En otras palabras, definen el comportamiento que ha de seguir el software.

Identificador	RUC-01
Descripción	Se permite iniciar la aplicación al usuario donde accederá a la lista de ubicaciones disponibles
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de Verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 7: RUC-01.

Identificador	RUC-02
Descripción	Se permite al usuario seleccionar cualquier ubicación de las presentes en la lista mostrada
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de Verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 8: RUC-02.

Identificador	RUC-03
Descripción	Se muestra en la pantalla el mapa correspondiente a la ubicación seleccionada
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de Verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 9: RUC-03.

Identificador	RUC-04
Descripción	Se muestra dentro del mapa visualizado la actual aproximada posición del usuario
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de Verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 10: RUC-04.

Identificador	RUC-05
Descripción	Se permite al usuario acceder a la información de la aplicación desde el menú principal
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de Verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 11: RUC-05.

Identificador	RUC-06
Descripción	Se permite al usuario salir del sistema de forma segura
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de Verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 12: RUC-06.

3.3.2 Requisitos de restricción

Los requisitos de restricción conforman las reglas y limitaciones definidas por los usuarios y que precisan la forma en que es resuelto el objetivo establecido. Es una manera de restringir el software sin adentrarse en sus capacidades.

Identificador	RUR-01
Descripción	La aplicación funciona en dispositivos *Android* con una versión igual o superior a la 2.3
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 13: RUR-01.

Identificador	RUR-02
Descripción	El dispositivo debe de tener una brújula digital en su hardware para el correcto funcionamiento de la aplicación
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 14: RUR-02.

Identificador	RUR-03
Descripción	La localización del usuario dentro del mapa especificado se obtiene mediante la brújula digital
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 15: RUR-03.

Identificador	RUR-04
Descripción	La dirección en la que se mueve el usuario dentro del mapa especificado se obtiene mediante la brújula digital
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 16: RUR-04.

Identificador	RUR-05
Descripción	Para el correcto funcionamiento es necesario que el usuario se encuentre en movimiento, de lo contrario puede no señalarse la posición correcta
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 17: RUR-05.

Identificador	RUR-06
Descripción	El sistema se encuentra diseñado íntegramente en castellano
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Origen	Cliente
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 18: RUR-06.

3.4 Casos de uso

Un caso de uso representa una secuencia de interacciones que existen entre un sistema y los actores, como respuesta a un evento que un actor inicia sobre el sistema. Los diagramas representan el comportamiento que sigue el sistema según su interacción con los actores y otros sistemas.

El objetivo de esta sección consiste en representar las tareas que puede realizar un usuario con la aplicación, partiendo de la premisa de que es el usuario el que toma el control del sistema mediante su dispositivo móvil.

El formato seguido para los requisitos de usuario se muestra en la Tabla 19: Formato de Caso de Uso. :

Identificador	
Nombre	
Actores	
Objetivo	
Descripción	
Pre-condición	
Post-condición	

Tabla 19: Formato de Caso de Uso.

Cada campo tiene el siguiente significado:

- **Identificador:** identificador único para cada caso de uso. Está formado por las siglas CU seguidas de un número que establece el orden.
- **Nombre:** constituye el nombre asignado al caso de uso.
- **Actores:** diferentes roles que participan en la interacción.
- **Objetivo:** finalidad del caso de uso.
- **Descripción:** descripción escrita del caso de uso representado.
- **Pre-condición:** especificación detallada de la situación previa a la interacción.
- **Post-condición:** especificación detallada de la situación posterior a la interacción.

Los casos de uso extraídos son:

Identificador	CU-01
Nombre	Inicio del sistema
Actores	Usuario
Objetivo	Iniciar la aplicación
Descripción	El usuario accede a la aplicación y pulsa el botón *Comenzar*
Pre-condición	No se ha iniciado la aplicación
Post-condición	Se muestran las ubicaciones disponibles

Tabla 20: CU-01.

Ilustración 4: CU-01.

Identificador	CU-02
Nombre	Selección de ubicación
Actores	Usuario
Objetivo	Seleccionar la ubicación para ver el mapa correspondiente
Descripción	El usuario visualiza una lista con todas las ubicaciones disponibles y selecciona aquella en la que se encuentra para comenzar la localización
Pre-condición	Se ha iniciado la aplicación
Post-condición	Se accede al mapa correspondiente

Tabla 21: CU-02.

Ilustración 5: CU-02.

Identificador	CU-03
Nombre	Visualizar información
Actores	Usuario
Objetivo	Visualizar la información relacionada con la aplicación
Descripción	Una vez que el usuario accede a la aplicación, pulsa el botón *Info* para acceder a la información
Pre-condición	Se ha accedido a la aplicación
Post-condición	Se muestra la pantalla que contiene la información proporcionada en la aplicación

Tabla 22: CU-03.

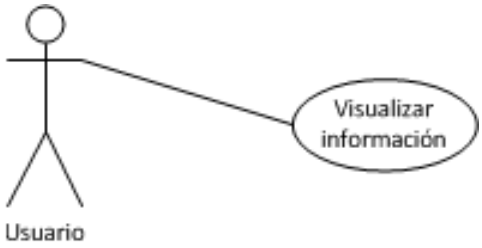

Ilustración 6: CU-03.

Identificador	CU-04
Nombre	Salir de la aplicación
Actores	Usuario
Objetivo	Cerrar la aplicación
Descripción	En el menú principal el usuario puede salir de la aplicación pulsando el botón *Salir*
Pre-condición	Se ha accedido a la aplicación
Post-condición	La aplicación se cierra

Tabla 23: CU04.

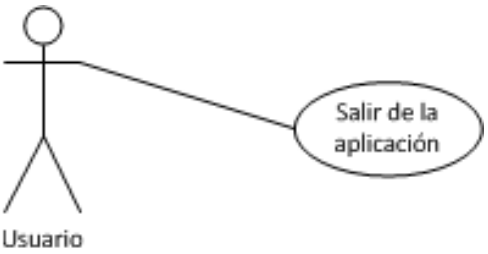

Ilustración 7: CU-04.

3.5 Requisitos software

En esta sección se enumeran todos los Requisitos Software (RS). Éstos definen qué debe hacer el sistema y cómo debe hacerlo. Describen conjuntamente el comportamiento del sistema de forma completa y constituyen una referencia para verificar el posterior diseño y el producto en sí, pues ambos cumplirán los requisitos que se determinan en este apartado.

La plantilla que utilizaremos para recoger todos los Requisitos de Software, será la mostrada en la Tabla 24: Formato de Requisito de Software.:

Identificador			
Descripción			
Necesidad	☐ Alta	☐ Media	☐ Baja
Prioridad	☐ Alta	☐ Media	☐ Baja
Estabilidad	☐ Alta	☐ Media	☐ Baja
Origen			
Grado de verificabilidad	☐ Alto	☐ Medio	☐ Bajo

Tabla 24: Formato de Requisito de Software.

Donde cada campo representa:

- **Identificador:** nombre alfanumérico, unívoco, que identifica al requisito. En caso de ser funcionales o no funcionales irán con F o NF seguidos (RSF y RSNF). Para finalizar, irá acompañado de un número de forma secuencial, para facilitar su búsqueda y ordenación.

- **Descripción:** pequeña definición general del requisito.

- **Necesidad:** importancia o necesidad del requisito en el sistema.

- **Prioridad:** hace referencia al orden temporal en que debe realizarse el requisito.

- **Estabilidad:** indica si el requisito se mantendrá igual a lo largo del proyecto o si por el contrario, podrá ser modificado.

- **Origen:** Indica la fuente del requisito. Puede ser un documento, otro requisito, una persona.

- **Grado de Verificabilidad:** indica la medida en que puede comprobarse que el requisito está cumplido.

3.5.1 Requisitos funcionales

Los Requisitos Software Funcionales (RSF) especifican los servicios que la aplicación debe proporcionar, y son los que se detallan a continuación:

Identificador	RSF-01
Descripción	Se permite iniciar la aplicación al usuario donde accederá a la lista de ubicaciones disponibles
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☐ Alta ☐ Media ☐ Baja
Origen	RUC-01
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 25: RSF-01.

Identificador	RSF-02
Descripción	Se permite al usuario seleccionar cualquier ubicación de las presentes en la lista mostrada
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUC-02
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 26: RSF-02.

Identificador	RSF-03
Descripción	Se muestra en la pantalla el mapa correspondiente a la ubicación seleccionada
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUC-03
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 27: RSF-03.

Identificador	RSF-04
Descripción	Se muestra dentro del mapa la posición aproximada del usuario
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUC-04
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 28: RSF-04.

Identificador	RSF-05
Descripción	Se actualiza la posición del usuario en el mapa tan rápido como permiten los sensores
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUC-04
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 29: RSF-05.

Identificador	RSF-06
Descripción	Se muestra la dirección en la que se mueve el usuario
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUC-04
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 30: RSF-06.

Identificador	RSF-07
Descripción	Se permite al usuario acceder a la información de la aplicación desde el menú principal
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUC-05
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 31: RSF-07.

Identificador	RSF-08
Descripción	Se permite al usuario salir del sistema de forma segura, es decir, borrando la memoria utilizada
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUC-06
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 32: RSF-08.

3.5.2 Requisitos no funcionales

Los requisitos no funcionales imponen restricciones en el producto desarrollado y en el proceso de desarrollo. Los relativos al software indican cómo debe realizar algo el software, y los relativos al hardware indican los componentes físicos del sistema.

HARDWARE

Identificador	RSNF-01
Descripción	El dispositivo debe de tener una brújula digital en su hardware para el correcto funcionamiento de la aplicación
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUR-01
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 33: RSNF-01.

SOFTWARE

Identificador	RSNF-02
Descripción	La aplicación funciona en dispositivos *Android* con una versión igual o superior a la 2.3
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUR-02
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 34: RSNF-02.

Identificador	RSNF-03
Descripción	La localización del usuario dentro del mapa especificado se obtiene mediante la brújula digital
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUR-03
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 35: RSNF-03.

Identificador	RSNF-04
Descripción	La dirección en la que se mueve el usuario dentro del mapa especificado se obtiene mediante la brújula digital
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUR-04
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 36: RSNF-04.

Identificador	RSNF-05
Descripción	Para el correcto funcionamiento es necesario que el usuario se encuentre en movimiento, de lo contrario puede no señalarse la posición correcta
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUR-04
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 37: RSNF-05

Identificador	RSNF-06
Descripción	El sistema se encuentra diseñado íntegramente en castellano
Necesidad	☒ Alta ☐ Media ☐ Baja
Prioridad	☒ Alta ☐ Media ☐ Baja
Estabilidad	☒ Alta ☐ Media ☐ Baja
Origen	RUR-05
Grado de verificabilidad	☒ Alto ☐ Medio ☐ Bajo

Tabla 38: RSNF-06.

3.6 Matrices de trazabilidad

Las matrices de trazabilidad muestran la relación entre requisitos de usuario y requisitos del software. Hay una matriz que define la identificación entre los requisitos de capacidad y los requisitos de software funcionales; la otra matriz defin la correspondencia entre los requisitos de restricción y los requisitos no funcionales.

3.6.1 Matriz de trazabilidad RUC-RSF

RUC/ RSF	RUC-01	RUC-02	RUC-03	RUC-04	RUC-05	RUC-06
RSF-01	X					
RSF-02		X				
RSF-03			X			
RSF-04				X		
RSF-05				X		
RSF-06				X		
RSF-07					X	
RSF-08						X

Tabla 39: Matriz de trazabilidad RUC-RSF.

3.6.2 Matriz de trazabilidad RUR-RNF

RUR/ RSNF	RUR-01	RUR-02	RUR-03	RUR-04	RUR-05	RUR-06
RSNF-01	X					
RSNF-02		X				
RSNF-03			X			
RSNF-04				X		
RSNF-05					X	
RSNF-06						X

Tabla 40: Matriz de trazabilidad RUR-RSNF.

4. Diseño del sistema

4.1 Arquitectura del sistema

Esta sección explica y justifica el diseño arquitectónico que ha sido realizado para la aplicación.

El sistema tiene como base un Modelo Vista Controlador. Éste constituye un patrón o modelo de abstracción de desarrollo de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes. El patrón MVC es comúnmente usado en aplicaciones web, donde la vista está formada por la página HTML junto con el código para datos dinámicos; el modelo es la base de datos y el controlador es el encargado de recibir y procesar los eventos de entrada desde la vista.

Cada uno de los elementos posee sus características propias:

- **Modelo:** es la representación de la información en el sistema. Trabaja junto con la vista para mostrar la información al usuario y es accedido por el controlador para añadir, eliminar, consultar o actualizar datos.
- **Vista:** Es la encargada de presentar el modelo en un formato adecuado para que el usuario pueda interactuar con él, casi siempre es la interfaz de usuario.
- **Controlador:** es el elemento más abstracto. Recibe, trata y responde los eventos enviados por el usuario o por la propia aplicación. Interactúa tanto con el modelo como con la vista.

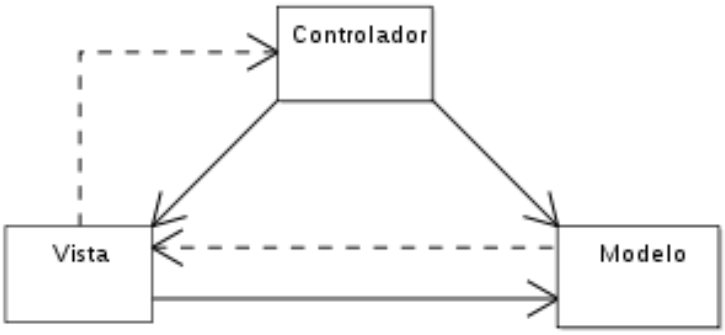

Ilustración 8: Patrón Modelo Vista Controlador.

4.2 Diseño detallado

En este apartado se muestra la especificación de las clases que conforman el sistema y las interfaces de usuario *Android* contenidas en nuestra aplicación.

4.2.1 Especificación de clases

Las clases que forman parte la aplicación se explican en esta sección para poder comprender de un modo más preciso el diseño arquitectónico de la aplicación. Para empezar se mostrará el diagrama de clases correspondiente y posteriormente se realizará una descripción detallada de cada una de ellas.

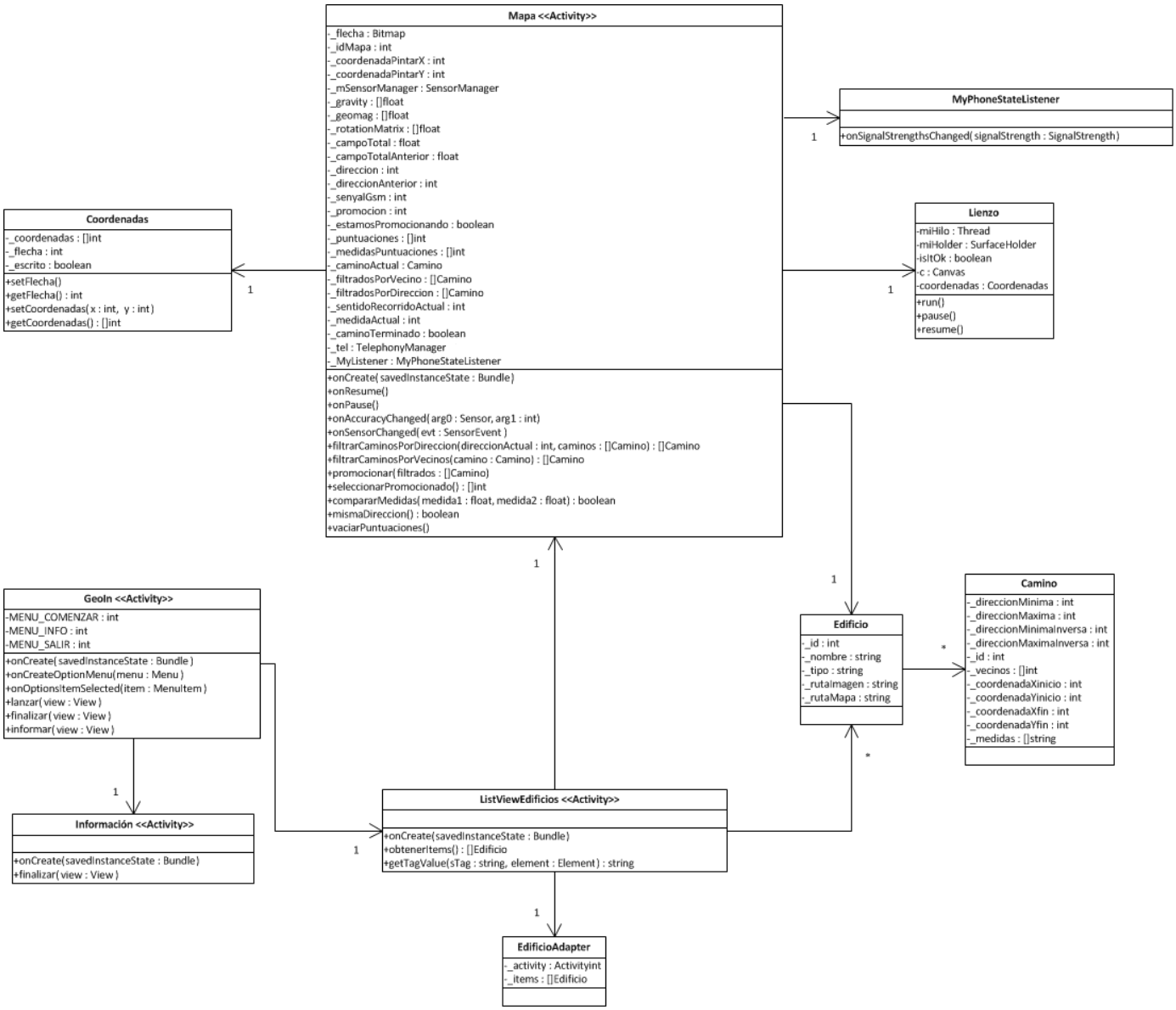

Ilustración 9: Diagrama de clases.

Varias de las clases se corresponden a su vez con las interfaces de la aplicación, hecho que se explicará de forma detallada en el punto 4.2.2.

La plantilla utilizada para la descripción de todas las clases se muestra en la Tabla 41: Formato de Clase.

Identificador	
Nombre	
Tipo	
Función	
Dependencias	
Atributos	
Métodos	

Tabla 41: Formato de Clase.

Cada campo de la plantilla representa:

- **Identificador:** caracteres que identifican unívocamente la clase. Están formados por las letras CL seguida del número de clase del que se trate.
- **Nombre:** representa el nombre de la clase y como está representado en el diseño.
- **Tipo:** indica qué es: clase o subclase.
- **Función:** indica la función de la clase, es decir, lo que hace.
- **Dependencias:** indica las relaciones con otras clases.
- **Atributos:** indica los atributos que tiene la clase.
- **Métodos**: indica los métodos de la clase mediante el formato "Nombre (parámetros): descripción".

Identificador	CL-01
Nombre	Camino
Tipo	Clase
Función	Define la estructura de un camino
Dependencias	Ninguna
Atributos	_id, medidas, _direccionMinima, _direccionMaxima, _direccionMinimaInversa, _direccionMaximaInversa, _vecinos, _coordenadaXinicio, _coordenadaYinicio, _coordenadaXfin, coordenadaYfin
Métodos	Métodos get

Tabla 42: CL-01.

Identificador	CL-02
Nombre	Edificio
Tipo	Clase
Función	Define la estructura de un edificio
Dependencias	Camino
Atributos	_id, _nombre, _tipo, _rutaImagen, _rutaMapa
Métodos	Métodos get

Tabla 43: CL-02.

Identificador	CL-03
Nombre	Coordenadas
Tipo	Clase
Función	Define una estructura para poder cambiar las coordenadas de forma síncrona
Dependencias	Ninguna
Atributos	_coordenadas, flecha, _escrito
Métodos	- **setFlecha:** cambia el valor de la variable _flecha de forma síncrona - **getFlecha:** obtiene el valor de la variable _flecha de forma síncrona - **setCoordenadas:** cambia el valor de la variable _coordenadas de forma síncrona - **getCoordenadas:** obtiene el valor de la variable _ coordenadas de forma síncrona

Tabla 44: CL-03.

Identificador	CL-04
Nombre	Información
Tipo	Clase-Interfaz gráfica
Función	Muestra la información proporcionada por la aplicación
Dependencias	Ninguna
Atributos	Ninguno
Métodos	- **onCreate:** método que se ejecuta al crear la pantalla - **finalizar:** finaliza la pantalla

Tabla 45: CL-04.

Identificador	CL-05
Nombre	EdificioAdapter
Tipo	Clase
Función	Proporcionar un Adapter propio para la lista mostrada en la clase ListViewEdificio
Dependencias	Ninguna
Atributos	_activity, _items
Métodos	Métodos get

Tabla 46: CL-05.

Identificador	CL-06
Nombre	ListViewEdificio
Tipo	Clase-Interfaz gráfica
Función	Muestra la lista de edificios disponibles en la aplicación
Dependencias	Edificio, EdificioAdapter
Atributos	Ninguno
Métodos	- **onCreate:** método que se ejecuta al crear la pantalla - **obtenerItems:** obtiene la lista de edificios de la base de datos - **getTagValue:** obtiene el valor de una etiqueta para un archivo *XML*

Tabla 47: CL-06.

Identificador	CL-07
Nombre	GeoIn
Tipo	Clase-Interfaz gráfica
Función	Constituye la pantalla principal
Dependencias	Información, ListViewEdificio
Atributos	MENU_COMENZAR, MENU_INFO, MENU_SALIR
Métodos	- **onCreate:** método que se ejecuta al crear la pantalla - **onCreateOptionMenu:** asigna operaciones al menú de opciones - **onOptionsItemSelected:** se ejecuta al pulsar una de las opciones del menú - **lanzar:** abre la pantalla ListViewEdificio - **finalizar:** finaliza la aplicación - **informar:** abre la pantalla Información

Tabla 48: CL-07.

Identificador	CL-08
Nombre	MyPhoneStateListener
Tipo	Clase
Función	Controla la señal *GSM* del teléfono
Dependencias	Ninguna
Atributos	Ninguno
Métodos	- **onSignalStrenghsChanged:** se ejecuta cada vez que cambia el valor de la intensidad de señal *GSM*

Tabla 49: CL-08.

Identificador	CL-09
Nombre	Lienzo
Tipo	Clase
Función	Muestra en pantalla el mapa y la posición del individuo
Dependencias	Ninguna
Atributos	miHilo, miHolder, isItOk, c, coordenadas
Métodos	- **run:** ejecución del hilo para que pinta en pantalla - **pause:** para la ejecución del hilo - **resume:** continúa la ejecución del hilo

Tabla 50: CL-09.

Identificador	CL-10
Nombre	Mapa
Tipo	Clase-Interfaz gráfica
Función	Realiza la lógica para conocer la posición del individuo en el mapa
Dependencias	Coordenadas, Edificio, Lienzo, MyPhoneStateListener
Atributos	_flecha, _idMapa, _gravity, _tel, _coordenadaPintarX, _coordenadaPintarY, _mSensorManager, _geomag, _rotationMatrix, _campoTotal, _campoTotalAnterior, _direccion, _direccionAnterior, _senyalGsm, _promocion, _estamosPromocionando, _medidaActual, _puntuaciones, _medidasPuntuaciones, _myListener, _caminoActual, _filtradosPorDireccion, _sentidoRecorridoActual, filtradosPorVecino, _caminoTerminado
Métodos	- **onCreate:** método que se ejecuta al crear la pantalla - **onResume:** método que se ejecuta cuando la actividad vuelve a estar activa - **onPause:** método que se ejecuta cuando la actividad pausa su ejecución. - **onAccuracyChanged:** método que se ejecuta cuando la precisión de algún sensor cambia - **onSensorChanged:** método que se ejecuta cuando cambia el valor recogido por alguno de los sensores - **filtrarCaminosPorDireccion:** método que filtra de una lista los caminos con la dirección especificada - **filtrarCaminosPorVecinos:** método que filtra de una lista los caminos que son vecinos de uno indicado - **promocionar:** método que suma un punto al camino que más similitud tenga con las medidas obtenidas de los sensores - **seleccionarPromocionado:** método que selecciona de una lista el camino con mayor puntuación - **compararMedidas:** método que compara dos medidas y devuelve true si la diferencia es menor o igual que una cantidad especificada -**mismaDireccion:** método que indica si la dirección medida es igual a la de la anterior medición -**vaciarPuntuaciones:** método que pone a cero las puntuaciones de los caminos promocionados

Tabla 51: CL-10.

4.2.2 Base de datos

Al ser un prototipo y manejar poca cantidad de datos, se ha optado por implementar el diseño de la base de datos de tipo relacional para un futuro, como una mejora. De esta manera, se ha preferido para este momento del desarrollo, crear una pequeña base de datos mediante un archivo *XML* que contendrá los datos necesarios en forma de árbol y al cual se accederá mediante las correspondientes clases de *Java*.

La estructura del archivo *XML* es:

```
<edificios>
      <edificio>
            <id></id>
            <nombre></nombre>
            <tipo></tipo>
            <imagen></imagen>
            <mapa></mapa>
            <caminos>
                  <camino>
                        <id></id>
                        <vecinos></vecinos>
                        <coordenadasInicio></coordenadasInicio>
                        <coordenadasFin></coordenadasFin>
                        <medidas>
                              <medida></medida>
                              .....
                        </medidas>
                  </camino>
                  .....
            </caminos>
      </edificio>
      .....
</edificios>
```

Donde:

- Las etiquetas *id* corresponden a números naturales y secuenciales que comienzan por 0.
- Las etiquetas *imagen* y *mapa* son rutas relativas de las imágenes.
- La etiqueta *vecinos* contiene los vecinos del camino, y su contenido son las *id* de los vecinos, separadas por un guión.
- Las etiquetas *coordenadasInicio* y *coordenadasFin* contienen números enteros con las coordenadas relativas al mapa donde comienza y termina el camino.
- La etiqueta *medida* contiene los valores del campo magnético, la dirección y la intensidad de señal *GSM* separados por un guión entre ellas. Son números decimales.

4.2.3 Proceso de recopilación de datos

Los conjuntos numéricos almacenados en la base de datos corresponden a los valores del campo magnético, señal *GSM y* dirección tomados para cada uno de los caminos que se definen para el plano. Para realizar este proceso se ha implementado una aplicación de apoyo que permita seleccionar un mapa concreto, señalar el camino que se van a recorrer para almacenar los valores anteriormente referidos y definir el comienzo y el fin del proceso. Es decir, que habrá que relizar tantos recorridos y mediciones como caminos haya.

Un camino es considerado una distancia en línea recta, de modo que cada uno de sus extremos concuerda con el extremo de otro camino.

Para explicar esto de forma más clara, vamos a mostrar los posibles caminos de la tercera planta del edificio Sabatini:

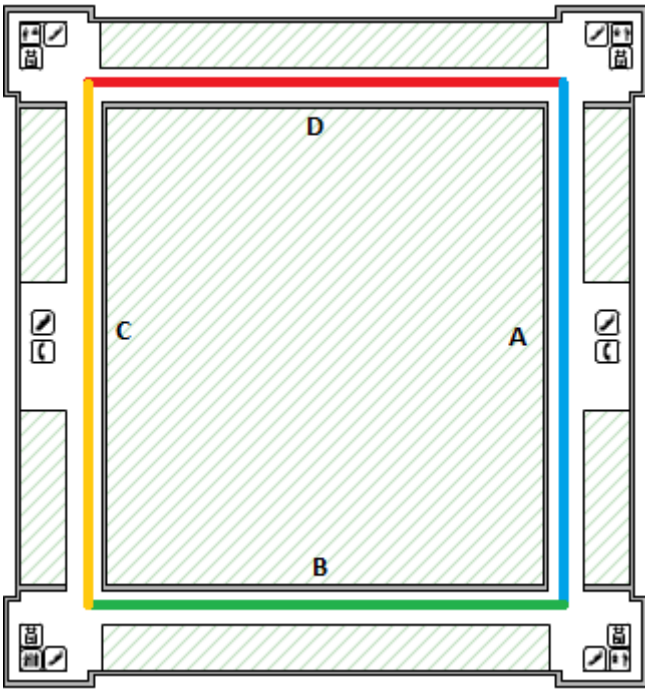

Ilustración 10: caminos de la planta 3 del edificio Sabatini.

Los cuatro caminos están recogidos por las líenas de colores. Aun pareciendo una obviedad, ha de quedar claro que no es válido tomar una muestra de los valores dentro de una habitación, a no ser que sea una habitación con un gran recorrido. Esto es debido a que la posición se determina con el conjunto de los valores de dirección, campo magnético y señal *GSM* y en caso de estar en un lugar cerrado y de poca extensión, no hay tiempo material para realizar las comprobaciones necesarias. De hecho, una de los puntos más importantes es la dirección, que hay que mantener constante, pues su variación fuera del rango establecido para un camino, hace que nuestra posición no

avance. En otras palabras, para cada camino existen dos direcciones de recorrido únicamente: al derecho y al revés.

Por supuesto no es un número fijo, si no un intervalo de ±x (siendo x un valor arbitrario, en el caso práctico de 15º) con respecto a la dirección media extraída de las medidas. Esto quiere decir que nuestro camino tiene como direcciones válidas todas las comprendidas entre:

dirección media – X ≤ dirección Actual ≤ dirección media + X

Se ha escogido esto así pues es imposible llevar el terminal siempre recto y apuntando en una dirección estable.

Fuera del rango, el sistema no reconoce una dirección válida y no "avanza". Por ello, en espacios reducidos al haber más variación de movimiento, más allá del simple desplazamiento en línea recta, no es aconsejable realizar mediciones.

La aplicación diseñada para este proceso es sencilla. Para empezar escogemos un mapa al que realizar las medidas precisas, por ejemplo el mismo edificio Sabatini. La aplicación nos pedirá poner un nombre al edificio, para guardar los datos en un archivo con ese nombre:

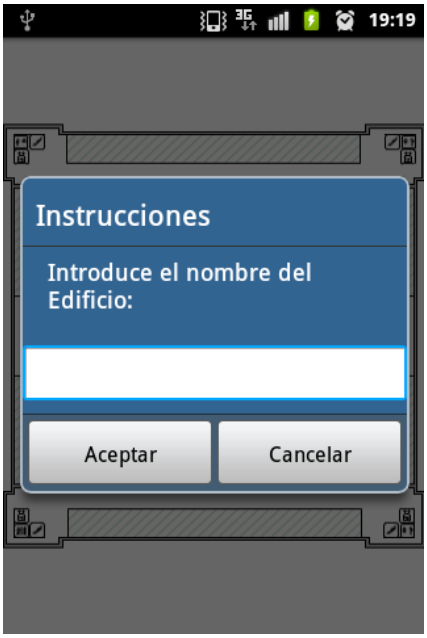

Ilustración 11: Recopilación de datos.

Una vez mostrado el mapa, al tocar la pantalla en un punto, se pintará un círculo pequeño que indica el punto de origen del camino.

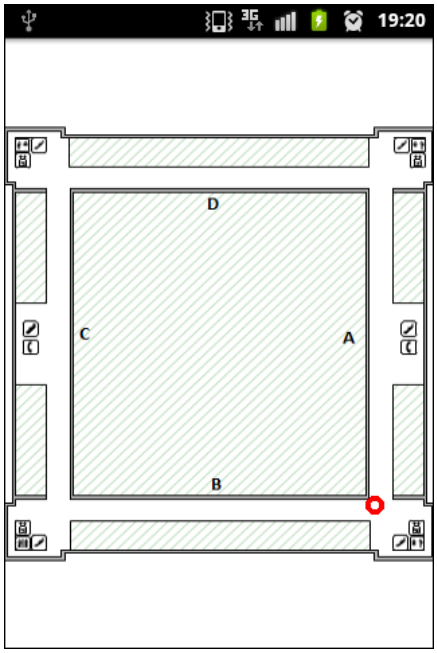

Ilustración 12: Recopilación de datos (II).

A continuación hay que pulsar en la pantalla nuevamente para definir el camino que a recorrer. Cuando se realice este paso, se pintará otro círculo idéntico al primero en la posición tocada y se unirán los puntos mediante una línea:

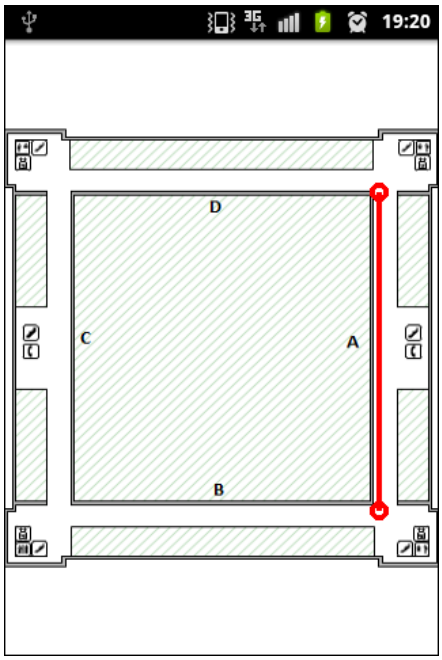

Ilustración 13: Recopilación de datos (III).

El último proceso que nos queda por hacer será recorrer el camino trazado por la línea, en el sentido que va desde el primer punto hasta el segundo. En el momento de recorrer el camino, se pulsa el botón de menú y a continuación el botón *Comenzar:*

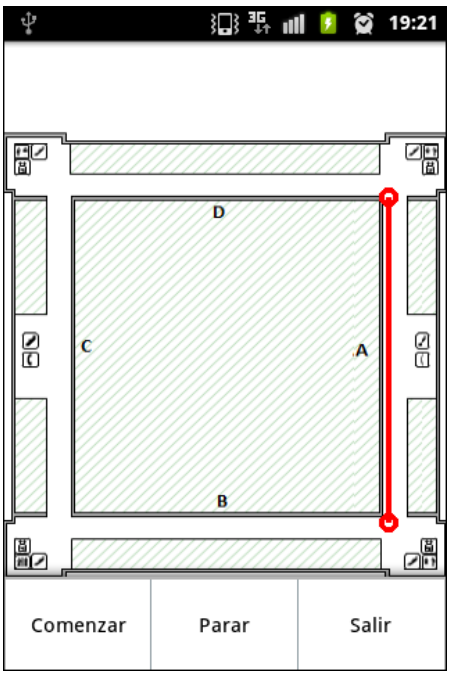

Ilustración 14: Recopilación de datos (IV).

Después de activar el botón, se recorre el camino y al llegar al final del mismo, se vuelve a pulsar el botón de menú para posteriormente pulsar el botón *Parar.*

Todo este proceso hay que repetirlo tantas veces como caminos se han definido medir. Cuando se termina la etapa de medición, en la memoria externa del teléfono se ha generado un archivo con el nombre escogido previamente y los valores recogidos para cada uno de los caminos. El archivo generado es idéntico al mostrado en el apartado 4.2.2, pero sin la etiqueta *edificios*.

Por último, este archivo se añade a la aplicación original *GeoIn* para poder usar las medidas en el proceso de localización.

4.2.4 Especificación de interfaces de usuario

Esta sección explica en profundidad las decisiones referentes al diseño de las interfaces de usuario y que serán el nexo entre aplicación y usuarios. Un diseño sobrio pero a la vez atractivo es un punto decisivo a la hora de diseñar interfaces. Por otro lado, es necesario que el diseño sea sencillo y no conduzca a errores de navegación.

Para que todo ello sea posible, hay que definir una serie de normas que se llevarán a cabo durante todo el proceso de diseño y que pueden aplicarse en una segunda fase de depuración de errores, si se considera que es necesaria una revisión.

Las normas establecidas son:

- El diseño de todas las interfaces ha de ser uniforme y debe tener el menor número de elementos posibles con el que cumpla su cometido.

- Cada una de las pantallas debe poseer una navegación intuitiva para todos los tipos de usuario.

- Se debe indicar de forma específica la acción que resultará al realizar ciertas interactuaciones.

- Es necesario mostrar un mensaje en caso de que la aplicación tarde más de lo normal en cargar ciertas operaciones.

Las interfaces que se ha decidido diseñar son cuatro:

- **Pantalla principal:** esta interfaz es la primera que se muestra al iniciar la aplicación. Desde ella es posible acceder a la lista de ubicaciones disponibles, a la información y salir de la aplicación. Cada una de las acciones se realiza mediante un botón asignado.

- **Información:** esta interfaz contiene información relacionada con la aplicación y contiene un correo electrónico de contacto en caso de que un usuario necesitara comunicar algo al creador de la aplicación. Desde esta pantalla únicamente se puede regresar a la pantalla principal mediante el botón *Salir*.

- **Pantalla con ubicaciones:** esta interfaz contiene una lista con todas las ubicaciones disponibles en la aplicación. Cada uno de los elementos de la lista constituye un ítem que puede ser pinchado. Al realizar esta acción, se accede al mapa correspondiente de la ubicación seleccionada. Por supuesto, también se puede regresar a la pantalla principal.

- **Pantalla con mapa:** esta interfaz contiene el mapa de la ubicación seleccionada. Este mapa representa un plano real a escala. En él se verá la localización en la que nos encontramos en cada momento.

La navegación entre las diferentes interfaces queda de la siguiente forma:

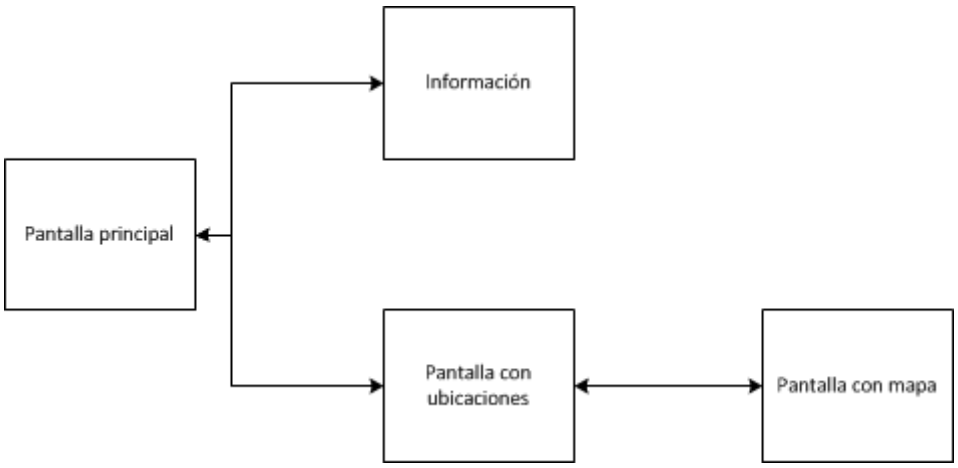

Ilustración 15: Diagrama de navegación.

5. Implementación

5.1 Implementación de interfaces Android

Las interfaces que se van a implementar son las mencionadas en el apartado 4.2.2 y que coinciden con las clases que constituyen una *Activity* mostradas en el diagrama de clases del apartado 4.2.1.

La primera de las interfaces de usuario es la pantalla principal. Se ha decidido añadir un logo con la forma de un radar como adorno y a la vez como un mensaje claro de la funcionalidad de la aplicación. La interfaz contiene tres botones diferentes, cuyo comportamiento queda bien explicado en el texto que poseen. Según el botón que pulsemos se puede acceder a la lista de ubicaciones disponibles, acceder a la información o salir de la aplicación.

Todos los botones poseen el mismo tamaño y están distribuidos de forma simétrica para destacar que cada uno de ellos tiene la misma relevancia que el resto y no hay prioridad por ninguno.

Ilustración 16: Pantalla principal.

La siguiente interfaz contiene la información que se quiera transmitir al usuario de la aplicación. En este caso aparece vacía, aunque podría contener información sobre los derechos de autor, un email de contacto o una explicación básica del funcionamiento de la aplicación. Posee un único botón que permite regresar a la pantalla de inicio.

Ilustración 17: Pantalla de información.

La tercera interfaz muestra una lista con todos los edificios o ubicaciones que se encuentran disponibles en la aplicación. Como se puede observar, cada elemento contiene el nombre del edificio con el tipo (ya sea universidad, centro comercial, etc.) y una imagen propia y contextual. Cada uno de los elementos de la lista ejecuta un evento cuando es pulsado. Como es lógico, los eventos son distintos entre sí y nos muestran la correspondiente pantalla al ejecutarse.

Ilustración 18: Pantalla de lista de edificios.

La última de las interfaces es la contiene el mapa del edificio seleccionado. Según el elemento de la lista que pulsemos aparecerá un mapa distinto, por lo que es imposible reflejar todos los posibles. Sin embargo, los factores comunes que poseen todos ellos es que están diseñados con la herramienta *Microsoft Visio* y muestran la posición y la dirección de movimiento del usuario mediante una flecha verde. Todo esto se muestra en pantalla completa, por lo que el título de la aplicación y la barra de notificaciones no aparecerán.

El mapa propuesto corresponde a la planta baja del edificio Torres Quevedo de la Universidad Carlos III de Leganés, en Madrid.

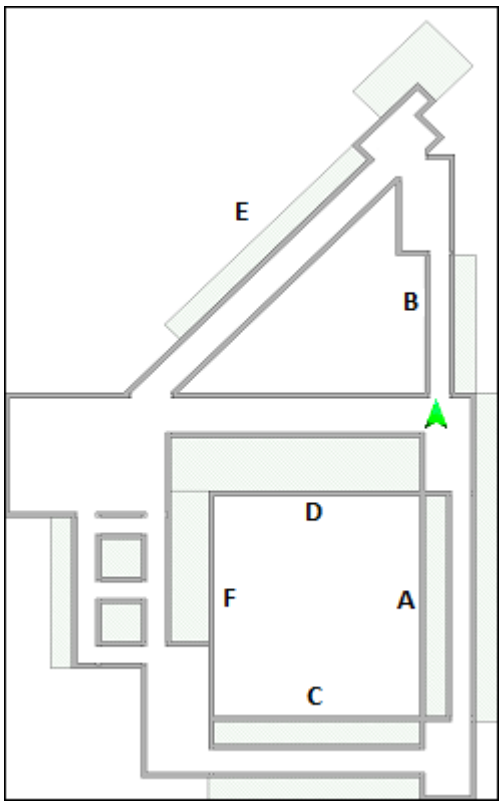

Ilustración 19: Pantalla de mapa.

Es evidente que las interfaces no componen toda la aplicación, pues quedaría por implementar el resto de clases del proyecto y todos los cálculos y acciones necesarias en cada una de las clases de interfaz para que funcione correctamente la aplicación.

5.2 Implementación de base de datos

La implementación de la base de datos se realiza mediante un programa auxiliar que recoge los datos y los almacena de forma ordenada en un archivo *XML*. Para la recogida de datos, es necesario recorrer cada uno de los caminos a pie con el móvil y seleccionar en el mapa el recorrido que se va a realizar.

6.Plan de pruebas

6.1 Pruebas de aceptación

Una vez detallado todo el diseño del sistema, es necesario comprobar que se cumplen todos los requisitos definidos por el usuario. Para ello se ha establecido un plan de pruebas que verifique el correcto funcionamiento una vez terminado el desarrollo.

La plantilla empleada para recoger las pruebas se muestra en la Tabla 52: Formato de prueba. :

Identificador	
Descripción	
CU relacionado	

Tabla 52: Formato de prueba.

Donde cada campo representa:

- **Identificador:** caracteres que identifican unívocamente la prueba. Están formados por las letras PR seguida del número de prueba del que se trate.
- **Descripción:** pequeña definición de la prueba.
- **CU relacionado:** caso de uso con el que está relacionado.

Identificador	PR-01
Descripción	Se comprueba que el usuario puede acceder a la aplicación e iniciarla
CU relacionado	CU-01

Tabla 53: PR-01.

Identificador	PR-02
Descripción	Se comprueba que el usuario puede visualizar la lista de ubicaciones al pulsar el botón *Comenzar*
CU relacionado	CU-02

Tabla 54: PR-02.

Identificador	PR-03
Descripción	Se comprueba que el usuario puede pulsar en cualquiera de las ubicaciones de la lista y acceder a la pantalla con el mapa
CU relacionado	CU-02

Tabla 55: PR-03.

Identificador	PR-04
Descripción	Se comprueba que el usuario puede visualizar su posición y dirección de movimiento dentro del mapa
CU relacionado	CU-02

Tabla 56: PR-04.

Identificador	PR-05
Descripción	Se comprueba que el usuario puede acceder a la información mediante el botón correspondiente situado en la pantalla principal
CU relacionado	CU-03

Tabla 57: PR-05.

Identificador	PR-06
Descripción	Se comprueba que el usuario puede salir de la aplicación mediante el botón correspondiente situado en la pantalla principal
CU relacionado	CU-04

Tabla 58: PR-06.

6.2 Matriz de trazabilidad CU-PR

La matriz de trazabilidad CU-PR muestra la relación existente entre los casos de uso definidos en el apartado 3.4 y las pruebas recogidas en el apartado 6.1.

CU/ PR	CU-01	CU-02	CU-03	CU-04
PR-01	X			
PR-02		X		
PR-03		X		
PR-04		X		
PR-05			X	
PR-06				X

Tabla 59: Matriz de trazabilidad CU-PR.

7.Planificación

La planificación del proyecto muestra cada una de las tareas que se han llevado a cabo, así como cada una de las subtareas que las componen. Mediante el diagrama de Gantt podemos observar las fechas exactas de comienzo y finalización de las etapas y la dependencia entre las mismas. El proyecto ha durado un total de cuatro meses, viéndose alargado en la etapa de implementación debido a la complejidad del sistema.

El listado de tareas se muestra en la Ilustración 20: Tareas de la planificación.

	Modo de	Nombre de tarea	Duración	Comienzo	Fin
1		**Geolocalización en interiores**	**121 días**	**jue 20/09/12**	**jue 24/01/13**
2		Propuesta del proyecto	5 días	jue 20/09/12	mar 25/09/12
3		**Análisis del sistema**	**30 días**	**jue 20/09/12**	**lun 22/10/12**
4		Descripción del problema	5 días	mié 26/09/12	lun 01/10/12
5		Requisitos de usuario	7 días	mar 02/10/12	mar 09/10/12
6		Casos de uso	2 días	mié 10/10/12	jue 11/10/12
7		Requisitos de software	8 días	vie 12/10/12	vie 19/10/12
8		**Diseño del sistema**	**17 días**	**sáb 20/10/12**	**mar 06/11/12**
9		Arquitectura del sistema	6 días	sáb 20/10/12	jue 25/10/12
10		Diseño detallado del sistema	10 días	vie 26/10/12	mar 06/11/12
11		**Implementación**	**60 días**	**mié 07/11/12**	**mar 08/01/13**
12		Implementación de interfaces	16 días	mié 07/11/12	jue 22/11/12
13		Implementación de la lógica	32 días	vie 23/11/12	mié 26/12/12
14		Implementación de base de datos	2 días	jue 27/12/12	vie 28/12/12
15		Plan de pruebas	3 días	mié 09/01/13	vie 11/01/13
16		Documentación	12 días	vie 11/01/13	jue 24/01/13

Ilustración 20: Tareas de la planificación.

El diagrama de Gantt resultante de la planificación de las tareas definidas se muestra en la Ilustración 21: Diagrama de Gantt.

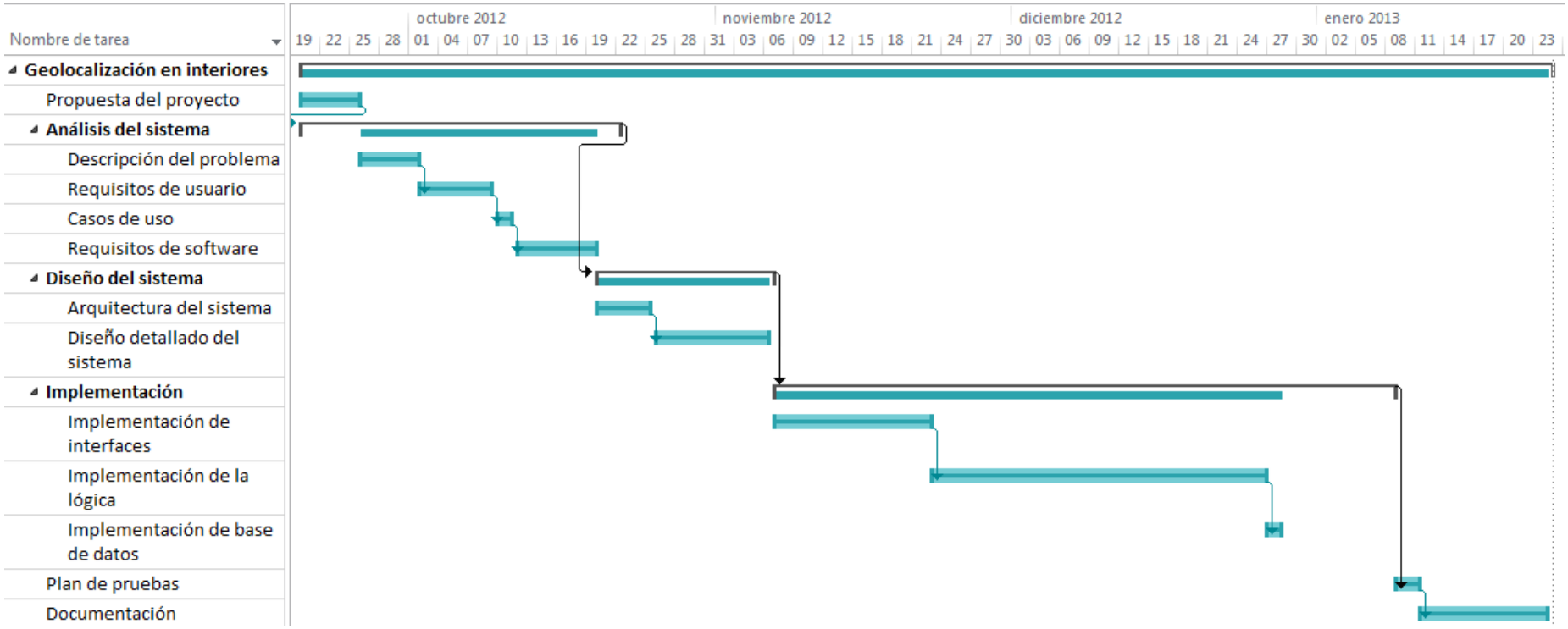

Ilustración 21: Diagrama de Gantt.

8.Presupuesto final

El presupuesto está compuesto por todos y cada uno de los factores cuantificables del proyecto. Es por ello que se muestra de forma desglosada y detallada.

8.1 Coste por persona

Para obtener un valor real del coste que ha generado el personal del proyecto, es necesario conocer el tiempo que se ha trabajado en horas y el salario de cada uno de los empleados.

En este caso hay un único trabajador en el proyecto de fin de carrera, cuyo salario estimado es de 14 €/hora. El coste personal queda así:

Puesto	Cantidad	Salario (€/h)	Días trabajados	Horas por día	Total (€)
Ingeniero junior	1	14	121	6	10.164
Total					10.164

Tabla 60: Coste por persona.

8.2 Coste de hardware

El hardware que ha sido utilizado para la realización del proyecto está limitado al dispositivo móvil y los ordenadores necesarios para la implementación de la memoria y la aplicación.

Hardware	Cantidad	Precio (€)	Total (€)
HP ENVY h8-403es Intel Core i7 3770	1	1399	1.399
Samsung Galaxy ACE	1	149	149
Total			1.548

Tabla 61: Coste de hardware.

8.3 Coste de software

El software incluido comprende desde las herramientas destinadas al desarrollo de la aplicación, hasta las necesarias para realizar el memorando y cada una de sus secciones.

Software	Cantidad	Precio (€)	Total (€)
Eclipse	1	0	0
SDK y herramientas Android	1	0	0
Microsoft Office 2010	1	249	249
Microsoft Visio Premium 2010	1	509	509
Microsoft Project 2010	1	1.067	1.067
Total			1.825

Tabla 62: Coste de software.

8.4 Coste total

Los costes ofrecidos en la Tabla 60, Tabla 61 y Tabla 62 cuentan con el I.V.A incluido, por lo que el coste total no necesita de una operación adicional más allá de la suma de los totales parciales.

Concepto	Coste (€)
Coste por persona	10.164
Coste de hardware	1.548
Coste de software	1.825
Total	13.537

Tabla 63: Coste total.

9. Conclusiones

9.1 Reflexión final

En la actualidad, la implementación de aplicaciones para dispositivos móviles constituye un frente en auge, en el que todo usuario puede aportar su grano de arena. En cuanto a esto, aprovechando el proyecto de fin de carrera, he querido aportar mi pequeña dosis de investigación y mis cualidades para el desarrollo de una aplicación funcional.

Durante este proceso, he contado con numerosas dificultades y problemas que surgían cada vez que daba un nuevo paso. Sin embargo, esto no ha impedido que terminara con éxito mi propósito. Algunos de los problemas más importantes que he superado son:

- El desconocimiento por completo del desarrollo de una aplicación *Android*. Al no tener ninguna experiencia en este campo, se ha dedicado un extenso marco temporal en el aprendizaje de esta habilidad.

- El diseño de las interfaces de usuario mediante la herramienta *Eclipse* ha sido bastante complicado. Debido a que usa archivos *XML* para ordenar los elementos y que se usa una distribución de los objetos mediante capas y posiciones relativas, ha sido bastante duro conseguir la apariencia deseada para la aplicación.

- Conseguir una localización sin el uso de datos móviles ni de *GPS* ha resultado el mayor de los retos. Considero esta tarea la más ardua, pues es la que más me ha hecho plantear la viabilidad de la aplicación y la que me ha llevado a la más completa desesperación.

- El desarrollo de hilos para comprobar el valor de los sensores con los datos almacenados, realizar cálculos exhaustivos y pintar en tiempo real la posición es un reto difícilmente superable. No debido al manejo de hilos, sino más bien a la prioridad de unas acciones sobre otras y a cómo los sensores paralizan el comportamiento del resto de la aplicación.

A pesar de todo ello, he de destacar que la realización de este proyecto me ha permitido la oportunidad de ponerme a prueba. Gracias a él he conseguido aplicar numerosas lecciones aprendidas en varias asignaturas, que pensé que no podría en práctica. Y por si fuera poco, he tenido la suerte de aprender el desarrollo de aplicaciones en *Android*, hecho de enorme valía si tenemos en cuenta la importancia y el esplendor que está teniendo en estos últimos años en el sector de la informática y en la sociedad.

9.2 Líneas futuras

Con el objetivo de retomar en un futuro la aplicación desarrollada en este proyecto, este apartado recoge una serie de posibles evoluciones a realizar. Es necesario insistir en que las posibles mejoras aquí descritas no deben afectar al actual funcionamiento y diseño de la aplicación.

La principal y más compleja de las mejoras es hacer accesible por completo la aplicación. Esto consiste en dar un mayor protagonismo al usuario para que pueda diseñar sus propios mapas, realizar las medidas del campo magnético, dirección y resto de magnitudes, y compartirlo con el resto de usuarios. Con ello se consigue que la aplicación evolucione de una forma más rápida, obteniendo gran cantidad de mapas para la localización en interiores. Para que todo se pudiera llevar a cabo, habría que crear un servicio homogéneo y único para que cada usuario diseñara los mapas desde la misma herramienta.

Para apoyar la técnica descrita en esta memoria, es posible en un futuro distribuir ciertas antenas por las localizaciones. Mediante el bluetooth se realizaría una triangulación para conseguir la posición del individuo aunque éste se mantenga inmóvil.

Otro modo de evolucionar la actual aplicación consiste en el empleo de más sensores. Con el paso del tiempo, los dispositivos cuentan con más sensores que miden una gran cantidad de datos, ya sea el campo magnético, la humedad, etc. Gracias a ellos sería posible discernir de un modo más preciso la posición de un usuario.

Llegado el momento, puede plantearse como ayuda suplementaria el uso de paquetes de datos o redes *WI-FI*. Sería otro camino para una mayor precisión de la aplicación, y más teniendo en cuenta que con el paso del tiempo estas tecnologías estarán disponibles incluso en sótanos y lugares con mala cobertura.

Todas estas son posibles mejoras y avances dentro de la aplicación, siempre y cuando no se modifique de forma drástica el funcionamiento actual.

Glosario

AAC	Advanced Audio Coding
API	Application Programming Interface
ARM	Acorn RISC Machine
CDMA	Code Division Multiple Access
ESA	European Space Agency
EVDO	Evolution-Data Optimized
GNU	GNU's Not Unix, Linux OS
GPS	Global System for Mobile
HTML	HyperText Markup Language
IEEE	Institute of Electrical and Electronics Engineers
iOS	iPhone Operating System
IVA	Impuesto sobre el Valor Añadido
JIT	Just In Time
MKV	Matroska Video
MMS	Multimedia Messaging System
MVC	Model View Controller
NFC	Near Field Communication
RIM	Research in Motion
SD	Secure Digital
SDK	Software Development Kit
SGL	Simple Graphics Library
SIP	Session Initiation Protocol
SMS	Short Message Service
SQLite	Structured Query Language ite
SSL	Secure Sockets Layer
USB	Universal Serial Bus
VoIP	Voice over IP
VPN	Virtual Private Network
WI-FI	Wireless Fidelity
WVGA	Wide Video Graphics Array
WXGA	Wide Extended Graphics Array
XML	Extensible Markup Language
YAFFS	Yet Another Flash File System

Referencias

[1] Página oficial de Android. Disponible [Internet]: http://www.android.com/

[2] Página oficial para desarrolladores de Android. Disponible [Internet]: < http://developer.android.com/index.html>

[3] Arquitectura de Android. Disponible [Internet]: < http://www.configurarequipos.com/doc1107.html>

[4] Página oficial de configuración de entorno Android en Eclipse. Disponible [Internet]: < http://developer.android.com/sdk/installing/installing-adt.html>

[5] Tutorial de Java DOM. Disponible [Internet]: < http://apuntes.delibertad.com/java/leer-archivo-xml-desde-java/>

[6] Tutorial de calibración de brújula digital. Disponible [Internet]: < http://forum.xda-developers.com/showthread.php?t=1142429>

[7] Tutorial de SurfaceView para Android. Disponible [Internet]: < http://stackoverflow.com/questions/10956583/android-draw-using-surfaceview-and-thread>

[8] MVC, Model View Controller. Disponible [Internet]: http://www.roseindia.net/tutorial/java/jdbc/javamvcdesignpattern.html

[9] ESA Lite. Disponible [Internet]: http://www.ie.inf.uc3m.es/grupo/docencia/reglada/Is1y2/ESA/ESA-Standards.htm

[10] Indoor Atlas. Disponible [Internet]: http://www.indooratlas.com/

Anexo A: Manual de Usuario

Instalación

La extensión de la aplicación, al igual que todas las desarrolladas en *Android*, es *.apk*. Para poder instalarla es necesario seguir dos sencillos pasos:

- En el dispositivo móvil hay que habilitar la instalación de aplicaciones que provienen de orígenes no conocidos o sin certificar. Para ello vamos al menú Ajustes →Aplicaciones y marcamos la opción *Fuentes desconocidas.*

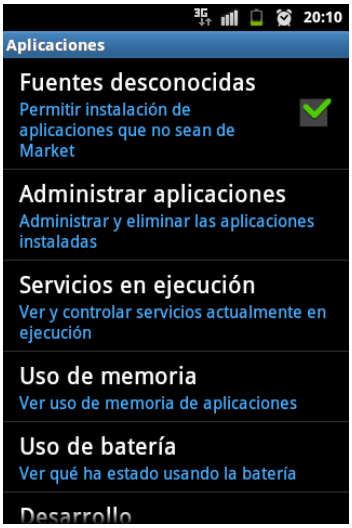

Ilustración 22: Instalación de .apk.

- A continuación, se copia el archivo de la aplicación en la memoria del teléfono (puede ser en una tarjeta *sd*). Una vez hecho esto, se busca el archivo mediante el explorador de archivos y se ejecuta para instalar la aplicación.

Una vez realizados estos dos pasos, la aplicación estará lista para ejecutar.

Guía de uso

Al ejecutar la aplicación, aparece la pantalla principal. Existen tres botones en ella y cada uno de ellos tiene su propósito: iniciar la aplicación, acceder a la información y salir de la aplicación.

Ilustración 23: Guía pantalla principal.

Estas tres acciones pueden realizarse también desde el botón de menú existente en los dispositivos.

Ilustración 24: Guía pantalla principal 2.

Si presionamos el botón *Info*, se abre una nueva pantalla con la información que el desarrollador quiera ofrecer a los usuarios. Dentro de esta pantalla existe un botón con el texto *Volver*, que si pulsamos nos devuelve a la página principal.

Ilustración 25: Guía información.

Una vez en la pantalla principal, si pulsamos *Comenzar*, se nos muestran las localizaciones que tiene disponible un mapa para comenzar nuestro posicionamiento.

Ilustración 26: Guía lista de edificios.

Cada uno de los elementos de la lista indica el nombre de la localización, el tipo de localización y muestra una foto descriptiva. Se puede pulsar cualquier elemento para mostrar la pantalla final que nos indicará nuestra posición precisa en tiempo real siempre que el usuario se muestre en movimiento.

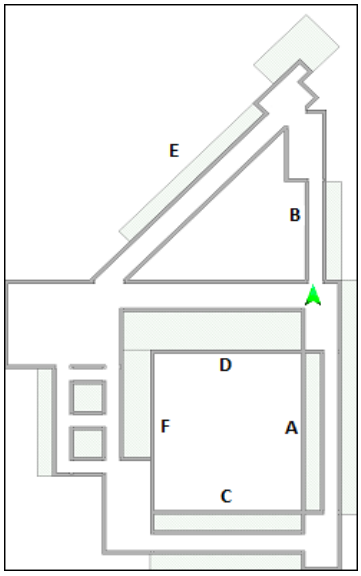

Ilustración 27: Guía mapa de posición.

Para salir de la pantalla del mapa podemos acceder al menú o pulsar la tecla de retorno del dispositivo.

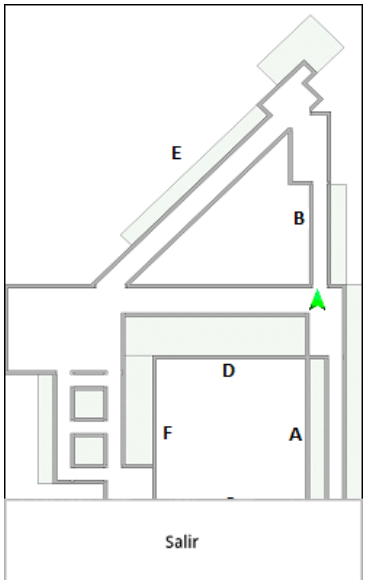

Ilustración 28: Guía mapa de posición 2.