Departamento de Informática

PROYECTO FIN DE CARRERA
INGENIERÍA INFORMÁTICA

Leganés, Septiembre de 2012

Diseño de una Aplicación de Gestión de Ubicaciones de Usuarios para Plataforma Android

Autor: Mario Bordas Martínez

Tutor 1: José Luis López Cuadrado

Tutor 2: Alejandro Rodríguez González

TÍTULO: DISEÑO DE UNA APLICACIÓN DE GESTIÓN DE UBICACIONES DE USUARIOS PARA PLATAFORMA ANDROID

AUTOR: MARIO BORDAS MARTÍNEZ

DIRECTOR: ALEJANDRO RODRÍGUEZ GONZÁLEZ

EL TRIBUNAL

PRESIDENTE: _____

VOCAL: _____

SECRETARIO: _____

Realizado el acto de defensa y lectura del Proyecto Fin de Carrera el día __ de _____ de 20__ en Leganés, en la Escuela Politécnica Superior de la Universidad Carlos III de Madrid, acuerda otorgarle la CALIFICACIÓN de

VOCAL

SECRETARIO PRESIDENTE

Agradecimientos

Dedicarles todo mi esfuerzo y el final de una etapa tan compleja como es el mundo de la universidad a toda mi familia, porque siempre han estado apoyándome en todas las metas que me he propuesto cumplir y han sabido enseñarme a crecer como persona. Y en especial: a mis padres que siempre me han animado a seguir adelante y a mis hermanos por su paciencia y por los buenos momentos que hemos vivido juntos.

Dedicarles también este esfuerzo a mis amigos que siempre han estado ahí en los buenos y malos momentos, tendiendo una mano cuando se necesitaba y una risa cuando se anhelaba: Torres, mis Albertos, José, Daniel, Alonso, Belén, entre muchos de ellos y sobre todo, Roberto.

Como no, dedicárselo a mis compañeros de la carrera, pues sin ellos esta etapa no hubiera sido posible: Aitor, Raúl, Roberto (el Canario), Durán, Fombellida, Laura y muchos más que habéis estado siempre a mi lado.

Dedicárselo a Coral ya que su apoyo, ayuda y comprensión en todos estos días ha significado mucho para mí, y es por ello por lo que he conseguido llegar al final.

En el ámbito académico a todos los profesores que me han visto crecer desde pequeño y me han visto madurar en destrezas, conocimiento y sabiduría.

Y como no, dedicárselo a mi tutor y mi cotutor, en especial a Alejandro que, aparte de dar la idea original, han velado en todo momento para que consiguiese tener un proyecto digno de un buen ingeniero superior.

A todos ellos:

Muchas gracias de corazón.

Resumen

El proyecto desarrollado es una aplicación móvil que obtiene la ubicación en la que el usuario se encuentra, toma una foto del lugar para el mejor reconocimiento del sitio y envía dicha información a los usuarios que hayamos seleccionado con el objetivo principal de establecer un punto de encuentro. De este modo los usuarios que reciben la información sabrán dónde se encuentra exactamente la persona que les ha enviado su ubicación, facilitando la localización. Todo ello gracias a un servidor que distribuye la información a los usuarios y a las tecnologías Android y Google Maps.

En los primeros apartados se hace un estudio de lo que se pretende realizar, abarcando por tanto todas estas tecnologías y el modo de operar con ellas. Se proporciona un resumen de la historia de las aplicaciones móviles y se hace una pequeña visión del estado del arte.

En los siguientes capítulos se muestra el análisis del sistema, los casos en los que el usuario puede interactuar con su modo de operación, la arquitectura propuesta para el desarrollo del proyecto y el diseño detallado del sistema.

En la parte final se muestra una batería de pruebas muy precisas que el sistema debe cumplir, siendo esta una de las partes más importantes del proyecto ya que las pruebas garantizan el correcto funcionamiento del sistema.

Abstract

The developed project is a mobile application which gets the location where the user is in this moment, takes a photo of the place for recognizing the place and sends all this information to the users that we have chosen with the primary objective of establishing a meeting point. So the users, who receive this information, know where is the person who send his location and meeting with this person will be easier. This is possible by a server that distributes the information to the users and the Android technologies, Google Maps.

In the first sections is done a study about the purpose which be carried out, so it includes all this technologies and the form to work with theirs. Also it is provided a resume of the history of the mobile applications and it is done a little vision of the art's state.

In the next sections it is seen the analysis of the system, the cases in which the users can interact with operation mode, the proposed architecture for project development and the detail design of the system.

At the end of the documentation it is shown a battery of tests very accurate that the system has to carry out, being one of the most important parts of the project because that tests the correct operation of the system.

Índice de Contenidos

Índice de Tablas

Índice de Ilustraciones

1 Introducción y Objetivos

En este apartado se detalla la motivación por la que nace este proyecto, explicando en qué consiste y las ventajas que ocasiona a la sociedad la utilización de la plataforma desarrollada. También se especifican los objetivos propuestos que se quieren adquirir una vez concluido el proyecto.

1.1 Introducción

En la actualidad la gente necesita reunirse con otras personas por varios motivos, ya sea por negocios, reuniones de amigos, salir a ver monumentos o museos, salir al cine, de fiesta, etc. Pero, ¿cuántas veces ha pasado que cuando hemos quedado con una persona o un grupo de personas no se encontraban y han tenido que llamar para poder encontrarse? Esta situación se repite una y otra vez en la sociedad de hoy día, ocasionando un coste asociado a dicha llamada.

En la actualidad se está en la era de las aplicaciones móviles, las cuales dan gran libertad a los usuarios que lo usan, ya que desde cualquier lugar se puede hacer lo que el usuario se proponga gracias a que los móviles suelen contar con las tarifas de datos que incluyen 3G o se pueden conectar casi en cualquier lugar a alguna red Wifi liberada. Esto facilita que cualquier aplicación que requiera de Internet pueda ejecutarse sin problemas. Se quiere emplear esta tecnología para dar solución al problema que se ha planteado.

No olvidar que la era Web está cada vez más desarrollada y aparecen conceptos nuevos para muchos informáticos y para muchas empresas. En particular hay un concepto que cada día se emplea más y más y es el concepto de Servicio Web. Para ello se instala un portal entre un servidor y sus clientes, donde estos pueden hacer las consultas pertinentes para que el servidor las responda. Este tipo de herramientas se usan mucho para comunicar los clientes móviles con los servidores que ofrecen sus servicios.

En el proyecto que se presenta a continuación trata de solucionar los problemas de la localización y el coste que nos acarrea el comunicar con otra persona para decirle dónde se encuentra el usuario exactamente. La solución que se ha desarrollado es un software que se instala en los móviles Smartphone de modo que se pueda enviar a la otra persona la localización exacta del usuario emisor, adjuntando una foto de referencia del lugar en el que se encuentre. De esta forma las personas que reciban la información por medio del 3G o Wifi podrán saber exactamente dónde está el lugar en el que se encuentra el usuario emisor y no le resulte difícil encontrarle.

1.2 Motivación del proyecto

Se podría afirmar de forma generalizada que tener que llamar y, por tanto, pagar para poder encontrarte con la persona que has quedado ya en un sitio, es bastante incómodo y es una situación que normalmente nos molesta. La principal motivación del proyecto es reducir estos casos y otorgando comodidad para solucionar dichos problemas, consiguiendo una mayor satisfacción a los usuarios que lo empleen.

Hoy día es muy común encontrarse con personas que dispongan de un teléfono Smartphone, no solo en los jóvenes, sino también en personas adultas, incluso en algunas personas de la tercera edad. Muchas de las empresas, incluso empresas con un potencial económico muy superior, ya se han sumado al desarrollo de aplicaciones enfocadas a estos dispositivos. Incluso muchos se atreven a desarrollar su propia aplicación y ofrecerla al público.

Está claro que las aplicaciones móviles hacen la vida más cómoda y sencilla a los usuarios, aunque es cierto que para estar totalmente satisfecho necesitas de varias de ellas que completen cada uno de los servicios que se te otorgan. Además gracias al desarrollo para las tecnologías móviles la vida se ha abaratado, no en el coste del día a día, sino que gracias a esas tecnologías se obtienen muchos servicios que antes no se tenían.

Por todo ello la motivación que impulsa a desarrollar este proyecto es sumarme a las miles de personas que desarrollan aplicaciones, para conseguir que la sociedad tenga más comodidades y consigamos sobrepasar los pequeños contratiempos que impulsan a llamar y gastar dinero.

1.3 Objetivos

Como ya se ha especificado anteriormente, el objetivo principal es desarrollar una aplicación que ayude a las personas o grupos de personas a encontrarse en un sitio sin necesidad de recurrir a llamadas telefónicas. Pero para ello se necesita antes comprender bien el problema y plantearse pequeños objetivos que serán necesarios para el buen funcionamiento de la aplicación.

Los sub-objetivos que se han planteado son:

- Entender y comprender a fondo los conceptos que se van a utilizar, documentándose por medio de Internet en tutoriales, foros, transparencias, etc. Todos los conceptos a dominar son:
    - Lenguaje Android
    - Servicio Web
    - Api de Google Maps
    - Protocolos de internet en telefonía móvil
- Diseñar una aplicación en Android compatible con la mayoría de los teléfonos que hay en el mercado.
- Diseñar la aplicación siguiendo los patrones de estilo para que ésta resulte sencilla de aprender y manejar.
- Instalar un servidor propio que ayude a la gestión de la información
- Diseñar un programa para el módulo servidor de tal forma que pueda servir a la aplicación móvil y que mande a los usuarios la información pertinente.
- Utilizar el Api de Google Maps para:
    - Localizar de forma optativa a la persona. De esta forma se podrá mandar la ubicación exacta del lugar en el que el usuario se encuentra.
    - Visualizar un mapa donde señale el punto de reunión.
- Utilizar el servicio de la cámara que ofrece Android en sus terminales móviles para hacer una foto de un lugar característico, un monumento significativo o cualquier cosa del lugar que ayude a reconocer el punto de encuentro.
- Diseñar una base de datos que gestione el alta de los usuarios de tal forma que no de servicio a usuarios no registrados.
- Desarrollar la aplicación que cumpla con los anteriores criterios.

2 Estado del Arte

En este apartado se realiza una investigación amplia sobre el estado en el que se encuentra actualmente las tecnologías que se pretenden utilizar y el desarrollo de ellas a lo largo de la historia. Todo ello ayuda a la comprensión del desarrollo del proyecto.

2.1 Introducción

En este capítulo se va a ver un estudio detallado de las tecnologías que se han utilizado para hacer efectivo el desarrollo de la aplicación. De esta forma se podrá analizar la información para conseguir nuestros objetivos definidos anteriormente con la mejor calidad posible. También nos servirá para enriquecer el vocabulario que se usará a lo largo de la memoria pudiendo dar definiciones precisas para que cualquiera pueda entender y seguir el diseño y desarrollo de la aplicación.

También se mostrarán los productos similares que existen hoy día ya están implementados y que están siendo utilizados por cientos de usuarios, de esta forma se podrá saber cuáles son los defectos que tiene para poder ofrecer un producto más focalizado a las necesidades de los clientes.

El objetivo a alcanzar con este apartado es enriquecer el conocimiento sobre las tecnologías que nos rodean y los productos que se están ofertando para que los usuarios puedan tener una vida más sencilla.

2.2 Android

Es un sistema operativo móvil basado en Linux, que junto con aplicaciones middleware (aplicaciones que se encuentran entre el Sistema Operativo y el usuario) está enfocado para ser utilizado en dispositivos móviles, ya sea para teléfonos Smartphone, tabletas, ciertos ordenadores de poca capacidad, entre otros. Lo que le diferencia de otros sistemas operativos es que está basado en Linux, un núcleo de sistema operativo libre, gratuito y multiplataforma.

Fue desarrollado inicialmente por Android Inc., una firma comprada por Google en 2005. Por ello la mayoría de estos dispositivos que venden hoy día ya tienen muchos de los productos desarrollados por Google integrados e insertados dentro de los terminales.

Android utiliza una variación del lenguaje de programación Java que es diferente a la Java ME. Una máquina virtual sirve para interpretar todo ese código que genera nuestro programa (bytecode) y pueda ejecutarse. Pues bien, Android, tiene una adaptación de esta máquina virtual y se llama Dalvik. Java,Dalvik es una excelente versión que optimiza muchas cosas en la plataforma y te sorprenderá saber que la rapidez de Android y de algunas aplicaciones se debe precisamente a esto.

Cabe mencionar que el sistema operativo proporciona todas las interfaces necesarias para desarrollar aplicaciones que accedan a las funciones del teléfono (como el GPS, las llamadas, la agenda, etc.) de una forma muy sencilla utilizando Java. También incluye su propio Gestor de Base de Datos con el lenguaje SQLite, que más tarde se escribirá sobre ello.

2.2.1 Historia

Android era un sistema operativo para móviles prácticamente desconocido propiedad de una empresa llamada Android Inc. hasta que en 2005 Google lo compró. Actualmente Andy Rubin, el creador de Android, trabaja como vicepresidente de ingeniería de Google y tiene bajo su mando este proyecto.

En noviembre de 2007 sólo circulaban rumores de que el gigante de Internet tenía en planes lanzar un proyecto para móviles, y precisamente por esas fechas se lanzó la *Open Handset Alliance*, que agrupaba a muchos fabricantes de teléfonos móviles, chipsets y Google y se proporcionó la primera versión de Android, junto con el SDK para que los programadores empezaran a crear sus aplicaciones para este sistema.

Aunque los inicios fueran un poco lentos, debido a que se lanzó el sistema operativo antes que el primer móvil, rápidamente se ha colocado como una plataforma que ya se ha ganado muchos adeptos y que ha demostrado una velocidad de madurez importante debido a los diferentes fabricantes que han adaptado interesantes piezas de hardware para acompañar las diferentes funciones de Android.

A principios del año 2011, en febrero, se anunció la versión 3.0 de Android, llamada *Honeycomb*, que está optimizado para tabletas en lugar de teléfonos móviles. Esta novedad viene a poner sobre la mesa otra gama de posibilidades que tiene Android en dispositivos portátiles.

2.2.2 Versiones de Android

Las versiones de Android, aparte del número, tienen un nombre de un postre en idioma inglés. En cada versión el postre elegido empieza con una letra distinta siguiendo un orden alfabético:

- C: Cupcake (v1.5), magdalena glaseada.
- D: Donut (v1.6), rosquilla.
- E: Éclair (v2.0/v2.1), pastel francés conocido en España como pepito o canuto.
- F: Froyo (v2.2), (abreviatura de «frozen yogurt») yogur helado.
- G: Gingerbread (v2.3), pan de jengibre.
- H: Honeycomb (v3.0/v3.1), panal de miel.
- I: IceCream Sandwich (sin número aún), sandwich de helado.

2.2.3 Arquitectura Interna

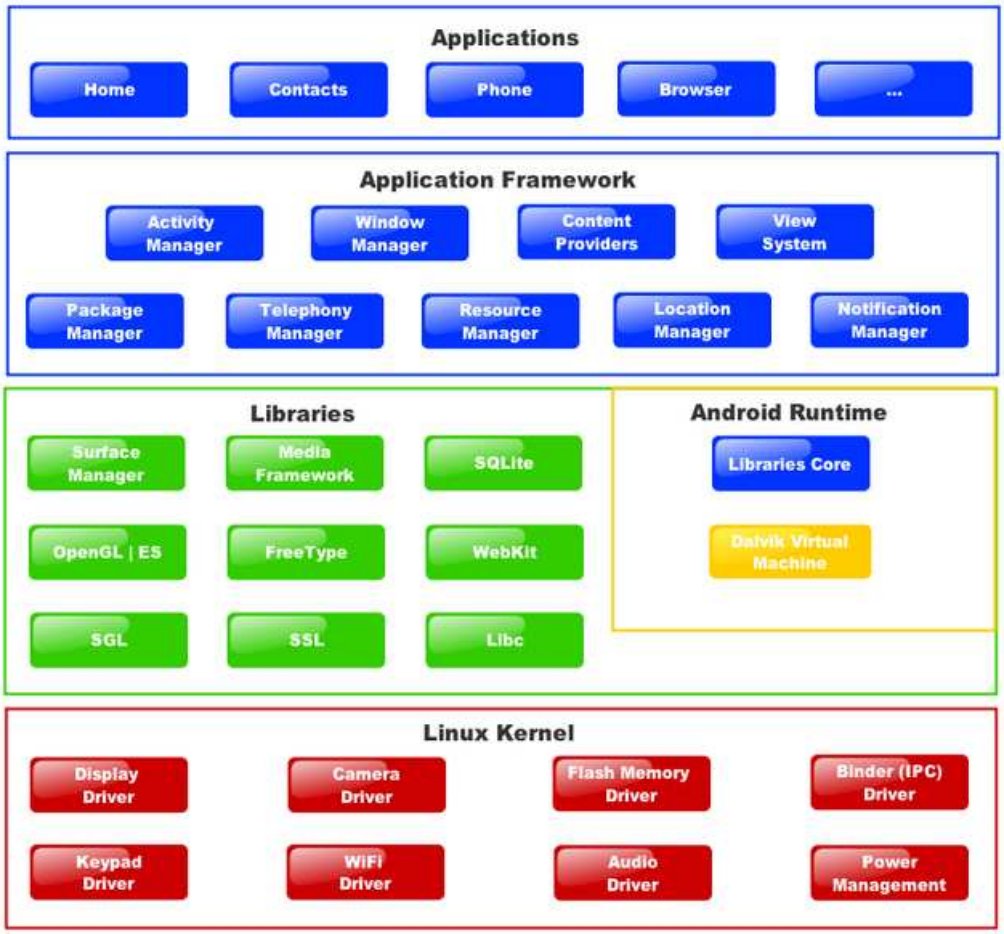
A continuación se presenta la arquitectura interna que lleva el sistema Android. Para poder entender mejor las capas de las que se compone y cada uno de los módulos por los que está compuesto, se presenta dos imágenes a diferente escala, una con una visión totalmente global y otra con una visión más detallada en la que se pueden ver los componentes por los que se compone cada nivel de la arquitectura.

Android es un sistema con una arquitectura por capas. A continuación se muestra una imagen en la que se aprecia las capas y se puede observar cómo está compuesto y montado este sistema que ha conseguido hacer nuestras vidas más sencillas.

Ilustracion 1. Arquitectura Interna Genérica de Android

La arquitectura que compone todo el sistema de Android es una estructura multicapa formada por cinco capas. En un primer básico tiene el Kernel construido en Linux dónde se montan los procesos y aísla del hardware. En un segundo nivel se montan las librerías con las que se proporciona funcionalidad. En tercer nivel se tiene lo que llaman Android Runtime, que sería el entorno de ejecución que es dónde estaría la máquina virtual anteriormente mencionada. En el cuarto nivel está el Framework de aplicaciones que es dónde se ubican los servicios del teléfono. Y en el nivel más alto se encuentran las aplicaciones que se desarrollan y que muchas de ellas podrán llamar a los servicios del teléfono ya que tiene contacto directo con dicha capa.

A continuación se presenta una ilustración de la arquitectura interna con un nivel de detalle más profundo y a continuación se explicarán cada una de las capas con más detalle.



Ilustracion 2. Arquitectura Interna Detallada de Android

A continuación se explica cada una de las capas iniciando de abajo hacia arriba.

- **Kernel de Linux:** Como ya se ha mencionado anteriormente, el núcleo del sistema operativo Android está basado en el kernel de Linux versión 2.6, similar al que puede incluir cualquier distribución de Linux, como Ubuntu, solo que adaptado a las características del hardware en el que se ejecutará Android, es decir, para dispositivos móviles.

  El núcleo actúa como una capa de abstracción entre el hardware y el resto de las capas de la arquitectura. El desarrollador no accede directamente a esta capa, sino que debe utilizar las librerías disponibles en capas superiores. De esta forma también nos evitamos el hecho de quebrarnos la cabeza para conocer las características precisas de cada teléfono. Si necesitamos hacer uso de la cámara, el sistema operativo se encarga de utilizar la que incluya el teléfono, sea cual sea. Para cada elemento de hardware del teléfono existe un controlador (o *driver*) dentro del kernel que permite utilizarlo desde el software.

  El kernel también se encarga de gestionar los diferentes recursos del teléfono (energía, memoria, etc.) y del sistema operativo en sí: procesos, elementos de comunicación (*networking*), etc.

- **Librerías:** La siguiente capa que se sitúa justo sobre el kernel la componen las bibliotecas nativas de Android, también llamadas librerías. Están escritas en C

o C++ y compiladas para la arquitectura hardware específica del teléfono. Estas normalmente están hechas por el fabricante, quien también se encarga de instalarlas en el dispositivo antes de ponerlo a la venta. El objetivo de las librerías es proporcionar funcionalidad a las aplicaciones para tareas que se repiten con frecuencia, evitando tener que codificarlas cada vez y garantizando que se llevan a cabo de la forma "más eficiente".

Entre las librerías incluidas habitualmente encontramos OpenGL (motor gráfico), Bibliotecas multimedia (formatos de audio, imagen y video), Webkit (navegador), SSL (cifrado de comunicaciones), FreeType (fuentes de texto), SQLite (base de datos), entre otras.

- **Entorno de ejecución:** Como podemos apreciar en el diagrama, el entorno de ejecución de Android no se considera una capa en sí mismo, dado que también está formado por librerías. Aquí encontramos las librerías con las funcionalidades habituales de Java así como otras específicas de Android.

  El componente principal del entorno de ejecución de Android es la máquina virtual *Dalvik*. Las aplicaciones se codifican en Java y son compiladas en un formato específico para que esta máquina virtual las ejecute. La ventaja de esto es que las aplicaciones se compilan una única vez y de esta forma estarán listas para distribuirse con la total garantía de que podrán ejecutarse en cualquier dispositivo Android que disponga de la versión mínima del sistema operativo que requiera la aplicación.

  Cabe aclarar que Dalvik es una variación de la máquina virtual de Java, por lo que no es compatible con el bytecode Java. Java se usa únicamente como lenguaje de programación, y los ejecutables que se generan con el SDK de Android tienen la extensión .dex que es específico para Dalvik, y por ello no podemos correr aplicaciones Java en Android ni viceversa.

- **Framework de aplicaciones:** La siguiente capa está formada por todas las clases y servicios que utilizan directamente las aplicaciones para realizar sus funciones. La mayoría de los componentes de esta capa son librerías Java que acceden a los recursos de las capas anteriores a través de la máquina virtual Dalvik. Siguiendo el diagrama encontramos:

  1. ***Activity Manager:*** Se encarga de administrar la pila de actividades de nuestra aplicación así como su ciclo de vida.
  2. ***Windows Manager:*** Se encarga de organizar lo que se mostrará en pantalla. Básicamente crea las superficies en la pantalla que posteriormente pasarán a ser ocupadas por las actividades.
  3. ***Content Provider:*** Esta librería es muy interesante porque crea una capa que encapsula los datos que se compartirán entre aplicaciones para tener control sobre cómo se accede a la información.
  4. ***Views:*** En Android, las vistas los elementos que nos ayudarán a construir las interfaces de usuario: botones, cuadros de texto, listas y hasta elementos más avanzados como un navegador web o un visor de Google Maps.
  5. ***Notification Manager:*** Engloba los servicios para notificar al usuario cuando algo requiera su atención mostrando alertas en la barra de estado.

Un dato importante es que esta biblioteca también permite jugar con sonidos, activar el vibrador o utilizar los LEDs del teléfono en caso de tenerlos.

6. ***Package Manager:*** Esta biblioteca permite obtener información sobre los paquetes instalados en el dispositivo Android, además de gestionar la instalación de nuevos paquetes. Con paquete nos referimos a la forma en que se distribuyen las aplicaciones Android, estos contienen el archivo .apk, que a su vez incluyen los archivos .dex con todos los recursos y archivos adicionales que necesite la aplicación, para facilitar su descarga e instalación.
7. ***Telephony Manager:*** Con esta librería podremos realizar llamadas o enviar y recibir SMS/MMS, aunque no permite reemplazar o eliminar la actividad que se muestra cuando una llamada está en curso.
8. ***Resource Manager:*** Con esta librería podremos gestionar todos los elementos que forman parte de la aplicación y que están fuera del código, es decir, cadenas de texto traducidas a diferentes idiomas, imágenes, sonidos o layouts. En un post relacionado a la estructura de un proyecto Android veremos esto más a fondo.
9. ***Location Manager:*** Permite determinar la posición geográfica del dispositivo Android mediante GPS o redes disponibles y trabajar con mapas.
10. ***Sensor Manager:*** Nos permite manipular los elementos de hardware del teléfono como el acelerómetro, giroscopio, sensor de luminosidad, sensor de campo magnético, brújula, sensor de presión, sensor de proximidad, sensor de temperatura, etc.
11. ***Cámara:*** Con esta librería podemos hacer uso de la(s) cámara(s) del dispositivo para tomar fotografías o para grabar vídeo.
12. ***Multimedia:*** Permiten reproducir y visualizar audio, vídeo e imágenes en el dispositivo.

- **Aplicaciones:** En la última capa se incluyen todas las aplicaciones del dispositivo, tanto las que tienen interfaz de usuario como las que no, las nativas (programadas en C o C++) y las administradas (programadas en Java), las que vienen preinstaladas en el dispositivo y aquellas que el usuario ha instalado.

  En esta capa encontramos también la aplicación principal del sistema: Inicio (Home) o lanzador (launcher), porque es la que permite ejecutar otras aplicaciones mediante una lista y mostrando diferentes escritorios donde se pueden colocar accesos directos a aplicaciones o incluso widgets, que son también aplicaciones de esta capa.

  Por todo ello Android nos proporciona un entorno sumamente poderoso para que podamos programar aplicaciones que hagan cualquier cosa. Nada dentro de Android es inaccesible y podemos jugar siempre con las aplicaciones de nuestro teléfono para optimizar cualquier tarea.

  El potencial de Android se sitúa en el control total que se le da al usuario para que haga de su teléfono un dispositivo a su medida.

2.3 SQLite

Como se ha mencionado anteriormente SQLite es un Gestor de Bases de Datos relacionales (GBBDD)

SQLite es un sistema de gestión de bases de datos relacional compatible con ACID, contenida en una relativamente pequeña (~275 kiB)[2] biblioteca escrita en C. SQLite es un proyecto de dominio público[1] creado por *D. Richard Hipp*.

A diferencia de los sistemas de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host. Este diseño simple se logra bloqueando todo el fichero de base de datos al principio de cada transacción.

En su versión 3, **SQLite** permite bases de datos de hasta 2 Terabytes de tamaño, y también permite la inclusión de campos tipo BLOB.

El autor de SQLite ofrece formación, contratos de soporte técnico y características adicionales como compresión y cifrado.

2.4 JAVA

Según la página oficial de Java se define como: *Java le permite jugar en línea, participar en sesiones de chat con internautas de todo el mundo, calcular los intereses de una hipoteca y ver imágenes en tres dimensiones, entre otras muchas aplicaciones. Es también esencial para las aplicaciones de intranet y otras soluciones de comercio electrónico que constituyen la base informática de las empresas.*

Hasta la fecha, la plataforma Java ha atraído a más de 9 millones de desarrolladores de software. Se utiliza en los principales sectores industriales y está presente en una amplia gama de dispositivos, computadoras y redes.

La versatilidad, eficacia, portabilidad de plataformas y seguridad de la tecnología Java la convierte en la tecnología ideal para la informática de redes. Desde portátiles hasta centros de datos, desde consolas para juegos hasta súper computadoras, desde teléfonos móviles hasta Internet, Java está en todas partes.

- 1,100 millones de escritorios ejecutan Oracle Java
- 930 millones de descargas de Java Runtime Environment cada año
- mil millones de teléfonos móviles ejecutan Java
- Se entregan 31 veces más al año teléfonos Java que Apple y Android juntos
- El 100% de los reproductores de Blu-ray ejecutan Java
- Se fabrican 1400 millones de tarjetas Java cada año
- Decodificadores, impresoras, cámaras web, juegos, sistemas de navegación en vehículos, terminales de loterías, dispositivos médicos, estaciones de pago de aparcamientos y mucho más.

Claramente esta información es algo imprecisa y por ello se ha contrastado con otras fuentes que explican Java de una forma más extensa.

Java es un lenguaje de programación de alto nivel ***orientado a objetos***, desarrollado por ***James Gosling*** en 1995. El lenguaje en sí mismo toma mucha de su sintaxis de C, Cobol y Visual Basic, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. La memoria es gestionada mediante un recolector de basura.

Las aplicaciones Java están típicamente compiladas en un ***bytecode***, aunque la compilación en código máquina nativo también es posible. En el tiempo de ejecución, el *bytecode* es normalmente interpretado o compilado a código nativo para la ejecución, aunque la ejecución directa por hardware del *bytecode* por un procesador Java también es posible.

La implementación original y de referencia del compilador, la máquina virtual y las bibliotecas de clases de Java fueron desarrollados por Sun Microsystems en 1995. Desde entonces, Sun ha controlado las especificaciones, el desarrollo y evolución del lenguaje a través del ***Java Community Process***, si bien otros han desarrollado también implementaciones alternativas de estas tecnologías de Sun, algunas incluso bajo licencias de software libre.

Entre diciembre de 2006 y mayo de 2007, Sun Microsystems liberó la mayor parte de sus tecnologías Java bajo la licencia GNU GPL, de acuerdo con las especificaciones del Java Community Process, de tal forma que prácticamente todo el Java de Sun es ahora software libre aunque la biblioteca de clases de páginas web comprendidas en las librerías de objetación de objetos para ser compilados como aplicaciones comprimidas no están totalmente acopladas de acuerdo con Sun que dice que se requiere un intérprete para ejecutar los programas de Java.

2.4.1 Historia

Java se creó como una herramienta de programación para ser usada en un proyecto de set-top-box en una pequeña operación denominada *the Green Project* en Sun Microsystems en el año1991. El equipo (*Green Team*), compuesto por trece personas y dirigido por James Gosling, trabajó durante 18 meses en Sand Hill Road en Menlo Park en su desarrollo.

El lenguaje se denominó inicialmente *Oak* (por un roble que había fuera de la oficina de Gosling), luego pasó a denominarse *Green* tras descubrir que *Oak* era ya una marca comercial registrada para adaptadores de tarjetas gráficas y finalmente se renombró a *Java*.

El término Java fue acuñado en una cafetería frecuentada por algunos de los miembros del equipo. Pero no está claro si es un acrónimo o no, aunque algunas fuentes señalan que podría tratarse de las iniciales de sus creadores: ***J****ames Gosling,* ***A****rthur* ***V****an Hoff, y* ***A****ndy Bechtolsheim*. Otros abogan por el siguiente acrónimo, ***J****ust* ***A****nother* ***V****ague* ***A****cronym* ("sólo otro acrónimo ambiguo más"). La hipótesis que más fuerza tiene es la que Java debe su nombre a un tipo de café disponible en la cafetería cercana, de ahí que el icono de java sea una taza de café caliente. Un pequeño signo que da fuerza a esta teoría es que los 4 primeros bytes (el *número mágico*) de los archivos.class que genera el compilador, son en hexadecimal, 0xCAFEBABE. A pesar de todas estas teorías, el nombre fue sacado al parecer de una lista aleatoria de palabras.[1]

Los objetivos de Gosling eran implementar una máquina virtual y un lenguaje con una estructura y sintaxis similar a C++. Entre junio y julio de 1994, tras una sesión maratoniana de tres días entre John Gaga, James Gosling, Patrick Naughton, Wayne Rosing y Eric Schmidt, el

equipo reorientó la plataforma hacia la Web. Sintieron que la llegada del navegador web Mosaic, propiciaría que Internet se convirtiese en un medio interactivo, como el que pensaban era la televisión por cable. Naughton creó entonces un prototipo de navegador, WebRunner, que más tarde sería conocido como HotJava.

En 1994, se les hizo una demostración de HotJava y la plataforma Java a los ejecutivos de Sun. Java 1.0a pudo descargarse por primera vez en 1994, pero hubo que esperar al 23 de mayo de 1995, durante las conferencias de SunWorld, a que vieran la luz pública Java y HotJava, el navegador Web. El acontecimiento fue anunciado por John Gage, el Director Científico de Sun Microsystems. El acto estuvo acompañado por una pequeña sorpresa adicional, el anuncio por parte de Marc Andreessen, Vicepresidente Ejecutivo de Netscape, de que Java sería soportado en sus navegadores. El 9 de enero del año siguiente, 1996, Sun fundó el grupo empresarial JavaSoft para que se encargase del desarrollo tecnológico. [1] Dos semanas más tarde la primera versión de Java fue publicada.

La promesa inicial de Gosling era *Write Once, Run Anywhere* (Escríbelo una vez, ejecútalo en cualquier lugar), proporcionando un lenguaje independiente de la plataforma y un entorno de ejecución (la JVM) ligero y gratuito para las plataformas más populares de forma que los binarios (bytecode) de las aplicaciones Java pudiesen ejecutarse en cualquier plataforma.

El entorno de ejecución era relativamente seguro y los principales navegadores web pronto incorporaron la posibilidad de ejecutar applets Java incrustadas en las páginas web.

Java ha experimentado numerosos cambios desde la versión primigenia, JDK 1.0, así como un enorme incremento en el número de clases y paquetes que componen la biblioteca estándar.[2]

Desde J2SE 1.4, la evolución del lenguaje ha sido regulada por el JCP (***Java Community Process***), que usa ***Java Specification Requests*** (JSRs) para proponer y especificar cambios en la plataforma Java. El lenguaje en sí mismo está especificado en la ***Java Language Specification*** (JLS), o Especificación del Lenguaje Java. Los cambios en los JLS son gestionados en ***JSR 901***

2.4.2 Filosofía

El lenguaje Java se creó con cinco objetivos principales:

1. Debería usar el paradigma de la programación orientada a objetos.
2. Debería permitir la ejecución de un mismo programa en múltiples sistemas operativos.
3. Debería incluir por defecto soporte para trabajo en red.
4. Debería diseñarse para ejecutar código en sistemas remotos de forma segura.
5. Debería ser fácil de usar y tomar lo mejor de otros lenguajes orientados a objetos, como C++.

Para conseguir la ejecución de código remoto y el soporte de red, los programadores de Java a veces recurren a extensiones como CORBA (***Common Object Request Broker Architecture***),***Internet Communications Engine*** o ***OSGi*** respectivamente.

2.4.2.1 Orientado a Objetos

La primera característica, orientado a objetos ("OO"), se refiere a un método de programación y al diseño del lenguaje. Aunque hay muchas interpretaciones para OO, una primera idea es diseñar el software de forma que los distintos tipos de datos que usen estén unidos a sus operaciones. Así, los datos y el código (funciones o métodos) se combinan en entidades llamadas objetos. Un objeto puede verse como un paquete que contiene el "comportamiento" (el código) y el "estado" (datos). El principio es separar aquello que cambia de las cosas que permanecen inalterables. Frecuentemente, cambiar una estructura de datos implica un cambio en el código que opera sobre los mismos, o viceversa. Esta separación en objetos coherentes e independientes ofrece una base más estable para el diseño de un sistema software. El objetivo es hacer que grandes proyectos sean fáciles de gestionar y manejar, mejorando como consecuencia su calidad y reduciendo el número de proyectos fallidos. Otra de las grandes promesas de la programación orientada a objetos es la creación de entidades más genéricas (objetos) que permitan la reutilización del software entre proyectos, una de las premisas fundamentales de la Ingeniería del Software. Un objeto genérico "cliente", por ejemplo, debería en teoría tener el mismo conjunto de comportamiento en diferentes proyectos, sobre todo cuando estos coinciden en cierta medida, algo que suele suceder en las grandes organizaciones. En este sentido, los objetos podrían verse como piezas reutilizables que pueden emplearse en múltiples proyectos distintos, posibilitando así a la industria del software a construir proyectos de envergadura empleando componentes ya existentes y de comprobada calidad; conduciendo esto finalmente a una reducción drástica del tiempo de desarrollo. Podemos usar como ejemplo de objeto el aluminio. Una vez definidos datos (peso, maleabilidad, etc.), y su "comportamiento" (soldar dos piezas, etc.), el objeto "aluminio" puede ser reutilizado en el campo de la construcción, del automóvil, de la aviación, etc.

La reutilización del software ha experimentado resultados dispares, encontrando dos dificultades principales: el diseño de objetos realmente genéricos es pobremente comprendido, y falta una metodología para la amplia comunicación de oportunidades de reutilización. Algunas comunidades de "código abierto" (open source) quieren ayudar en este problema dando medios a los desarrolladores para diseminar la información sobre el uso y versatilidad de objetos reutilizables y bibliotecas de objetos.

2.4.2.2 Independencia de la Plataforma

La segunda característica, la independencia de la plataforma, significa que programas escritos en el lenguaje Java pueden ejecutarse igualmente en cualquier tipo de hardware. Este es el significado de ser capaz de escribir un programa una vez y que pueda ejecutarse en cualquier dispositivo, tal como reza el axioma de Java, '***''write once, run anywhere*** '''.

Para ello, se compila el código fuente escrito en lenguaje Java, para generar un código conocido como "bytecode" (específicamente Java bytecode) — instrucciones máquina simplificadas específicas de la plataforma Java. Esta pieza está "a medio camino" entre el código fuente y el código máquina que entiende el dispositivo destino. El bytecode es ejecutado entonces en la máquina virtual (JVM), un programa escrito en código nativo de la plataforma destino (que es el que entiende su hardware), que interpreta y ejecuta el código. Además, se suministran bibliotecas adicionales para acceder a las características de cada dispositivo (como los gráficos, ejecución mediante hebras o threads, la interfaz de red) de forma unificada. Se debe tener presente que, aunque hay una etapa explícita de compilación, el bytecode generado es interpretado o convertido a instrucciones máquina del código nativo por el compilador JIT (Just In Time).

Hay implementaciones del compilador de Java que convierten el código fuente directamente en código objeto nativo, como GCJ. Esto elimina la etapa intermedia donde se genera el bytecode, pero la salida de este tipo de compiladores sólo puede ejecutarse en un tipo de arquitectura.

La licencia sobre Java de Sun insiste que todas las implementaciones sean "compatibles". Esto dio lugar a una disputa legal entre Microsoft y Sun, cuando éste último alegó que la implementación de Microsoft no daba soporte a las interfaces RMI y JNI además de haber añadido características ''dependientes'' de su plataforma. Sun demandó a Microsoft y ganó por daños y perjuicios (unos 20 millones de dólares) así como una orden judicial forzando la acatación de la licencia de Sun. Como respuesta, Microsoft no ofrece Java con su versión de sistema operativo, y en recientes versiones de Windows, su navegador Internet Explorer no admite la ejecución de applets sin un conector (o plugin) aparte. Sin embargo, Sun y otras fuentes ofrecen versiones gratuitas para distintas versiones de Windows.

Las primeras implementaciones del lenguaje usaban una máquina virtual interpretada para conseguir la portabilidad. Sin embargo, el resultado eran programas que se ejecutaban comparativamente más lentos que aquellos escritos en C o C++. Esto hizo que Java se ganase una reputación de lento en rendimiento. Las implementaciones recientes de la JVM dan lugar a programas que se ejecutan considerablemente más rápido que las versiones antiguas, empleando diversas técnicas, aunque sigue siendo mucho más lento que otros lenguajes.

La primera de estas técnicas es simplemente compilar directamente en código nativo como hacen los compiladores tradicionales, eliminando la etapa del bytecode. Esto da lugar a un gran rendimiento en la ejecución, pero tapa el camino a la portabilidad. Otra técnica, conocida como compilación JIT (Just In Time, o '''compilación al vuelo'''), convierte el bytecode a código nativo cuando se ejecuta la aplicación. Otras máquinas virtuales más sofisticadas usan una '''recompilación dinámica''' en la que la VM es capaz de analizar el comportamiento del programa en ejecución y recompila y optimiza las partes críticas. La recompilación dinámica puede lograr mayor grado de optimización que la compilación tradicional (o estática), ya que puede basar su trabajo en el conocimiento que de primera mano tiene sobre el entorno de ejecución y el conjunto de clases cargadas en memoria. La compilación JIT y la recompilación dinámica permiten a los programas Java aprovechar la velocidad de ejecución del código nativo sin por ello perder la ventaja de la portabilidad en ambos.

La portabilidad es técnicamente difícil de lograr, y el éxito de Java en ese campo ha sido dispar. Aunque es de hecho posible escribir programas para la plataforma Java que actúen de forma correcta en múltiples plataformas de distinta arquitectura, el gran número de estas con pequeños errores o inconsistencias llevan a que a veces se parodie el eslogan de Sun, "***Write once, run anywhere***" como "***Write once, debug everywhere***" (o "Escríbelo una vez, ejecútalo en cualquier parte" por "Escríbelo una vez, depúralo en todas partes")

El concepto de independencia de la plataforma de Java cuenta, sin embargo, con un gran éxito en las aplicaciones en el entorno del servidor, como los Servicios Web, los Servlets, los Java Beans, así como en sistemas empotrados basados en OSGi, usando entornos Java empotrados.

2.4.2.3 Recolector de Basuras

En Java el problema de las fugas de memoria se evita en gran medida gracias a la recolección de basura (***o automatic garbage collector***). El programador determina cuándo se crean los objetos y el entorno en tiempo de ejecución de Java (Java runtime) es el responsable de gestionar el ciclo de vida de los objetos. El programa, u otros objetos pueden tener localizado un objeto mediante una referencia a éste. Cuando no quedan referencias a un objeto, el recolector de basura de Java borra el objeto, liberando así la memoria que ocupaba previniendo posibles fugas (ejemplo: un objeto creado y únicamente usado dentro de un método sólo tiene entidad dentro de éste; al salir del método el objeto es eliminado). Aún así, es posible que se produzcan fugas de memoria si el código almacena referencias a objetos que ya no son necesarios—es decir, pueden aún ocurrir, pero en un nivel conceptual superior. En definitiva, el recolector de basura de Java permite una fácil creación y eliminación de objetos y mayor seguridad.

2.5 SOAP

SOAP (siglas de ***Simple Object Access Protocol***) es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Este protocolo deriva de un protocolo creado por ***David Winer*** en 1998, llamado XML-RPC. SOAP fue creado por Microsoft, IBM y otros y está actualmente bajo el auspicio de la W3C. Es uno de los protocolos utilizados en los servicios Web.

2.5.1 Características

SOAP puede formar la capa base de una "***pila de protocolo de web service***", ofreciendo un framework de mensajería básica en la cual los web services se puedan construir. Este protocolo basado en XML consiste de tres partes: un sobre (envelope), el cual define qué hay en el mensaje y cómo procesarlo; un conjunto de reglas de codificación para expresar instancias de tipos de datos; y una conversión para representar llamadas a procedimientos y respuestas. El protocolo SOAP tiene tres características principales:

- **Extensibilidad** (seguridad y WS-routing son extensiones aplicadas en el desarrollo).
- **Neutralidad** (SOAP puede ser utilizado sobre cualquier protocolo de transporte como HTTP, SMTP, TCP o JMS).
- **Independencia** (SOAP permite cualquier modelo de programación).

Como ejemplo de cómo los procedimientos SOAP pueden ser utilizados, un mensaje SOAP podría ser enviado a un sitio Web que tiene habilitado Web service, para realizar la búsqueda de algún precio en una base de datos, indicando los parámetros necesitados en la consulta. El sitio podría retornar un documento formateado en XML con el resultado, ejemplo, precios, localización, características. Teniendo los datos de respuesta en un formato estandarizado "parseable", este puede ser integrado directamente en un sitio Web o aplicación externa.

La arquitectura SOAP consiste de muchas capas de especificación: para el formato del mensaje, MEP (***Message Exchange Patterns***), subyacentes enlaces de protocolo de transporte, modelo de procesamiento de mensajes, y extensibilidad del protocolo. SOAP es el sucesor de XML-RPC, a pesar de que toma el transporte y la neutralidad de la interacción y el ***envelope / header /body*** de otra parte (probablemente de WDDX).

2.5.2 Historia

SOAP fue designado como un protocolo de acceso a objetos en 1998 por ***Daved Winer***, ***Don Box***, ***Bob Atkinson*** y ***Mohsen Al-Ghosein*** por Microsoft, donde Atkinson y Al-Ghosein trabajaban en aquel entonces. La especificación SOAP actualmente es mantenida por el XML ***Protocol Working Group*** del ***World Wide Web Consortium***.

SOAP originalmente significaba "***Simple Object Access Protocol***", pero esta sigla se abandonó con la versión 1.2 de la norma. La versión 1.2 se convirtió en una recomendación del W3C el 24 de junio de 2003. El acrónimo se confunde a veces con SOA, siglas de arquitectura orientada a servicios, pero las siglas no están relacionadas.

Después que SOAP se introdujo por primera vez, se convirtió en la capa subyacente de un conjunto más complejo de los web services, basada en la WSDL (***Web Services Description Language***) y UDDI (***Universal Description Discovery and Integration***). Estos servicios, especialmente UDDI, han demostrado ser de mucho menos interés, pero una apreciación de ellos da una comprensión más completa del esperado rol de SOAP comparado a como los web services están actualmente desarrollados.

2.6 Sockets

Posiblemente a estas alturas de la era de Internet se haya escuchado muchas definiciones muy precisas y otras muy generales de lo que es realmente un Socket. Entre ellos podemos ver las siguientes definiciones.

Socket designa un concepto abstracto por el cual dos programas (posiblemente situados en computadoras distintas) pueden intercambiar cualquier flujo de datos, generalmente de manera fiable y ordenada.

El término *socket* es también usado como el nombre de una interfaz de programación de aplicaciones (API) para la familia de protocolos de Internet TCP/IP, provista usualmente por el sistema operativo.

Los *sockets* de Internet constituyen el mecanismo para la entrega de paquetes de datos provenientes de la tarjeta de red a los procesos o hilos apropiados. Un *socket* queda definido por un par de direcciones IP local y remota, un protocolo de transporte y un par de números de puerto local y remoto.

En definitiva es uno de los mecanismos de comunicación usados para comunicar un cliente con un servidor o dos clientes, aunque su uso más normal es para un servidor. El problema que tiene es que, al tener un puerto totalmente abierto, dejas en peligro a todo tu sistema, ya que el socket se salta toda la seguridad de los firewalls. Las ventajas que proporciona es que puedes intercambiar cualquier tipo de dato que quieras y permite la programación de tu propio protocolo de actuación.

2.7 Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma, típicamente ha sido usada para desarrollar entornos de desarrollo integrados (del inglés IDE), como el IDE de Java llamado ***Java Development Toolkit*** (JDT) y el compilador (ECJ) que se entrega como parte de Eclipse (y que son usados también para desarrollar el mismo Eclipse). Sin embargo, también se puede usar para otros tipos de aplicaciones cliente, como BitTorrent o Azureus.

Eclipse es también una comunidad de usuarios, extendiendo constantemente las áreas de aplicación cubiertas. Un ejemplo es el recientemente creado Eclipse ***Modeling Project***, cubriendo casi todas las áreas de ***Model Driven Engineering***.

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para ***VisualAge***. Eclipse es ahora desarrollado por la Fundación Eclipse, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto y un conjunto de productos complementarios, capacidades y servicios.

Eclipse fue liberado originalmente bajo la ***Common Public License***, pero después fue re-licenciado bajo la ***Eclipse Public License***. La ***Free Software Foundation*** ha dicho que ambas licencias son licencias de software libre, pero son incompatibles con Licencia pública general de GNU (GNU GPL).[3]

2.7.1 Arquitectura

La base para Eclipse es la Plataforma de cliente enriquecido (del Inglés ***Rich Client Platform*** RCP). Los siguientes componentes constituyen la plataforma de cliente enriquecido:

- Plataforma principal - inicio de Eclipse, ejecución de plugins
- ***OSGi*** - una plataforma para bundling estándar.
- El ***Standard Widget Toolkit*** (SWT) - Un widget toolkit portable.
- ***JFace*** - manejo de archivos, manejo de texto, editores de texto
- El Workbench de Eclipse - vistas, editores, perspectivas, asistentes

Los widgets de Eclipse están implementados por una herramienta de widget para Java llamada SWT, a diferencia de la mayoría de las aplicaciones Java, que usan las opciones estándar ***Abstract Window Toolkit*** (AWT) o Swing. La interfaz de usuario de Eclipse también tiene una capa GUI intermedia llamada JFace, la cual simplifica la construcción de aplicaciones basadas en SWT.

El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés *plug-in*) para proporcionar toda su funcionalidad al frente de la plataforma de cliente enriquecido, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse usando otros lenguajes de programación como son ***C/C++***y ***Python***, permite a Eclipse trabajar con lenguajes para procesado de texto como ***LaTeX***, aplicaciones en red como Telnet y Sistema de Gestión de Bases de Datos (SGBBDD). La arquitectura plugin permite escribir cualquier extensión deseada en el ambiente, como sería Gestión de la configuración. Se provee soporte para Java y CVS en el SDK de Eclipse. Y no tiene por qué ser usado únicamente para soportar otros lenguajes de programación.

La definición que da el proyecto Eclipse acerca de su software es: "***una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular***".

En cuanto a las aplicaciones clientes, Eclipse provee al programador con frameworks muy ricos para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc. Por ejemplo, GEF (Graphic Editing Framework - Framework para la edición gráfica) es un plugin de Eclipse para el desarrollo de editores visuales que pueden ir desde procesadores de texto wysiwyg hasta editores de diagramas UML, interfaces gráficas para el usuario (GUI), etc. Dado que los editores realizados con GEF "viven" dentro de Eclipse, además de poder ser usados conjuntamente con otros plugins, hacen uso de su interfaz gráfica personalizable y profesional.

El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código. Mediante diversos plugins estas herramientas están también disponibles para otros lenguajes como C/C++ (Eclipse CDT) y en la medida de lo posible para lenguajes de script no tipados como PHP o Javascript. El IDE también hace uso de un espacio de trabajo, en este caso un grupo de metadata en un espacio para archivos plano, permitiendo modificaciones externas a los archivos en tanto se refresque el espacio de trabajo correspondiente.

2.7.2 Características

Eclipse dispone de un Editor de texto con resaltado de sintaxis. La compilación es en tiempo real. Tiene pruebas unitarias con ***JUnit***, control de versiones con CVS, integración con ***Ant***, asistentes (*wizards*) para creación de proyectos, clases, tests, etc., y refactorización.

Asimismo, a través de "plugins" libremente disponibles es posible añadir control de versiones con Subversion e integración con ***Hibernate***.

2.7.3 Historia

Eclipse comenzó como un proyecto de ***IBM Canadá***. Fue desarrollado por OTI (***Object Technology International***) como reemplazo de ***VisualAge*** también desarrollado por OTI. En noviembre del 2001, se formó un consorcio para el desarrollo futuro de Eclipse como código abierto. En 2003, fue creada la fundación independiente de IBM.

Resumen de las versiones de Eclipse:

Versión	Fecha de lanzamiento	Versión de plataforma	Proyectos
Juno	8 de junio de 2012	4.2	Juno projects
Indigo	22 de junio de 2011	3.7	Indigo projects
Helios	23 junio de 2010	3.6	Helios projects
Galileo	24 de junio de 2009	3.5	Galileo projects
Ganymede	25 junio de 2008	3.4	Ganymede projects
Europa	29 de junio de 2007	3.3	Europa projects
Callisto	30 de junio de 2006	3.2	Callisto projects
Eclipse 3.1	28 de junio 2005	3.1	
Eclipse 3.0	28 de junio de 2004	3.0	

Tabla 1: Versiones de Eclipse

2.8 Netbeans

Es un entorno de desarrollo integrado y libre, el cuál fue diseñado principalmente para desarrollar proyectos con el lenguaje de programación Java. Actualmente se han integrado un gran número de módulos que extienden este entorno a otros lenguajes. Más adelante se especifican los módulos que se han integrado. Una de las ventajas fundamentales de Netbeans es que es un producto libre y gratuito sin restricciones de uso.

Netbeans está integrado por un gran número de usuarios que componen una sociedad que se encuentra en constante crecimiento y que tiene cerca de 100 socios en todo el mundo. Fue fundado por Sun MicroSystems en junio de 2000 tras estos años sigue siendo el patrocinador principal de los proyectos.

La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados ***módulos***. Un módulo es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que los módulos pueden ser desarrollados independientemente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software.

2.8.1 Historia

NetBeans comenzó como un proyecto estudiantil en la República Checa (originalmente llamado Xelfi), en 1996 bajo la tutoría de la Facultad de Matemáticas y Física en la ***Universidad Carolina*** en Praga. La meta era escribir un entorno de desarrollo integrado (IDE) para Java parecido a Delphi. Xelfi fue el primer IDE escrito en Java; tuvo su primer prelanzamiento en 1997.

Xelfi fue un proyecto divertido para trabajar, ya que los IDE escritos en Java eran un territorio desconocido en esa época. El proyecto atrajo suficiente interés, por lo que los estudiantes, después de graduarse, decidieron que lo podían convertir en un proyecto comercial. Prestando espacios web de amigos y familiares, formaron una compañía alrededor de esto. Casi todos ellos siguen trabajando en NetBeans.

Tiempo después, fueron contactados por Roman Stanek, un empresario que ya había estado relacionado con varias iniciativas en la República Checa. Estaba buscando una buena idea en la que invertir, y encontró en Xelfi una buena oportunidad. Así, tras una reunión, el negocio surgió.

El plan original era desarrollar unos componentes JavaBeans para redes. Jarda Tulach, quien diseñó la arquitectura básica de la IDE, propuso la idea de llamarlo NetBeans, a fin de describir este propósito. Cuando las especificaciones de los Enterprise JavaBeans salieron, decidieron trabajar con este estándar, ya que no tenía sentido competir contra él, sin embargo permaneció el nombre de NetBeans.

En la primavera de 1999, Netbeans DeveloperX2 fue lanzado, soportando Swing. Las mejoras de rendimiento que llegaron con el JDK 1.3, lanzado en otoño de 1999, hicieron de NetBeans una alternativa realmente viable para el desarrollo de herramientas. En el verano de 1999, el equipo trabajó duro para rediseñar DeveloperX2 en un NetBeans más modular, lo que lo convirtió en la base de NetBeans hoy en día.

Algo más ocurrió en el verano de 1999. Sun Microsystems quería una herramienta mejor de desarrollo en Java, y comenzó a estar interesado en NetBeans. En otoño de 1999, con la nueva generación de NetBeans en Beta, se llegaría a un acuerdo.

Sun adquirió otra compañía de herramientas al mismo tiempo, Forté, y decidió renombrar NetBeans a Forté for Java. El nombre de NetBeans desapareció por un tiempo.

Seis meses después, se tomó la decisión de hacer a NetBeans open source. Mientras que Sun había contribuido considerablemente con líneas de código en varios proyectos de código abierto a través de los años, NetBeans se convirtió en el primer proyecto de código abierto patrocinado por ellos. En junio del 2000 NetBeans.org fue lanzado.

2.8.2 Plataforma

Durante el desarrollo del NetBeans IDE ocurrió una cosa interesante. La gente empezó a construir aplicaciones usando el NetBeans core runtime con sus propios plug-ins, de hecho, esto se convirtió en un mercado bastante grande.

La Plataforma NetBeans es una base modular y extensible usada como una estructura de integración para crear aplicaciones de escritorio grandes. Empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones.

La plataforma ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma están:

- Administración de las interfaces de usuario (ej. menús y barras de herramientas)
- Administración de las configuraciones del usuario
- Administración del almacenamiento (guardando y cargando cualquier tipo de dato)
- Administración de ventanas
- Framework basado en asistentes (diálogos paso a paso)

2.8.3 IDE

El IDE NetBeans es un entorno de desarrollo integrado - una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso.

El NetBeans IDE es un IDE de código abierto escrito completamente en Java usando la plataforma NetBeans. El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactoring.

NetBeans IDE 6.5, la cual fue publicada el 19 de noviembre de 2008, extiende las características existentes del Java EE (incluyendo Soporte a Persistencia, EJB 3 y JAX-WS). Adicionalmente, el NetBeans Enterprise Pack soporta el desarrollo de Aplicaciones empresariales con Java EE 5, incluyendo herramientas de desarrollo visuales de SOA, herramientas de esquemas XML, orientación a web servicies (for BPEL), y modelado UML. El NetBeans C/C++ Pack soporta proyectos de C/C++, mientras el PHP Pack, soporta PHP 5.

Modularidad. Todas las funciones del IDE son provistas por módulos. Cada módulo provee una función bien definida, tales como el soporte de Java, edición, o soporte para el sistema de control de versiones. NetBeans contiene todos los módulos necesarios para el desarrollo de aplicaciones Java en una sola descarga, permitiéndole al usuario comenzar a trabajar inmediatamente.

Sun Studio, ***Sun Java Studio Enterprise***, y ***Sun Java Studio Creator*** de Sun Microsystems han sido todos basados en el IDE NetBeans.

Desde julio de 2006, NetBeans IDE es licenciado bajo la Common Development and Distribution License (CDDL), una licencia basada en la ***Mozilla Public License*** (MPL).

2.8.4 Versiones

Las versiones que se han publicado se muestran en la siguiente tabla:

Versión	Fecha de lanzamiento
Netbeans 7.1.2	Mayo de 2012
Netbeans 7.0.1	01 de agosto de 2011
Netbeans 7.0	20 de abril de 2011
Netbeans 6.9.1	4 de agosto de 2010
Netbeans 6.9	15 de junio de 2010
Netbeans 6.8	10 de diciembre de 2009
Netbeans 6.7.1	27 de julio de 2009
Netbeans 6.7	29 de junio de 2009
Netbeans 6.5	25 de noviembre de 2008
Netbeans 6.1	28 de abril de 2008
Netbeans 6.0	3 de diciembre de 2007
Netbeans 5.5.1	24 de mayo de 2007
Netbeans 5.5	30 de octubre de 2006
Netbeans 5.0	Enero de 2006
Netbeans 4.1	Mayo de 2005
Netbeans 4.0	Diciembre de 2004
Netbeans 3.6	Abril de 2004
Netbeans 3.5	Junio de 2003

Tabla 2: Versiones de Neatbeans

3 Análisis

En este apartado se muestra el análisis realizado, que es una fase anterior del desarrollo del proyecto, en el que por medio de entrevistas con el Cliente, en este caso, el Tutor del proyecto, se analiza en qué consiste el producto final y cómo debe comportarse dependiendo de las necesidades de los usuarios. En él se muestran los requisitos que tiene que cumplir y los casos en los que los actores, en este caso solo los usuarios, pueden realizar.

3.1 Requisitos de usuario

En este apartado se detallan cada uno de los Requisitos de Usuario que son especificados dentro del sistema.

Para especificar cada requisito se ha seguido la siguiente plantilla en forma de tabla que ayuda a focaliza la información:

RUY-XX	
Descripción	
Necesidad	
Prioridad	
Estabilidad	
Origen	
Grado de Verificación	

Tabla 3: Diseño de tabla usada para los requisitos de Usuario

Siendo:

- **RUY XX:** es el identificador del requisito.
  - La Y puede tomar los siguientes valores:
    - C: para especificar que es un requisito de capacidad.
    - R: para especificar que es un requisito de restricción.
  - La XX es un número de dos cifras que sigue el orden en el que se encuentra.
- **Descripción:** es una breve descripción que detalla en qué consiste ese requisito.
- **Necesidad:** es el grado de necesidad que se tiene en que dicho requisito se encuentre implementado en el sistema. Pueden ser:
  - Esenciales: no se puede negociar dicho requisito, debe ser implementado en el sistema,
  - Deseable: es importante implementar el requisito pero no obligatorio.
  - Opcional: es irrelevante la implementación del requisito. Se podrá implementar pero no es obligatorio.

- **Prioridad**: indica la importancia del requisito en el proceso de diseño e implementación. Los valores que puede tomar este campo son:
    - Alta: el requisito debe ser diseñado y añadido al sistema en primer lugar.
    - Media: el requisito debe ser diseñado y añadido al sistema tras haber acabado con los requisitos de prioridad alta.
    - Baja: el requisito debe ser diseñado y añadido al sistema tras haber acabado con los requisitos de prioridad media.
- **Estabilidad**: indica si el requisito no variará durante todo el proceso de diseño e implementación del sistema. Los valores que puede tomar este campo son:
    - Alta: el requisito no se modificará ni borrará durante la vida del sistema.
    - Media: el requisito puede ser modificado o eliminado en cualquier fase del proyecto.
    - Baja: existe una alta probabilidad de que el requisito cambie o sea eliminado en cualquier fase del proyecto.
- **Origen**: indica la procedencia del requisito.
- **Grado de verificabilidad**: indica el grado de sencillez a la hora de comprobar si el requisito ha sido implementado de forma correcta. Los valores que puede tomar este campo son:
    - Alta: la comprobación del requisito es complicada.
    - Media: el requisito se puede comprobar en un tiempo razonable y de manera fácil.
    - Baja: el requisito se comprueba de una manera rápida y sencilla.

A continuación se presentan los Requisitos de Usuario que se han identificado, divididos en diferentes secciones, según su naturaleza.

3.1.1 Requisitos de Usuario de Capacidad

En este apartado se especifican los Requisitos de Usuario que se categoricen como capacidad del sistema.

RUC-01	
Descripción	Registrar al usuario la primera vez que utilice la aplicación
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Media

Tabla 4: RUC-01

RUC-02	
Descripción	Crear grupos de usuarios
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 5: RUC-02

RUC-03	
Descripción	Enviar localización a los usuarios elegidos
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 6: RUC-03

RUC-04	
Descripción	Captar la localización del usuario, sin necesidad de que el propio usuario la escriba, por medio de señales GPS, redes móviles o Wifi
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Media

Tabla 7: RUC-04

RUC-05	
Descripción	Recoger la localización como primera opción por medio de entrada manual de teclado
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 8: RUC-05

RUC-06	
Descripción	Dar la opción de elegir la hora de encuentro, aunque la hora por defecto proporcionada será la hora en la que se está creando el encuentro
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 9: RUC-06

RUC-07	
Descripción	Enviar una fotografía del sitio de encuentro. Este campo es opcional, la elección la toma el usuario
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Media

Tabla 10: RUC-07

RUC-08	
Descripción	El usuario puede elegir enviar a un contacto o un grupo de usuarios el encuentro que crea
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 11: RUC-08

RUC-09	
Descripción	Tener una miga de pan para el procedimiento de enviar localización
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 12: RUC-09

RUC-10	
Descripción	Tener la opción de volver atrás para reeditar la información cuando se está creando el encuentro
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 13: RUC-10

RUC-11	
Descripción	Notificar al usuario receptor cuando le llega una ubicación enviada por otro usuario
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 14: RUC-11

RUC-12	
Descripción	Disponer de una lista de encuentros creados y recibidos
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 15: RUC-12

RUC-13	
Descripción	Mostrar en una pantalla toda la información de un encuentro elegido
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 16: RUC-13

RUC-14	
Descripción	Mostrar un mapa con la localización exacta del encuentro
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 17: RUC-14

RUC-15	
Descripción	Tener un control de errores que evite que el usuario cometa fallos
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 18: RUC-15

RUC-16	
Descripción	El servidor tendrá que escuchar en todo momento las peticiones de los clientes
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 19: RUC-16

RUC-17	
Descripción	El servidor controlará los usuarios registrados
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 20: RUC-17

RUC-18	
Descripción	El servidor responderá a las peticiones hechas por los clientes
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 21: RUC-18

RUC-19	
Descripción	El servidor tendrá que mantener la seguridad de información de los usuarios en todo momento
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 22: RUC-19

3.1.2 Requisitos de Usuario de Restricción

En este apartado se especifican los Requisitos de Usuario que sean restricciones que tenga que cumplir el sistema.

RUR-01	
Descripción	Desarrollar la aplicación para que un cliente se comunique con un servidor y éste distribuya la información
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 23: RUR-01

RUR-02	
Descripción	Desarrollar en Android con versión mínima Éclair (v2.1) para la aplicación que se ejecute en el móvil
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 24: RUR-02

RUR-03	
Descripción	Desarrollar en Java la parte del Servidor
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 25: RUR-03

RUR-04	
Descripción	Base de Datos SQLite para el registro de usuarios en el sistema y ficheros en la memoria SD para la información de la aplicación móvil
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Baja

Tabla 26: RUR-04

RUR-05	
Descripción	Reducir al máximo los tiempos de respuestas, ya sean del servidor o de los clientes.
Necesidad	Deseable
Prioridad	Media
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 27: RUR-05

RUR-06	
Descripción	Tener una interfaz fácil de usar, intuitiva y fácil de recordar
Necesidad	Deseable
Prioridad	Media
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 28: RUR-06

RUR-07	
Descripción	Ser un programa escalable, puede que en un futuro se inserten nuevas funcionalidades
Necesidad	Deseable
Prioridad	Media
Estabilidad	Media
Origen	Cliente
Grado de Verificación	Alta

Tabla 29: RUR-07

RUR-08	
Descripción	Ser un programa robusto, en el que no haya fallos, ya que puede ocasionar pérdidas de dinero y dejar a la gente sin saber dónde está el usuario con el que se quiere encontrar
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 30: RUR-08

RUR-09	
Descripción	Tener seguridad en los datos que se almacenen para conservar la privacidad de los usuarios
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 31: RUR-09

RUR-10	
Descripción	Usar conexiones 3G o Wifi para comunicar los móviles con el Servidor
Necesidad	Alta
Prioridad	Alta
Estabilidad	Media
Origen	Cliente
Grado de Verificación	Media

Tabla 32: RUR-10

RUR-11	
Descripción	Usar señal GPS o de telefonía para coger ubicación en caso de no insertarla de forma manual por teclado
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Media

Tabla 33: RUR-11

RUR-12	
Descripción	Mostrar los mapas de Google Maps cuando se abra el mapa para ver la localizaión, ya que dan más usos adicionales como la opción de usar “Google Navegator” para que te guíe hasta dicho usuario
Necesidad	Deseable
Prioridad	Media
Estabilidad	Media
Origen	Cliente
Grado de Verificación	Baja

Tabla 34: RUR-12

RUR-13	
Descripción	Se tendrá que acceder a los contactos del móvil para conocer e identificar los números que se usen en el sistema
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 35: RUR-13

RUR-14	
Descripción	Se tendrán que usar las estructuras adecuadas para trabajar con los datos
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 36: RUR-14

RUR-15	
Descripción	Hasta que no haya información necesaria para identificar una ubicación y a quien se quiere mandar dicha información no se podrá mandar nada de la información
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 37: RUR-15

RUR-16	
Descripción	El sistema de avisos a los usuarios de un encuentro se realizarán por medio de notificaciones
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Media

Tabla 38: RUR-16

RUR-17	
Descripción	La documentación deberá seguir un estándar y deberá estar escrita de forma correcta tanto en escritura con en formato
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

Tabla 39: RUR-17

RUR-18	
Descripción	El sistema de comunicación será por sockets
Necesidad	Alta
Prioridad	Alta
Estabilidad	Alta
Origen	Cliente
Grado de Verificación	Alta

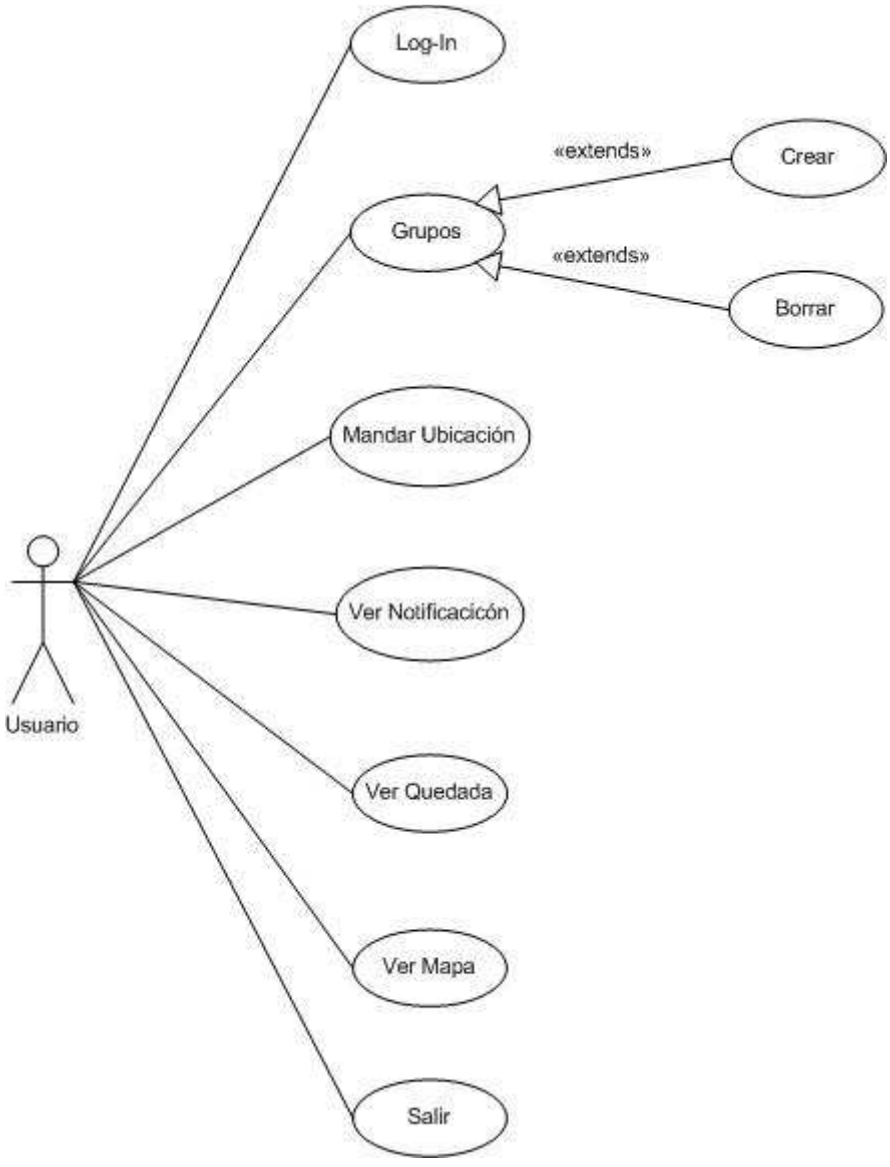
Tabla 40: RUR-18

3.2 Casos de uso

En este apartado se detallaran los Casos de Uso que los actores del sistema pueden desempeñar. Para aclarar el resultado obtenido del diagrama en este proyecto los actores solo serán usuarios del sistema.

3.2.1 Diagrama de Casos de Uso

En este apartado se muestra en un diagrama los Casos de Uso que los actores pueden realizar:



Ilustracion 3. Diagrama de Casos de Uso

3.2.2 Especificación de los Casos de Uso

En este apartado se detallan los Casos de Uso de forma clara y concisa. Para ello se sigue la siguiente tabla:

CU-XX	
Nombre	
Actor	
Objetivo	
Pre-condición	
Descripción	
Post-condición	

Tabla 41: Modelo de tabla de Caso de Uso

Siendo:

- **CU XX:** El identificador del Caso de Uso.
- **Nombre:** es el nombre que detalla la acción que se realiza con dicho Caso de Uso.
- **Actor:** es el rol o roles que intervienen en el Caso de Uso.
- **Objetivo:** es el propósito que se quiere alcanzar con el Casi de Uso.
- **Descripción:** son los pasos detallados de lo que tiene que hacer el Caso de Uso.
- **Pre-condición:** son las condiciones que tiene que cumplir para que se active el Caso de uso.
- **Post-condición:** son las condiciones en las que se tiene que quedar el sistema una vez se haya culminado el Caso de Uso.

A continuación se especifican los Casos de Uso:

CU 01	
Nombre	Hacer log-in
Actor	Usuario
Objetivo	El usuario quede registrado en el sistema
Pre-condición	• Arrancar el sistema • No haya ya un registro del usuario en el móvil
Descripción	1 El usuario arranca el sistema 2 Mete el número de teléfono al que quiere que se le asocie 3 Inserta la dirección de correo mail 4 Pulsa el botón “ENVIAR”
Post-condición	• Queda un registro del usuario en el servidor • Se guarda un fichero con la clave de acceso al sistema en la memoria SD • Se muestra la pantalla principal

Tabla 42: CU 01: Hacer log-in

CU 02	
Nombre	Crear grupo
Actor	Usuario
Objetivo	Crear un nuevo grupo de usuarios para mandar una ubicación a varios usuarios.
Pre-condición	• Sistema arrancado. • Estar registrado en el sistema.
Descripción	1 Ir a la pantalla “Nuevo grupo” para ello se puede ir: a. Desde la pantalla principal pulsando el botón “Grupos >> Crear”. b. Desde cualquier pantalla dando al botón de Menú del móvil y seleccionando la opción “Crear un Grupo”. 2 Insertar un nombre de grupo. 3 Elegir los usuarios, para ello: a. Pulsar el botón añadir usuario b. Seleccionar usuario c. Seguir haciéndolo los pasos a y b tantas veces como se quiera hasta un máximo de 10 usuarios insertados. 4 Pulsar el botón “CREAR”.
Post-condición	• El nuevo grupo se queda guardado en el fichero grupos.txt en la posición que le corresponde alfanuméricamente.

Tabla 43: CU 02: Crear grupo

CU 03	
Nombre	Borrar grupo
Actor	Usuario
Objetivo	Borrar un grupo ya existente que no se vaya a usar
Pre-condición	• Sistema arrancado. • Estar registrado en el sistema. • Tener al menos un grupo ya creado
Descripción	1 Ir a la pantalla “Opciones de Grupos” para ello se puede ir: a. Desde la pantalla principal pulsando el botón “Grupos”. 2 Seleccionar la opción “Borrar”. 3 Seleccionar el grupo a Borrar
Post-condición	• El grupo se borra de la aplicación del móvil del usuario sin quedar ninguna constancia en el fichero grupos.txt.

Tabla 44: CU 03: Borrar grupo

CU 04	
Nombre	Mandar ubicación
Actor	Usuario
Objetivo	Mandar la ubicación a los usuarios que se desee para que sepan donde te encuentras.
Pre-condición	• Sistema arrancado. • Estar registrado en el sistema.
Descripción	1 Ir a la pantalla “Enviar Localización” para ello se puede ir: a. Desde la pantalla principal pulsando el botón “Enviar Localización”. b. Desde cualquier pantalla, que no sea ninguna de las de mandar ubicación, dando al botón de Menú del móvil y seleccionando la opción “Enviar Localización”. 2 Elegir mandar la ubicación a un usuario o a un grupo de usuarios 3 Seleccionar el usuario o el grupo 4 Insertar la dirección o pulsar para que te cojan la ubicación por GPS o redes de telefonía móvil 5 Elegir hora 6 Como forma opcional hacer una foto a algo característico 7 Pulsar el botón “ENVIAR”
Post-condición	• Ubicación enviada al servidor • Se muestra la pantalla principal

Tabla 45: CU 04: Mandar ubicación

CU 05	
Nombre	Ver notificación
Actor	Usuario
Objetivo	Cuando te llega la ubicación de otro usuario se te avisa por una notificación. Se pretende que se vea la información de la ubicación por medio de la notificación
Pre-condición	• Estar registrado en el sistema • Que te hayan mandado una ubicación y te haya llegado correctamente
Descripción	1 Bajar el panel de las notificaciones
Post-condición	• Desaparece la notificación

Tabla 46: CU 05: Ver notificación

CU 06	
Nombre	Ver encuentro
Actor	Usuario
Objetivo	Ver toda la información del encuentro, inclusive la foto.
Pre-condición	• Que haya llegado una notificación porque otro usuario haya mandado su ubicación
Descripción	1 Hay dos modos de ver el encuentro: a. Tener la aplicación arrancada i. Ir a la pantalla principal ii. Pulsar el botón “Mis encuentros” iii. Seleccionar el encuentro que quieres ver b. No tener la aplicación arrancada i. Ir a las notificaciones ii. Pulsar encima de la notificación de “WherAmI”
Post-condición	• Se muestra la pantalla del encuentro con toda la información

Tabla 47: CU 06: Ver encuentro

CU 07	
Nombre	Ver mapa
Actor	Usuario
Objetivo	Ver un mapa con la ubicación señalada para verlo de forma visual y poder saber mejor dónde está.
Pre-condición	• Estar en la pantalla del encuentro que se quiere consultar
Descripción	1 Pulsar el botón “Ver Mapa”
Post-condición	• Abrir un mapa de Google Maps con la ubicación señalada

Tabla 48: CU 07: Ver mapa

CU 08	
Nombre	Salir
Actor	Usuario
Objetivo	Salir del sistema
Pre-condición	• Estar registrado • Tener el sistema arrancado
Descripción	1 Hay dos formas de salir: a. Si se está en la pantalla principal darle al botón de “Atrás” del móvil b. En cualquier pantalla pulsar el botón “Menú” del móvil y seleccionar “Salir”
Post-condición	• El sistema dejará de funcionar y saldrá de él

Tabla 49: CU 08: Salir

3.3 Requisitos Software

En este apartado se detallan cada uno de los Requisitos de Usuario que son especificados dentro del sistema.

Para especificar cada requisito se ha seguido la siguiente plantilla en forma de tabla que ayuda a focaliza la información:

RS[N]F[-YY]-XX	
Descripción	
Necesidad	
Prioridad	
Estabilidad	
Origen	
Grado de Verificación	

Tabla 50: Diseño de tabla usada para los requisitos de Usuario

Siendo:

- **RS[N]F[-YY]- XX:** es el identificador del requisito.
  - RSF será los Requisitos Software Funcionales y los RSNF serán los Requisitos Software No Funcionales.
  - La YY puede tomar los siguientes valores:
    - IN: para especificar que es un requisito software no funcional de interfaz.
    - SE: para especificar que es un requisito software no funcional de seguridad
    - DE: para especificar que es un requisito software no funcional de desarrollo
    - BD: para especificar que es un requisito software no funcional de base de datos
    - OP: para especificar que es un requisito software no funcional de operación
    - DO: para especificar que es un requisito software no funcional de documentación
  - La XX es un número de dos cifras que sigue el orden en el que se encuentra.
- **Descripción:** es una breve descripción que detalla en qué consiste ese requisito.
- **Necesidad:** es el grado de necesidad que se tiene en que dicho requisito se encuentre implementado en el sistema. Pueden ser:

  - Esenciales: no se puede negociar dicho requisito, debe ser implementado en el sistema,
  - Deseable: es importante implementar el requisito pero no obligatorio.
  - Opcional: es irrelevante la implementación del requisito. Se podrá implementar pero no es obligatorio.
- **Prioridad**: indica la importancia del requisito en el proceso de diseño e implementación. Los valores que puede tomar este campo son:
  - Alta: el requisito debe ser diseñado y añadido al sistema en primer lugar.
  - Media: el requisito debe ser diseñado y añadido al sistema tras haber acabado con los requisitos de prioridad alta.
  - Baja: el requisito debe ser diseñado y añadido al sistema tras haber acabado con los requisitos de prioridad media.
- **Estabilidad**: indica si el requisito no variará durante todo el proceso de diseño e implementación del sistema. Los valores que puede tomar este campo son:
  - Alta: el requisito no será modificado ni borrado durante toda la vida del sistema.
  - Media: el requisito puede ser modificado o eliminado en cualquier fase del proyecto.
  - Baja: existe una alta probabilidad de que el requisito cambie o sea eliminado en cualquier fase del proyecto.
- **Origen**: indica la procedencia del requisito. En este caso muestra la trazabilidad con respecto a los requisitos de usuario.
- **Grado de verificabilidad**: indica el grado de sencillez a la hora de comprobar si el requisito ha sido implementado de forma correcta. Los valores que puede tomar este campo son:
  - Alta: la comprobación del requisito es complicada.
  - Media: el requisito se puede comprobar en un tiempo razonable y de manera fácil.
  - Baja: el requisito se comprueba de una manera rápida y sencilla.

A continuación se presentan los Requisitos de Usuario que se han identificado, divididos en diferentes secciones, según su naturaleza.

3.3.1 Requisitos Software Funcionales

En este apartado se especifican los Requisitos de Usuario que sean restricciones que tenga que cumplir el sistema.

RSF-01	
Descripción	Registrar a los usuarios que quieran usar la aplicación para que no se le mande a alguien que no tenga la aplicación instalada y por tanto no pueda recibir las notificaciones
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-01,
Grado de Verificación	Baja

Tabla 51: RSF-01

RSF-02	
Descripción	Crear grupos de usuarios para permitir enviar la ubicación a varias personas de forma rápida
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-02
Grado de Verificación	Media

Tabla 52: RSF-02

RSF-03	
Descripción	Borrar grupos que ya no se vayan a usar para no ocupar espacio y poder encontrar los grupos deseados en menos tiempo
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-02
Grado de Verificación	Baja

Tabla 53: RSF-03

RSF-04	
Descripción	Para añadir usuarios a un grupo se seleccionará de una lista de usuarios registrados
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-02
Grado de Verificación	Media

Tabla 54: RSF-04

RSF-05	
Descripción	El sistema debe mandar la ubicación deseada a los usuarios elegidos
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-03
Grado de Verificación	Alta

Tabla 55: RSF-05

RSF-06	
Descripción	Podrá insertarse la ubicación de dos maneras posibles dando al usuario la opción de elegir el modo: • Inserción manual • Inserción automática
Necesidad	Esencial
Prioridad	Media
Estabilidad	Alta
Origen	RUC-04, RUC-05
Grado de Verificación	Baja

Tabla 56: RSF-06

RSF-07	
Descripción	El modo de inserción de ubicación por defecto será el de insertarlo por teclado
Necesidad	Esencial
Prioridad	Media
Estabilidad	Media
Origen	RUC-04
Grado de Verificación	Baja

Tabla 57: RSF-07

RSF-08	
Descripción	El modo de inserción de ubicación alternativo será el de insertarlo por medio de GPS o red de telefonía móvil
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Media
Origen	RUC-05
Grado de Verificación	Alta

Tabla 58: RSF-08

RSF-09	
Descripción	El botón que permite el cambio de inserción de la ubicación estará desactivado cuando el modo sea inserción por teclado y activado cuando sea inserción por GPS o red de telefonía móvil
Necesidad	Esencial
Prioridad	Baja
Estabilidad	Media
Origen	RUC-04, RUC-05
Grado de Verificación	Baja

Tabla 59: RSF-09

RSF-10	
Descripción	El usuario podrá seleccionar una hora de encuentro diferente a la actual, aunque el valor predeterminado será la hora en la que se está creando
Necesidad	Deseable
Prioridad	Media
Estabilidad	Alta
Origen	RUC-06
Grado de Verificación	Baja

Tabla 60: RSF-10

RSF-11	
Descripción	El usuario podrá adjuntar como opcional una foto del lugar donde se ha quedado
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-07
Grado de Verificación	Baja

Tabla 61: RSF-11

RSF-12	
Descripción	La foto se tomará de la cámara y no dejará la opción de cogerla de una galería para evitar el uso indebido de la aplicación
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-07
Grado de Verificación	Baja

Tabla 62: RSF-12

RSF-13	
Descripción	Para coger la foto se tendrá que pulsar un botón como parte de esa opcionalidad que se le da al usuario
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-07
Grado de Verificación	Baja

Tabla 63: RSF-13

RSF-14	
Descripción	El sistema que coge las fotos será el propio servicio de fotos interno del móvil
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-07
Grado de Verificación	Baja

Tabla 64: RSF-14

RSF-15	
Descripción	La información que se manda es: • Quién manda el encuentro • A quién se manda el encuentro • La dirección donde se ha quedado • La hora que se ha quedado • Una foto, en caso de adjuntarla, del sitio
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-04, RUC-05, RUC-06, RUC-07
Grado de Verificación	Alta

Tabla 65: RSF-15

RSF-16	
Descripción	La información mínima para mandar a otros usuarios será: • Quién lo manda • A quién se lo manda • dirección • hora del encuentro
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-04, RUC-05, RUC-06, RUC-07
Grado de Verificación	Alta

Tabla 66: RSF-16

RSF-17	
Descripción	El usuario puede elegir entre mandárselo a un usuario particularmente o a un grupo de usuarios que haya creado previamente
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-08
Grado de Verificación	Baja

Tabla 67: RSF-17

RSF-18	
Descripción	En las pantallas de edición de ubicación se usará una miga de pan para que el usuario sepa la información que se le va a pedir
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-04, RUC-05, RUC-06, RUC-07, RUC-08, RUC-09
Grado de Verificación	Media

Tabla 68: RSF-18

RSF-19	
Descripción	En la miga de pan se usará un sistema visual que represente dónde está el usuario y lo que le queda por rellenar. Para ello proporcionará la información mínima de las pantallas de información señalando de forma especial en la que se encuentra actualmente
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-09
Grado de Verificación	Media

Tabla 69: RSF-19

RSF-20	
Descripción	Se podrá navegar entre las pantallas de información por si se quiere reeditar alguna de la información requerida
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-04, RUC-05, RUC-06, RUC-07, RUC-08, RUC-10
Grado de Verificación	Media

Tabla 70: RSF-20

RSF-21	
Descripción	En el caso en el que alguna pantalla no pueda navegar a otra porque se requiere antes algo de información, el elemento de navegación aparecerá deshabilitado
Necesidad	Esencial
Prioridad	Media
Estabilidad	Alta
Origen	RUC-10
Grado de Verificación	Media

Tabla 71: RSF-21

RSF-22	
Descripción	En todas las pantallas de información aparecerá de forma inmóvil en botón para enviar, estando habilitado cuando ya disponga de toda la información mínima para poder mandársela al usuario o al grupo de usuarios.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-04, RUC-05, RUC-06, RUC-07, RUC-08
Grado de Verificación	Media

Tabla 72: RSF-22

RSF-23	
Descripción	Cuando llegue una ubicación de otro usuario se mostrará una notificación
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-11
Grado de Verificación	Media

Tabla 73: RSF-23

RSF-24	
Descripción	Todos los usuarios tendrán una lista de encuentros independientemente de si son enviadas o recibidas
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-12
Grado de Verificación	Baja

Tabla 74: RSF-24

RSF-25	
Descripción	Cuando se seleccione un encuentro para visualizar toda la información aparecerá una pantalla en la que se muestre toda la información y tendrá la opción de visualizar la ubicación en un mapa.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-13
Grado de Verificación	Media

Tabla 75: RSF-25

RSF-26	
Descripción	Siempre aparecerá un botón que lleve a ver el mapa.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-13, RUC-14
Grado de Verificación	Baja

Tabla 76: RSF-26

RSF-27	
Descripción	El mapa será la de Google Maps con la ubicación señalada.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-14
Grado de Verificación	Media

Tabla 77: RSF-27

RSF-28	
Descripción	Cuando se inserta algo incorrectamente tiene que notificar el error
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-15
Grado de Verificación	Alta

Tabla 78: RSF-28

RSF-29	
Descripción	Deshabilitar botones cuando conllevan acciones que no se pueden realizar para evitar errores del usuario
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-15
Grado de Verificación	Alta

Tabla 79: RSF-29

RSF-30	
Descripción	No añadir al mismo usuario a un grupo
Necesidad	Esencial
Prioridad	Media
Estabilidad	Alta
Origen	RUC-02
Grado de Verificación	Baja

Tabla 80: RSF-30

RSF-31	
Descripción	El receptor se tendrá que conectar cada cierto tiempo al servidor para pedir las nuevas reuniones que hayan sido enviadas para él como receptor
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-11, RUC-16
Grado de Verificación	Alta

Tabla 81: RSF-31

RSF-32	
Descripción	El servidor escuchará en todo momento nuevas peticiones de conexión de usuarios, otorgando para ello un puerto abierto a la escucha
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-16
Grado de Verificación	Alta

Tabla 82: RSF-32

RSF-33	
Descripción	El servidor solo operará con usuarios registrados excepto en la parte de registro
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-17
Grado de Verificación	Alta

Tabla 83: RSF-33

RSF-34	
Descripción	El servidor consultará a la base de datos para responder a los usuarios que le están haciendo peticiones con información totalmente actualizada y lo más rápido posible
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-18
Grado de Verificación	Alta

Tabla 84: RSF-34

RSF-35	
Descripción	El servidor borrará aquella información que ya haya sido usada y que no vaya a tener más uso por ningún cliente
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUC-19
Grado de Verificación	Alta

Tabla 85: *RSF-35*

3.3.2 Requisitos Software No Funcionales

En este apartado se describirán todos los requisitos Software que propiamente no tengan nada que ver con la funcionalidad como tal, sino de otras partes del sistema como interface, bases de datos, formas de operar, la seguridad entre otras.

A continuación se detallan cada uno de estos requisitos.

3.3.2.1 Requisitos de Interfaz

En este apartado se detallaran los requisitos Software no funcionales que tengan relación directa con las interfaces que se usarán en el sistema.

RSNF-IN-01	
Descripción	En la pantalla de registro solo se pedirá el teléfono por medio de un texto editables con formato de números
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Baja

Tabla 86: RSNF- IN-01

RSNF-IN-02	
Descripción	En la cabecera de la pantalla de la aplicación móvil, mostrará en qué pantalla te encuentras
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Alta

Tabla 87: RSNF- IN-02

RSNF-IN-03	
Descripción	Las cabeceras tendrán el siguiente estilo: color de letra blanco con el fondo azul marino
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Alta

Tabla 88: RSNF- IN-03

RSNF-IN-04	
Descripción	Todas las pantallas tendrán el mismo estilo en la pantalla de información, de forma que focalice de forma directa lo esencial de cada pantalla
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Alta

Tabla 89: RSNF- IN-04

RSNF-IN-05	
Descripción	El estilo de las pantallas será: Fondo azul claro con letras negras y el borde de la pantalla será gris con esquinas redondeadas
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Media
Origen	RUR-06
Grado de Verificación	Alta

Tabla 90: RSNF- IN-05

RSNF-IN-06	
Descripción	En las pantallas que piden información para mandar una ubicación tendrá en la parte alta de la pantalla un espacio en negro con las pantallas de petición de información en negro y letra gris y la pantalla actual en gris con letra negra, es decir, con los colores invertidos
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Media
Origen	RUR-06
Grado de Verificación	Alta

Tabla 91: RSNF- IN-06

RSNF-IN-07	
Descripción	En las pantallas que piden información para mandar una ubicación aparecerá en la parte baja de la pantalla un espacio en negro con los botones de navegación y el de enviar.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Alta

Tabla 92: RSNF- IN-07

RSNF-IN-08	
Descripción	Cuando un botón no debe ser usado en las pantallas de petición de información aparecerá deshabilitado, para que el usuario sepa que no puede usarlo.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Alta

Tabla 93: RSNF- IN-08

RSNF-IN-09	
Descripción	Cuando aparezca una lista de usuarios o de grupos el fondo será gris oscuro con letras blancas, borde blanco y esquinas redondeadas
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Media
Origen	RUR-06
Grado de Verificación	Alta

Tabla 94: RSNF- IN-09

RSNF-IN-10	
Descripción	Todos los botones que aparezcan en la aplicación deben ser lo más grandes posibles para facilitar la visión a personas menos capacitadas visualmente
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Alta

Tabla 95: RSNF- IN-10

RSNF-IN-11	
Descripción	El servidor no dispondrá de ninguna interfaz gráfica
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Alta

Tabla 96: RSNF- IN-11

RSNF-IN-12	
Descripción	La información de la notificación será: • Quien lo manda • La dirección • La hora
Necesidad	Esencial
Prioridad	Alta
Estabilidad	RUR-06
Origen	Alta
Grado de Verificación	Alta

Tabla 97: RSNF- IN-12

RSNF-IN-13	
Descripción	La información de cada encuentro dentro de la lista será la información mínima y el usuario que ha mandado la ubicación
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Alta

Tabla 98: RSNF- IN-13

RSNF-IN-14	
Descripción	Para elegir si mandar a un contacto o a un grupo de usuarios habrá dos botones y dependiendo de cual se pulse irá a una lista u otra
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Baja

Tabla 99: RSNF- IN-14

RSNF-IN-15	
Descripción	Para insertar la ubicación se tendrá una caja editable
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Baja

Tabla 100: RSNF- IN-15

RSNF-IN-16	
Descripción	Para elegir si se coge ubicación por medio tanto de GPS o red de telefonía móvil o se inserta de forma manual habrá un “Toggle” debajo del editable
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06, RUR-11
Grado de Verificación	Baja

Tabla 101: RSNF- IN-16

RSNF-IN-17	
Descripción	Para cambiar la hora habrá una rueda en la que las horas y los minutos serán independientes y el formato será 24h
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Baja

Tabla 102: RSNF- IN-17

RSNF-IN-18	
Descripción	No se define interfaz para la toma de fotos porque la aplicación coge el propio servicio de cámara del móvil, por lo que no se podrá diseñar nada, ya que depende de cada móvil
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Baja

Tabla 103: RSNF- IN-18

RSNF-IN-19	
Descripción	Cuando se crea un grupo el nombre del grupo será un editable
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06
Grado de Verificación	Baja

Tabla 104: RSNF- IN-19

RSNF-IN-20	
Descripción	Cuando se crea un grupo para ir a la lista de contactos habrá un botón con una imagen significativa de insertar usuario
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Media
Origen	RUR-06
Grado de Verificación	Baja

Tabla 105: RSNF- IN-20

RSNF-IN-21	
Descripción	Si el emisor es el propio usuario de su dispositivo aparecerá como emisor “YO”
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06, RUR-13, RUR-16
Grado de Verificación	Baja

Tabla 106: RSNF- IN-21

RSNF-IN-22	
Descripción	Si el receptor es el propio usuario de su dispositivo aparecerá como receptor “YO”
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06, RUR-13, RUR-16
Grado de Verificación	Baja

Tabla 107: RSNF-IN-22

RSNF-IN-23	
Descripción	Si en el campo del receptor y/o emisor el número es distinto al propio usuario del dispositivo aparecerá el nombre de quien lo manda en caso de que le tenga como amigo en mi agenda, en caso contrario aparecerá el número de teléfono
Necesidad	Deseable
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-06, RUR-13, RUR-16
Grado de Verificación	Media

Tabla 108: RSNF-IN-23

3.3.2.2 Requisitos de Seguridad

En este apartado se detallan los requisitos de Software que tengan una relación directa con la seguridad del sistema.

RSNF-SE-01	
Descripción	Solo se almacenará en el servidor el número de teléfono del usuario para poder saber quién manda la ubicación y a quién mandársela.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-01, RUR-04, RUR-09
Grado de Verificación	Alta

Tabla 109: RSNF-SE-01

RSNF-SE-02	
Descripción	La información se irá borrando del servidor según se vaya usando para no comprometer a los usuarios
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-01, RUR-09
Grado de Verificación	Alta

Tabla 110: RSNF-SE-02

3.3.2.3 Requisitos de Desarrollo

En este apartado se detallan los requisitos Software que tengan una relación directa con el desarrollo que se utilizará en el sistema.

RSNF-DE-01	
Descripción	Se usará la plataforma Android para desarrollar la parte del Cliente en el terminal móvil
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-01, RUR-02
Grado de Verificación	Alta

Tabla 111: RSNF-DE-01

RSNF-DE-02	
Descripción	Se usará la plataforma Java para desarrollar la parte del Servidor que planifica las rutas
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-01, RUR-03
Grado de Verificación	Alta

Tabla 112: RSNF-DE-02

RSNF-DE-03	
Descripción	Todo el código deberá estar bien documentado
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-08
Grado de Verificación	Alta

Tabla 113: RSNF-DE-03

3.3.2.4 Requisitos de Base de Datos

En este apartado se detallan los requisitos Software relacionados con las Bases de Datos y los ficheros de almacenaje, ya sea en la parte del cliente o del servidor.

RSNF-BD-01	
Descripción	El servidor guardará en una base de datos SQLite todos los usuarios registrados en una tabla
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04
Grado de Verificación	Alta

Tabla 114: RSNF-BD-01

RSNF-BD-02	
Descripción	Se utilizará una base de datos SQLite en el servidor con varias tablas relacionadas para conseguir la integridad del sistema
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04
Grado de Verificación	Alta

Tabla 115: RSNF-BD-02

RSNF-BD-03	
Descripción	Los grupos de usuarios se guardaran en la tarjeta SD para no saturar el móvil y que no se quede sin memoria y se guardarán en un único fichero, al igual que el resto de información útil.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04
Grado de Verificación	Alta

Tabla 116: RSNF-BD-03

3.3.2.5 Requisitos de Operación

En este apartado se detallan los requisitos Software que tengan una relación directa con el modo de operación que debe cumplirse dentro del sistema.

RSNF-OP-01	
Descripción	Para añadir un nuevo usuario a un grupo se hará por medio de una lista con todos los contactos del usuario que estén registrados en el sistema
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-13, RUR-14
Grado de Verificación	Alta

Tabla 117: RSNF-OP-01

RSNF-OP-02	
Descripción	Para elegir los contactos a los que se les va a mandar la ubicación se hará por medio de listas, ya sea de contactos o de grupo. La opción la elegirá el usuario
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-14
Grado de Verificación	Alta

Tabla 118: RSNF-OP-02

RSNF-OP-03	
Descripción	No se podrá mandar información hasta que no tenga toda la información mínima, es decir, hasta que no se encuentre en la pantalla de toma de foto, en la que se podrá mandar sin haber sacado ninguna foto.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-15
Grado de Verificación	Alta

Tabla 119: RSNF-OP-03

RSNF-OP-04	
Descripción	La información se la mandará al servidor y luego el servidor se lo mandará a cada uno de los usuarios escogidos según se lo vayan solicitando
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-01
Grado de Verificación	Alta

Tabla 120: RSNF-OP-04

RSNF-OP-05	
Descripción	Cuando el usuario reciba un encuentro le saltará una notificación con los avisos pertinentes del dispositivo
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-16
Grado de Verificación	Alta

Tabla 121: RSNF-OP-05

RSNF-OP-06	
Descripción	En el campo de ubicación deberá tener una dirección para pasar a la siguiente pantalla
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-15
Grado de Verificación	Alta

Tabla 122: RSNF-OP-06

RSNF-OP-07	
Descripción	La información que se vaya recogiendo en cada pantalla, se la irá mandando a la siguiente para tener al final toda la información a mandar
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-08
Grado de Verificación	Alta

Tabla 123: RSNF-OP-07

RSNF-OP-08	
Descripción	Si se navega hacia atrás en una pantalla se le volverá a pasar la información que le hubiera pasado la pantalla previa a ella
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-08
Grado de Verificación	Alta

Tabla 124: RSNF-OP-08

RSNF-OP-09	
Descripción	La información de la foto se mandará en una array de bytes
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-14
Grado de Verificación	Alta

Tabla 125: RSNF-OP-09

RSNF-OP-10	
Descripción	Cuando se está en la petición de información, si se le da al botón atrás del dispositivo móvil volverá a la pantalla principal. Solo sirve la navegación por medio de los botones de navegación
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-08
Grado de Verificación	Alta

Tabla 126: RSNF-OP-10

RSNF-OP-11	
Descripción	Cuando se selecciona un encuentro se abrirá una pantalla con toda la información del encuentro. La información se cargará del fichero de encuentros que esté en la memoria SD del dispositivo.
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04, RUR-08
Grado de Verificación	Alta

Tabla 127: RSNF-OP-11

RSNF-OP-12	
Descripción	A la semana de que se haya producido el encuentro, esta se borrará de forma automática
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-08
Grado de Verificación	Alta

Tabla 128: RSNF-OP-12

RSNF-OP-13	
Descripción	Cuando se crea un grupo la información se pasará a un fichero situado en la memoria SD
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04
Grado de Verificación	Alta

Tabla 129: RSNF-OP-13

RSNF-OP-14	
Descripción	Para mostrar los contactos se hará una consulta al propio dispositivo de tal forma que actualizará la lista con la información sacada de la consulta
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04, RUR-13, RUR-14
Grado de Verificación	Alta

Tabla 130: RSNF-OP-14

RSNF-OP-15	
Descripción	Para mostrar los grupos se hará una consulta al fichero de grupos de tal forma que actualizará la lista con la información sacada de la consulta
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04, RUR-14
Grado de Verificación	Alta

Tabla 131: RSNF-OP-15

RSNF-OP-16	
Descripción	Cuando llega la información del servidor ésta se guardará en un fichero de encuentros
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04
Grado de Verificación	Alta

Tabla 132: RSNF-OP-16

RSNF-OP-17	
Descripción	Siempre que se vaya a usar la memoria SD se comprobará que esta está montada
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-04
Grado de Verificación	Alta

Tabla 133: RSNF-OP-17

RSNF-OP-18	
Descripción	El sistema de comunicación serán sockets en el que el servidor dará un puerto de escucha
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-18
Grado de Verificación	Alta

Tabla 134: RSNF-OP-18

RSNF-OP-19	
Descripción	Siempre la comunicación la iniciará el cliente
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-18
Grado de Verificación	Alta

Tabla 135: RSNF-OP-19

3.3.2.6 Requisitos de Documentación

En este apartado se detallan los requisitos Software que tengan una relación directa con la documentación del proyecto.

RSNF-DO-01	
Descripción	Todo el proyecto seguirá el estándar ESA Lite
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-17
Grado de Verificación	Alta

Tabla 136: RSNF-DO-01

RSNF-DO-02	
Descripción	Se seguirá una plantilla de Word para asegurar el formato concordante en todo momento
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-17
Grado de Verificación	Baja

Tabla 137: RSNF-DO-02

RSNF-DO-03	
Descripción	El idioma empleado en la documentación del proyecto será el Español
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-07, RUR-17
Grado de Verificación	Alta

Tabla 138: RSNF-DO-03

RSNF-DO-04	
Descripción	El idioma empleado en la implementación de la aplicación será el Inglés
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-07, RUR-17
Grado de Verificación	Alta

Tabla 139: RSNF-DO-04

RSNF-DO-05	
Descripción	El idioma de las pantallas de la aplicación será el Español
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-07, RUR-17
Grado de Verificación	Media

Tabla 140: RSNF-DO-05

RSNF-DO-06	
Descripción	Los diagramas que se muestren seguirán el estándar UML
Necesidad	Esencial
Prioridad	Alta
Estabilidad	Alta
Origen	RUR-17
Grado de Verificación	Alta

Tabla 141: RSNF-DO-06

3.3.3 Trazabilidad Requisitos Usuario – Requisito de Software (RU-RS)

En este apartado se mostrará la trazabilidad que existe entre los requisitos de usuario y los requisitos Software. Para poder organizar mejor la trazabilidad se separará la trazabilidad de los Requisitos de Usuario de Capacidad (RUC) con los Requisitos Software Funcionales (RSF) de la de los Requisitos de Usuario de Restricción (RUR) con los Requisitos Software No Funcionales (RSNF).

3.3.3.1 Trazabilidad RUC-RSF

En este apartado se muestra, por medio de una tabla, la trazabilidad existente entre los requisitos de usuario con los Requisitos Software Funcionales. Claramente los Requisitos Software Funcionales se relacionan de forma directa con los Requisitos de Usuario de Capacidad. A continuación se muestra dicha trazabilidad:

RSF \ RUC	RUC-01	RUC-02	RUC-03	RUC-04	RUC-05	RUC-06	RUC-07	RUC-08	RUC-09	RUC-10	RUC-11	RUC-12	RUC-13	RUC-14	RUC-15	RUC-16	RUC-17	RUC-18	RUC-19
RSF-01	X																		
RSF-02		X																	
RSF-03		X																	
RSF-04		X																	
RSF-05			X																
RSF-06				X	X														
RSF-07				X															
RSF-08					X														
RSF-09				X	X														
RSF-10						X													
RSF-11							X												
RSF-12							X												
RSF-13							X												
RSF-14							X												
RSF-15				X	X	X	X												
RSF-16				X	X	X	X												
RSF-17								X											
RSF-18				X	X	X	X	X	X										

RSF \ RUC	RUC-01	RUC-02	RUC-03	RUC-04	RUC-05	RUC-06	RUC-07	RUC-08	RUC-09	RUC-10	RUC-11	RUC-12	RUC-13	RUC-14	RUC-15	RUC-16	RUC-17	RUC-18	RUC-19
RSF-19									X										
RSF-20				X	X	X	X	X		X									
RSF-21										X									
RSF-22				X	X	X	X	X											
RSF-23											X								
RSF-24												X							
RSF-25													X						
RSF-26												X	X						
RSF-27														X					
RSF-28															X				
RSF-29															X				
RSF-30																			
RSF-31											X					X			
RSF-32																X			
RSF-33																	X		
RSF-34																		X	
RSF-35																			X

Tabla 142: Trazabilidad de RUC – RSF

3.3.3.2 Trazabilidad RUR-RSNF

En este apartado se muestra, por medio de una tabla, la trazabilidad existente entre los requisitos de usuario con los Requisitos Software No Funcionales. Claramente los Requisitos Software No Funcionales se relacionan de forma directa con los Requisitos de Usuario de Restricción. A continuación se muestra dicha trazabilidad:

RSNF \ RUR	RUR-01	RUR-02	RUR-03	RUR-04	RUR-05	RUR-06	RUR-07	RUR-08	RUR-09	RUR-10	RUR-11	RUR-12	RUR-13	RUR-14	RUR-15	RUR-16	RUR-17	RUR-18
RSNF-IN-01						X												
RSNF-IN-02						X												
RSNF-IN-03						X												
RSNF-IN-04						X												
RSNF-IN-05						X												
RSNF-IN-06						X												
RSNF-IN-07						X												
RSNF-IN-08						X												
RSNF-IN-09						X												
RSNF-IN-10						X												
RSNF-IN-11						X												
RSNF-IN-12						X												
RSNF-IN-13						X												
RSNF-IN-14						X												
RSNF-IN-15						X												
RSNF-IN-16						X					X							
RSNF-IN-17						X												
RSNF-IN-18						X												
RSNF-IN-19						X												
RSNF-IN-20						X												
RSNF-IN-21						X					X					X		
RSNF-IN-22						X					X					X		

RSNF \ RUR	RUR-01	RUR-02	RUR-03	RUR-04	RUR-05	RUR-06	RUR-07	RUR-08	RUR-09	RUR-10	RUR-11	RUR-12	RUR-13	RUR-14	RUR-15	RUR-16	RUR-17	RUR-18
RSNF-IN-23						X							X			X		
RSNF-SE-01	X			X					X									
RSNF-SE-02									X									
RSNF-DE-01	X	X																
RSNF-DE-02	X		X															
RSNF-DE-03								X										
RSNF-BD-01				X														
RSNF-BD-02				X														
RSNF-BD-03				X														
RSNF-BD-04				X														
RSNF-OP-01													X	X				
RSNF-OP-02														X				
RSNF-OP-03															X			
RSNF-OP-04	X																	
RSNF-OP-05																X		
RSNF-OP-06															X			
RSNF-OP-07								X										
RSNF-OP-08								X										
RSNF-OP-09														X				
RSNF-OP-10								X										
RSNF-OP-11				X				X										
RSNF-OP-12								X										
RSNF-OP-13				X														
RSNF-OP-14				X									X	X				
RSNF-OP-15				X										X				

RSNF \ RUR	RUR-01	RUR-02	RUR-03	RUR-04	RUR-05	RUR-06	RUR-07	RUR-08	RUR-09	RUR-10	RUR-11	RUR-12	RUR-13	RUR-14	RUR-15	RUR-16	RUR-17	RUR-18
RSNF-OP-16				X														
RSNF-OP-17				X														
RSNF-OP-18																		X
RSNF-OP-19																		X
RSNF-DO-01																	X	
RSNF-DO-02																	X	
RSNF-DO-03							X										X	
RSNF-DO-04							X										X	
RSNF-DO-05							X										X	
RSNF-DO-06																	X	

Tabla 143: Trazabilidad de RUR - RSNF

4 Diseño

En este apartado se muestra el diseño planteado para desarrollar la aplicación. Para ello se ve una visión general, prácticamente estructural, por medio de la arquitectura que debe seguir todo el diseño del proyecto y una parte detallada de cada uno de los módulos de la arquitectura por medio de varios diagramas de clases UML. Se utiliza este lenguaje ya que Android es una tecnología basada en Java y el servidor a montar se aprovechará para seguir el mismo lenguaje y el lenguaje de modelado que mejor se interpreta para entender los modelos es el lenguaje de modelado UML.

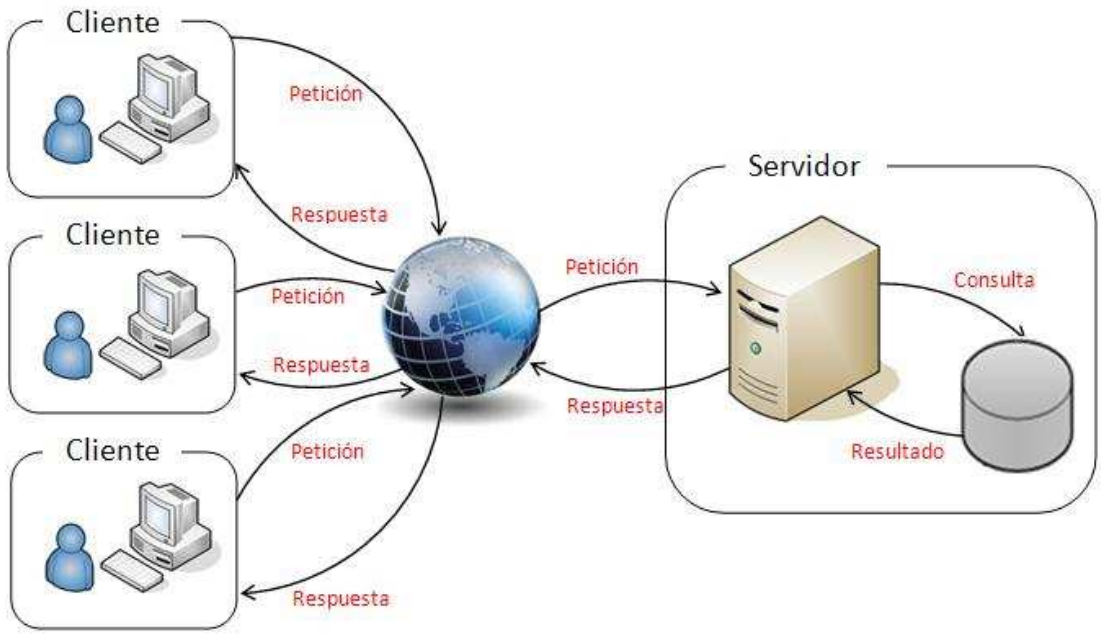
4.1 Arquitectura

En este apartado se va a detallar la arquitectura que se ha pensado para el desarrollo de la aplicación. Para ello se analiza cuáles son los componentes necesarios, para el cumplimiento de los requisitos detallados y los objetivos propuestos, y el modo de conexión entre ellos, de forma que sea un diseño lo más óptimo posible. Para poder entender de la mejor la arquitectura planteada se irá detallando módulo por módulo haciendo un resumen final con una visión global.

4.1.1 Cliente - Servidor

En el análisis se ha visto que existen dos partes diferenciadas dentro del proyecto que se pretende desarrollar. Por un lado se tiene la parte del cliente. Ésta sería la aplicación para los móviles, desarrollada en este caso para móviles con sistema operativo Android. Por otro lado se tiene la parte del servidor que recibe la información de un usuario y la distribuye al resto de los usuarios, que previamente hayan sido registrados en el sistema. Por ello la base de la arquitectura será un Cliente - Servidor.

Una arquitectura Cliente - Servidor consiste en dos módulos diferenciados. Un módulo del sistema hace consultas a un servidor o responde ciertas peticiones que necesita dicho servidor. Éste módulo es el componente cliente. En sí el cliente tiene la interfaz que se le muestra al usuario y es con la que éste maneja la aplicación. Por otro lado se dispone de un segundo componente en el sistema que recibe las peticiones del cliente y se las resuelve. Este componente es el servidor. Normalmente el servidor tiene una base de datos al que le realiza consultas sobre la información y así poder responder a las peticiones del cliente. Por tanto la arquitectura tiene el siguiente diseño.



Ilustracion 4. Arquitectura Cliente - Servidor

El protocolo normal de actuación es el siguiente:

- Uno de los clientes envía por medio de una red de comunicación una petición al servidor.
- El servidor para poder darle una respuesta consulta la base de datos.
- La base de datos le devuelve el resultado de la consulta al servidor.
- El servidor envía una respuesta al cliente con los datos deseados por medio de una red de comunicación.
- El cliente procesa la respuesta.

Pero para la aplicación que se va a desarrollar se pretende usar un protocolo algo distinto que es otro de los protocolos de actuación:

- El cliente envía su información al servidor por medio de una red de comunicación.
- El servidor guarda en la base de datos la información.
- La base de datos almacena los datos.
- El servidor devuelve una respuesta al cliente por medio de una red de comunicación.
- El cliente procesa la respuesta.
- Cada cliente periódicamente realiza consultas al servidor siguiendo el protocolo normal de actuación.

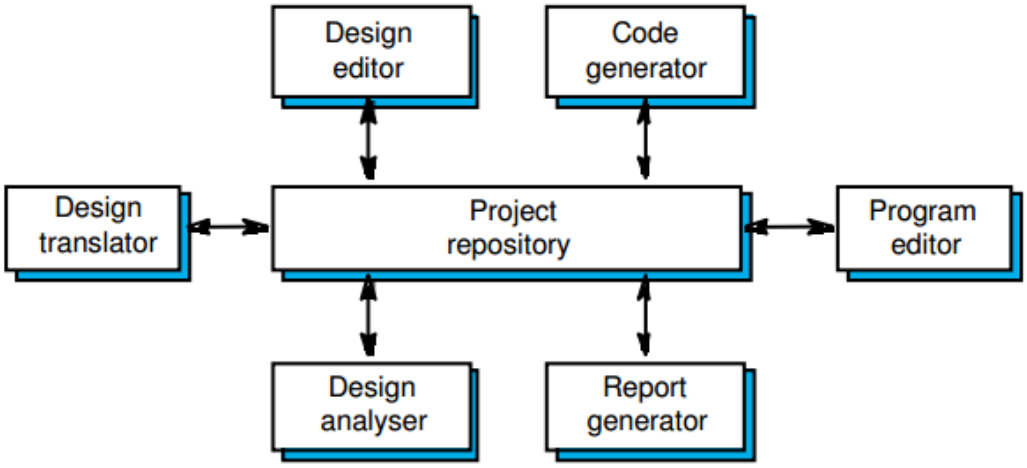
Esta arquitectura muchas veces se la denomina arquitectura Cliente - Servidor de tres capas ya que la tercera capa podría atribuírsele a la base de datos y la conexión entre la capa de la base de datos y el servidor son las consultas SQL que realiza el servidor sobre la base de datos.

4.1.2 Repositorio de Base de Datos

Por otra parte los clientes deben registrarse en el sistema y ello se hará por medio del servidor, almacenando en una base de datos la información necesaria para que un usuario quede registrado debidamente. También el servidor almacena en dicha base de datos la información de los encuentros creados pudiendo de este modo que el usuario receptor pueda solicitar al servidor si tiene algo de información para ellos. De esta forma los usuarios podrán sincronizarse gracias a las consultas al servidor. Por ello el servidor tiene que consultar a la base de datos por cada petición que se vaya haciendo, por lo que en la parte del Servidor se tendrá otra arquitectura a modo de repositorio con la base de datos.

El diseño de la arquitectura de repositorio consiste en una arquitectura construida principalmente en torno a los datos. Consiste en aislar todos los componentes de los datos, por lo que se queda en un módulo a parte la base de dato., El resto de componentes, en el caso de la aplicación a desarrollar será el servidor, le hacen consultas para obtener ciertos datos.

El diseño es el siguiente:



Ilustracion 5. Arquitectura de Repositorio

Como se puede apreciar cada módulo le consulta al repositorio de datos y obtiene los resultados de las consultas.

4.1.3 Modelo-Vista-Controlador

Dentro del cliente, de la arquitectura Cliente - Servidor, irá la aplicación como tal del dispositivo desarrollado para móviles Android. Como toda arquitectura de aplicación para móvil con un servidor con el que comunicarse, debe tener una arquitectura interna de Modelo-Vista-Controlador (MVC). Esta arquitectura tiene varias versiones y dependiendo del producto se diseña emplean una versión u otra. Antes de explicar las versiones que se tienen y las ventajas y desventajas de cada una de ellas se explicará el porqué de coger dicho diseño interno.

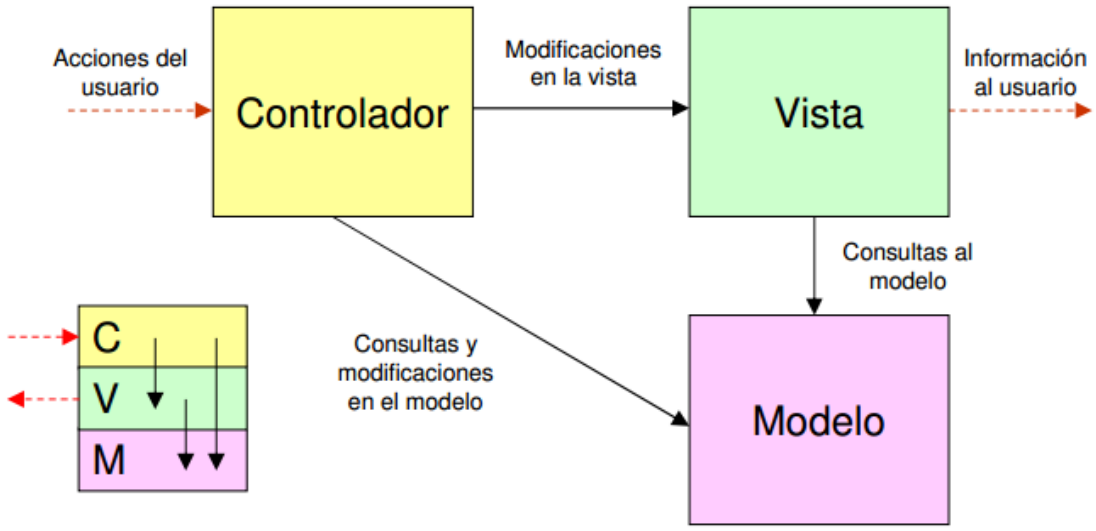
Si analizamos qué debe hacer la aplicación dentro del dispositivo móvil se considera que por un lado debe tener la interfaz gráfica con la que el usuario se comunicará y donde se mostrará la información enviada por el servidor. Esta interfaz es lo que se denomina la Vista de la arquitectura. Por otro lado se deben mandar información al servidor y recibir la información del servidor, por lo que dichos mensajes deben estar controlados antes de hacer uso de ello, al igual que la información que transcurre entre el usuario y el núcleo del programa. Por lo que debe existir un componente Controlador. Y por último debe haber un motor o un núcleo que

gestione las acciones de la vista y actúe en consecuencia a ellas, pudiendo mandar mensajes pertinentes al servidor y gestionar la información que nos otorga. Éste es el módulo Modelo.

Como ya se ha mencionado anteriormente existen varias versiones de esta arquitectura. Dependiendo de la comunicación entre los tres módulos se consiguen unas ventajas u otras. Primero se explicarán las versiones existentes y dependiendo de las características que otorgan se elegirá una de ellas.

4.1.3.1 MVC Versión Original

Por un lado se tiene la versión original de la arquitectura MVC. Ésta tiene el siguiente diseño:



Ilustracion 6. Arquitectura Modelo-Vista-Controlador original

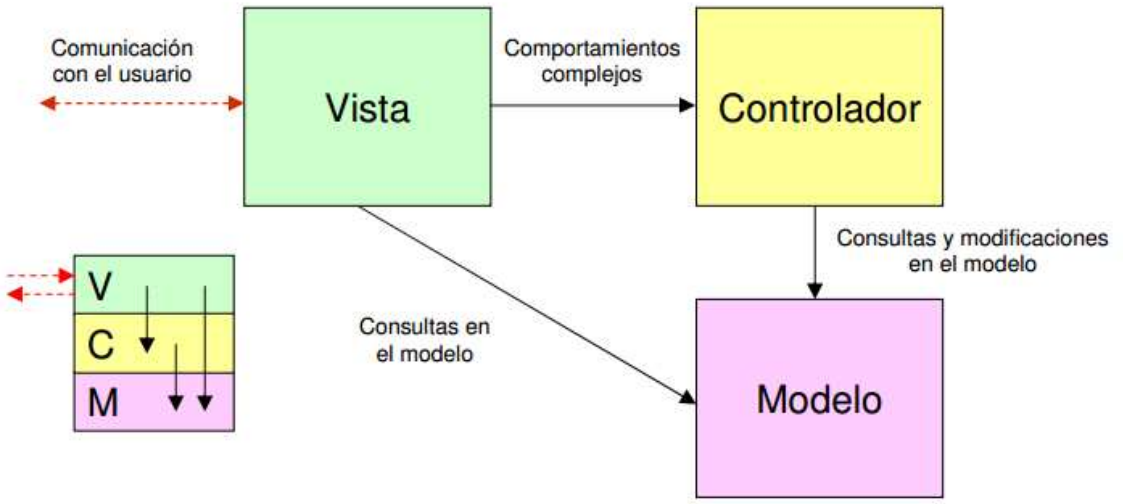
Esta versión presenta unas relaciones entre los componentes y una comunicación con el usuario que se representa como:

- **Modelo:** modelo conceptual, clases resultantes del análisis (refinadas en diseño). Es propiamente la funcionalidad del núcleo del programa.
- **Vista:** clases dedicadas a la presentación (gráfica) de la aplicación (interfaz de salida). En este caso son las pantallas de información al usuario.
- **Controlador:** gestiona los mensajes recibidos del usuario (interfaz de entrada) y los traduce a mensajes comprensibles por el modelo conceptual; es responsable, también, del flujo de la aplicación y del control de la navegación entre ventanas. Es el que ofrece las interfaces de entrada por parte del usuario y el que se encarga de la navegación entre las pantallas para conseguir una continuidad.

En sí podemos entender la analogía de dicha arquitectura como: Vista: los ojos del usuario, Controlador: sus manos, Modelo: su mente.

4.1.3.2 MVC Versión Moderna

Por otro lado se tiene la versión moderna de la arquitectura MVC. Ésta tiene el siguiente diseño:



Ilustracion 7. Arquitectura Modelo-Vista-Controlador moderno

Esta versión presenta otras relaciones entre los componentes y una comunicación distinta con el usuario, aunque el usuario no es consciente de ello, que se representa como:

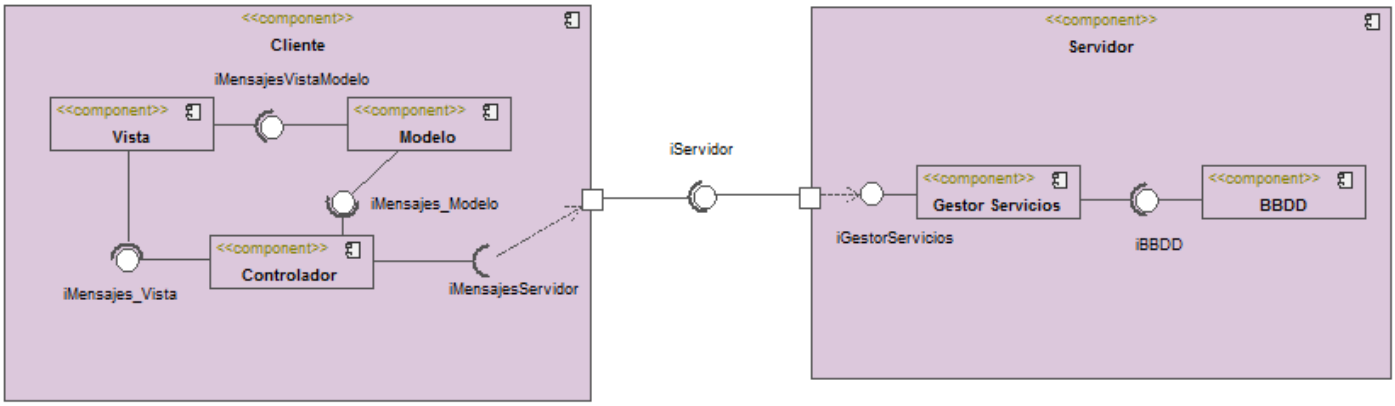
- **Modelo:** Al igual que en la versión original es modelo conceptual, clases resultantes del análisis (refinadas en diseño). Es propiamente la funcionalidad del núcleo del programa.
- **Vista:** engloba Vista original y la parte de interfaz de entrada de Controlador, y se añade la posibilidad tecnológica de generarla mediante aplicaciones de generación de GUI. Es toda la parte de la interfaz, ya sea de entrada como de salida.
- **Controlador:** centrado en el control interno de la aplicación, gestiona los mensajes recibidos de los distintos componentes del GUI, los verifica (correctos/completos), los traduce para el modelo conceptual, y aplica las reglas de negocio que estén definidas. Es sin más un traductor para que el modelo y la interfaz se entiendan.

Para el proyecto que se quiere realizar es más directo y tiene mejor administración usar esta versión moderna.

4.1.4 Arquitectura Visión Completa

Una vez que ya se ha visto por partes cada una de las arquitecturas que se van a integrar para resolver la aplicación que se quiere desarrollar, hay que ver y comprender el modo de encajarlas entre sí. Para ello se ha utilizado la herramienta Altova UModel para hacer un diagrama de componentes UML, de modo que se podrá visualizar el acoplamiento de los componentes.

La arquitectura final del sistema es la siguiente:



Arquitectura: Modelo de Componentes

Como se puede apreciar la estructura principal de la aplicación es el Cliente – Servidor en el cual ambos componentes se comunican por medio de la interfaz iServidor.

Dentro del Servidor nos encontramos con la arquitectura de repositorio con la Base de Datos. Como es normal en este tipo de diseños en Gestor de Servicios, que ofrece el Servidor al Cliente, consulta a la Base de Datos la información necesaria para devolvérsela al Cliente.

Por último el Cliente tiene la arquitectura Modelo-Vista-Controlador (MVC) versión moderna, en el que la Vista recoge y proporciona toda la información del usuario, el Controlador gestiona todos los mensajes; tanto de la Vista como del Servidor; que se dirigen al Módulo, y el Módulo que tiene la funcionalidad de la aplicación móvil. Todo este paso de mensajes se hace por sus respectivas interfaces.

4.1.5 Descripción de Componentes

En este apartado se van a describir los componentes de forma individual y cuáles son las relaciones que mantienen con el resto de componentes. Para ello se va a emplear el siguiente modelo de tabla:

Identificador	CO-XX
Nombre	
Tipo	
Función	
Interfaces	
Dependencias	
Origen	

Tabla 144: Descripción tabla componentes

- **Identificador:** Código a través del cual se identifica al componente, en este caso el identificador será CO-XX, donde CO significa Componente y XX el orden de creación.
- **Nombre:** Nombre del componente
- **Tipo:** tipo o nivel al que pertenece el componente. Para aclarar en qué consiste los niveles se explica a continuación la forma de nombrar a los componentes de cada nivel:
  - ***Subsistema:*** es el nivel más alto de la jerarquía de niveles, dentro de él se encuentran todos los componentes que realizan su funcionalidad.
  - ***Componente:*** Son todos los niveles intermedios, son módulos de funcionalidad compleja que cuelgan de los subsistemas y que son formados por otros componentes u otras clases.
  - ***Clase:*** es el nivel más bajo, sería lo equivalente a las hojas del árbol de componentes. Son módulos de funcionalidad sencilla que al agruparse con varias clases forman la estructura, insertada dentro de un componente, con funcionalidad compleja. En este tipo de diagramas de componentes no suelen aparecer, ya que en el Diseño Detallado de los apartados posteriores se muestra un diagrama de clases y se explica cada uno de ellos, pero sí pueden aparecer de forma excepcional en ciertas arquitecturas complejas.
- **Función:** Objetivos que debe cumplir el componente.
- **Interfaces:** Puntos de acceso que tiene el componente
- **Dependencias:** Enlace con otros componentes para ayudarse a llevar a cabo su función.
- **Origen:** requisitos de los que proviene la creación del componente.

A continuación mostramos los componentes de forma detallada:

Identificador	CO-01
Nombre	Cliente
Tipo	Subsistema
Función	Subsistema que controla la interfaz ofrecida al usuario y hace las consultas pertinentes al servidor
Interfaces	Ninguna
Dependencias	iServidor
Origen	RSF-01, RSF-28, RSF-29, RSNF-DE-01, RSNF-OP-04

Tabla 145: CO-01: Cliente

Identificador	CO-02
Nombre	Vista
Tipo	Componente
Función	Componente que controla la interfaz ofrecida al usuario mostrando las pantallas pertinentes con la información establecida y recogiendo los datos y acciones del usuario
Interfaces	Ninguna
Dependencias	iMensajesVistaModelo, iMensajes_Vista
Origen	RSF-02, RSF-03, RSF-04, RSF-05, RSF-06, RSF-07, RSF-08, RSF-09, RSF-10, RSF-11, RSF-12, RSF-13, RSF-14, RSF-15, RSF-16, RSF-18, RSF-19, RSF-20, RSF-21, RSF-22, RSF-23, RSF-24, RSF-25, RSF-26, RSF-27, RSF-28, RSF-29, RSF-30, RSNF-IN-01, RSNF-IN-02, RSNF-IN-03, RSNF-IN-04, RSNF-IN-05, RSNF-IN-06, RSNF-IN-07, RSNF-IN-08, RSNF-IN-09, RSNF-IN-10, RSNF-IN-11, RSNF-IN-12, RSNF-IN-13, RSNF-IN-14, RSNF-IN-15, RSNF-IN-16, RSNF-IN-17, RSNF-IN-18, RSNF-IN-19, RSNF-IN-20, RSNF-IN-21, RSNF-IN-22, RSNF-IN-23, RSNF-DE-01, RSNF-OP-01, RSNF-OP-02, RSNF-OP-03, RSNF-OP-05, RSNF-OP-06, RSNF-OP-07, RSNF-OP-08, RSNF-OP-10, RSNF-OP-11

Tabla 146: CO-02: Vista

Identificador	CO-03
Nombre	Controlador
Tipo	Componente
Función	Componente que controla los mensajes que se pasan de la interfaz al modelo y los mensajes que se envían y reciben del Servidor
Interfaces	iMensajes_Vista
Dependencias	iMensajes_Modelo, iMensajesServidor
Origen	RSF-05, RSF-15, RSF-16, RSF-20, RSF-25, RSF-28, RSF-29, RSF-31, RSNF-DE-01, RSNF-OP-01, RSNF-OP-03, RSNF-OP-04, RSNF-OP-07, RSNF-OP-08, RSNF-OP-09, RSNF-OP-10, RSNF-OP-11, RSNF-OP-18, RSNF-OP-19

Tabla 147: CO-03: Controlador

Identificador	CO-04
Nombre	Modelo
Tipo	Componente
Función	Componente que gestiona la funcionalidad de la aplicación del dispositivo móvil
Interfaces	iMensajesVistaModelo, iMensajes_Modelo
Dependencias	Ninguna
Origen	RSF-02, RSF-03, RSF-04, RSF-05, RSF-08, RSF-12, RSF-14, RSF-15, RSF-16, RSF-17, RSF-20, RSF-24, RSF-25, RSF-27, RSF-28, RSF-29, RSF-30, RSNF-DE-01, RSNF-BD-03, RSNF-BD-04, RSNF-OP-01, RSNF-OP-03, RSNF-OP-06, RSNF-OP-07, RSNF-OP-08, RSNF-OP-09, RSNF-OP-10, RSNF-OP-11, RSNF-OP-12, RSNF-OP-13, RSNF-OP-14, RSNF-OP-15, RSNF-OP-16, RSNF-OP-17

Tabla 148: CO-04: Modelo

Identificador	CO-05
Nombre	Servidor
Tipo	Subsistema
Función	Subsistema que realiza las funciones del servidor y tiene el almacén de datos
Interfaces	iServidor
Dependencias	Ninguna
Origen	RSF-01, RSF-04, RSF-05, RSF-32, RSF-33, RSF-34, RSF-35, RSNF-IN-11, RSNF-SE-01, RSNF-SE-02, RSNF-DE-02, RSNF-OP-04, RSNF-OP-18, RSNF-OP-19

Tabla 149: CO-05: Servidor

Identificador	CO-06
Nombre	Gestor_Servicios
Tipo	Componente
Función	Componente que gestiona las peticiones del cliente y realiza las consultas necesarias a la Base de Datos
Interfaces	iGestorServicios
Dependencias	iBBDD
Origen	RSF-05, RSF-15, RSF-16, RSF-32, RSF-33, RSF-34, RSF-35, RSNF-SE-01, RSNF-SE-02, RSNF-DE-02, RSNF-OP-04, RSNF-OP-18, RSNF-OP-19

Tabla 150: CO-06: Gestor_Servicios

Identificador	CO-07
Nombre	BBDD
Tipo	Componente
Función	Componente que gestiona la Base de Datos y devuelve las peticiones acerca de los datos que se le consultan
Interfaces	iBBDD
Dependencias	Ninguna
Origen	RSF-05, RSF-33, RSF-34, RSF-35, RSNF-SE-01, RSNF-SE-02, RSNF-BD-01, RSNF-BD-02

Tabla 151: CO-07: BBDD

4.1.6 Trazabilidad Requisitos Usuario – Componentes (RU-CO)

RS \ CO	CO-01	CO-02	CO-03	CO-04	CO-05	CO-06	CO-07
RSF-01	X				X		
RSF-02		X		X			
RSF-03		X		X			
RSF-04		X		X	X		
RSF-05		X	X	X	X	X	X
RSF-06		X					
RSF-07		X					
RSF-08		X		X			
RSF-09		X					
RSF-10		X					
RSF-11		X					
RSF-12		X		X			
RSF-13		X					
RSF-14		X		X			
RSF-15		X	X	X		X	

RS \ CO	CO-01	CO-02	CO-03	CO-04	CO-05	CO-06	CO-07
RSF-16		X	X	X		X	
RSF-17				X			
RSF-18		X					
RSF-19		X					
RSF-20		X	X	X			
RSF-21		X					
RSF-22		X					
RSF-23		X					
RSF-24		X		X			
RSF-25		X	X	X			
RSF-26		X					
RSF-27		X		X			
RSF-28	X	X	X	X			
RSF-29	X	X	X	X			
RSF-30		X		X			
RSF-31			X				
RSF-32					X	X	
RSF-33					X	X	X
RSF-34					X	X	X
RSF-35					X	X	X
RSNF-IN-01		X					
RSNF-IN-02		X					
RSNF-IN-03		X					
RSNF-IN-04		X					
RSNF-IN-05		X					
RSNF-IN-06		X					

RS \ CO	CO-01	CO-02	CO-03	CO-04	CO-05	CO-06	CO-07
RSNF-IN-07		X					
RSNF-IN-08		X					
RSNF-IN-09		X					
RSNF-IN-10		X					
RSNF-IN-11		X			X		
RSNF-IN-12		X					
RSNF-IN-13		X					
RSNF-IN-14		X					
RSNF-IN-15		X					
RSNF-IN-16		X					
RSNF-IN-17		X					
RSNF-IN-18		X					
RSNF-IN-19		X					
RSNF-IN-20		X					
RSNF-IN-21		X					
RSNF-IN-22		X					
RSNF-IN-23		X					
RSNF-SE-01					X	X	X
RSNF-SE-02					X	X	X
RSNF-DE-01	X	X	X	X			
RSNF-DE-02					X	X	
RSNF-DE-03							
RSNF-BD-01							X
RSNF-BD-02							X
RSNF-BD-03				X			
RSNF-BD-04				X			

RS \ CO	CO-01	CO-02	CO-03	CO-04	CO-05	CO-06	CO-07
RSNF-OP-01		X	X	X			
RSNF-OP-02		X					
RSNF-OP-03		X	X	X			
RSNF-OP-04	X		X		X	X	
RSNF-OP-05		X					
RSNF-OP-06		X		X			
RSNF-OP-07		X	X	X			
RSNF-OP-08		X	X	X			
RSNF-OP-09			X	X			
RSNF-OP-10		X	X	X			
RSNF-OP-11		X	X	X			
RSNF-OP-12				X			
RSNF-OP-13				X			
RSNF-OP-14				X			
RSNF-OP-15				X			
RSNF-OP-16				X			
RSNF-OP-17				X			
RSNF-OP-18					X	X	
RSNF-OP-19					X	X	
RSNF-DO-01							
RSNF-DO-02							
RSNF-DO-03							
RSNF-DO-04							
RSNF-DO-05							
RSNF-DO-06							

Tabla 152: Matriz de Trazabilidad RS - CO

4.2 Diseño Detallado

En este apartado se mostrará el diseño detallado de la aplicación. Para ello se cuenta con varios apartados, diferenciados por funcionalidad y uso, que ayudarán a la mejor comprensión de la solución propuesta.

Por un lado se detallarán los diagramas de clases que modulan el comportamiento de la aplicación, ya sea en el módulo del cliente y en el módulo del servidor.

Por otro lado de detallan los sistemas de almacenamiento de información para cliente y servidor, que es totalmente diferente entre ellos, salvando sus similitudes. En uno de ellos se usa una Base de Datos SQLite para el almacenamiento de datos del servidor y en otra un sistema de ficheros independientes para el almacenamiento de información del cliente.

Por último se detalla los protocolos diseñados para la comunicación entre ambos módulos, así como el prototipo de mensaje que se ha propuesto.

4.2.1 Diagrama de Clases

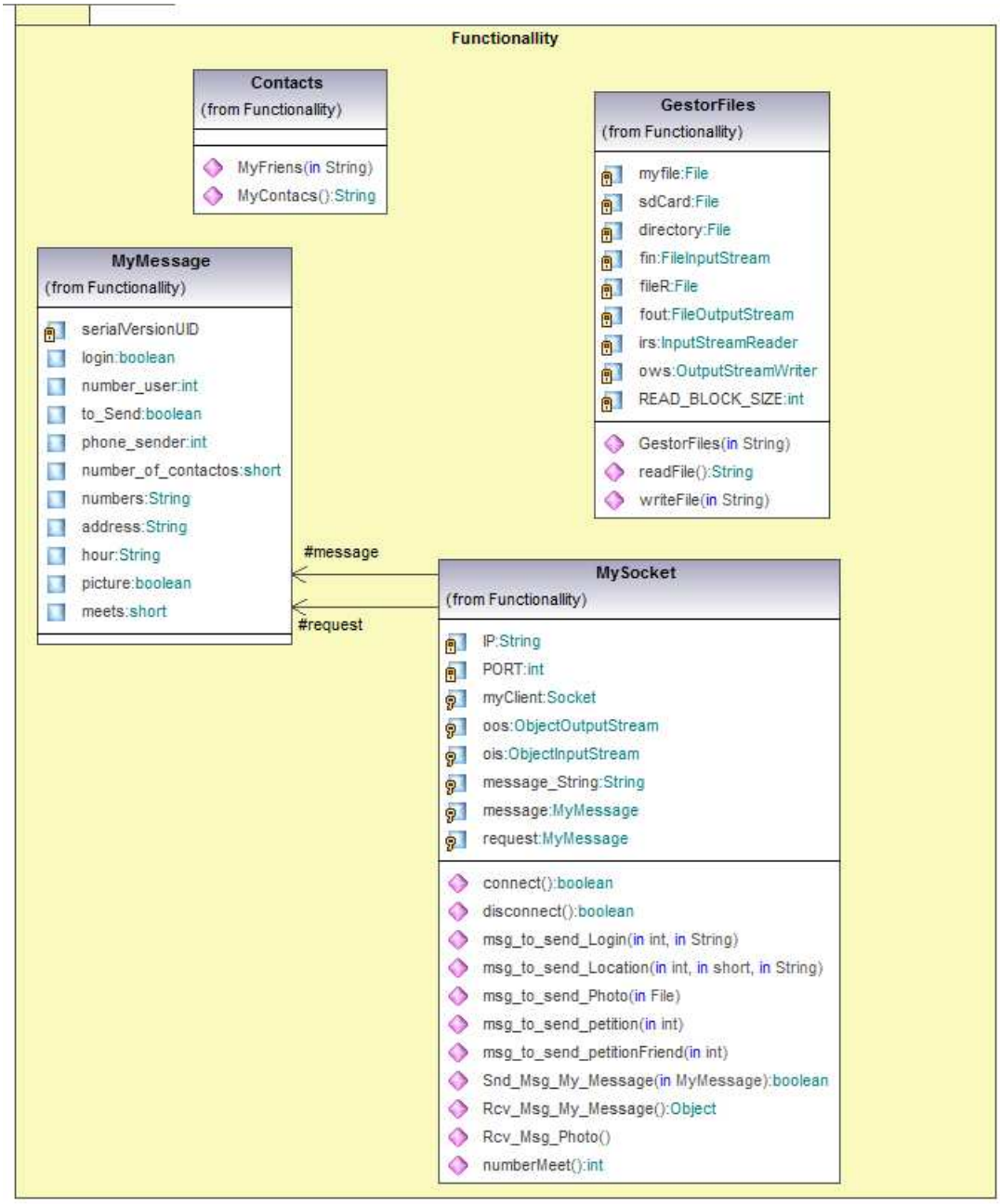
En este apartado se detalla el diagrama de clases que modula la aplicación. Para ello se detallarán siguiendo la arquitectura propuesta. Por un lado se mostrará el diagrama del módulo cliente y por otro el diagrama del módulo servidor.

4.2.1.1 Diagrama de Clases: CLIENTE

En este apartado se detalla y explica en profundidad el diagrama de clases del cliente. El cliente cuenta con dos paquetes de clases relacionadas entre sí que llevan por un lado la parte gráfica con su respectivo paso de mensajes y por otro lado la funcionalidad con el paso de mensajes con el componente servidor.

Para entender mejor la explicación se expone primero el diagrama y después se explicará cada uno de las clases, sus relaciones y sus usos.

Respecto al paquete de funcionalidad las clases diseñadas son las siguientes:



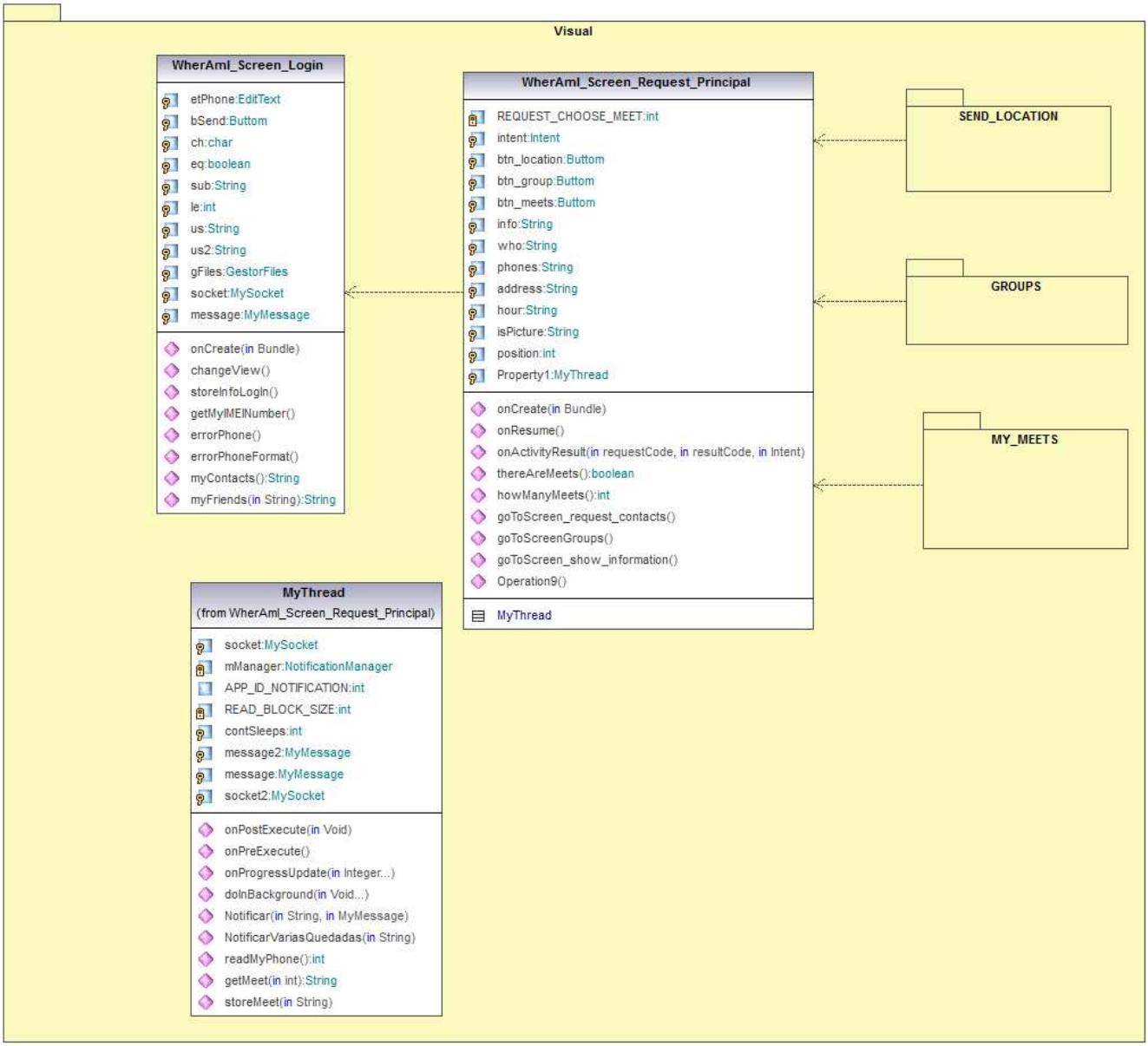
Ilustracion 8. Diagrama de clases cliente I: Paquete de funcionalidad

En este paquete se tienen las siguientes clases:

- **Contacts:** es la clase que se encarga de gestionar los contactos de la agenda y de comprobar cuáles de ellos están registrados en el sistema.
- **GestorFiles:** es la clase que se encarga de gestionar los archivos, es decir, leer y escribir de ellos.
- **MyMessage:** es una estructura que se ha creado para el protocolo de comunicación con el servidor. En él están todos los parámetros necesarios para entenderse con el servidor.
- **MySocket:** es la clase que se encarga de gestionar el socket, de enviar mensajes y recibir sus correspondientes mensajes.

Por otro lado se tiene el paquete de la interfaz, que dará uso a todas las clases del paquete funcionalidad. Para poder entender mejor el diagrama se ha dividido en varias secciones, para dar un concepto más visual y ayudar a su comprensión.

La primera parte sería la del registro en el sistema con los tres módulos generales que existen en la aplicación.



Ilustracion 9. Diagrama de clases cliente II: Paquete de visualización

En este paquete se muestra la pantalla que se usa para hacer el registro y la comunicación con la pantalla principal. A su vez vemos como la pantalla principal es la que va llamando a los diferentes módulos según el usuario los va solicitando y además tiene el hilo que corre en segundo plano que es el que se encarga de hacer las peticiones y de hacer las notificaciones.

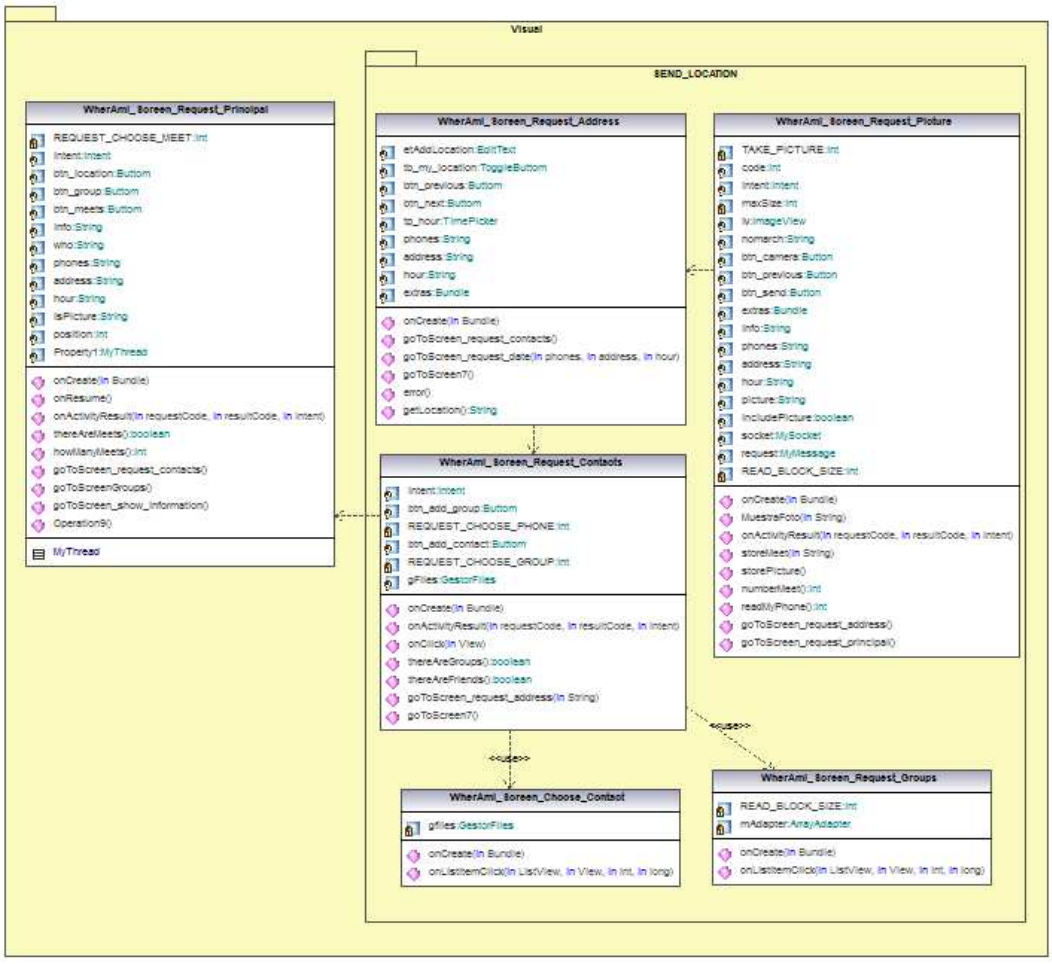
En esta ilustración se muestran las clases:

- **WherAmI_Screen_Login:** es la clase encargada del registro de los usuarios en el sistema. Hace una comprobación para corroborar que el número que se

inserta es un número móvil real y establece comunicación con el servidor para realizar el registro. Una vez que éste es exitoso pasa a la clase *WherAmI_Screen_Request_Principal*

- **WherAmI_Screen_Request_Principal:** es la clase que se encarga de la administración general de la aplicación. Desde esta clase se llama a cada uno de los módulos de los que se compone la aplicación móvil. Esta clase tiene una clase interna (*MyThread()*) en la cual se ejecuta un hilo secundario.
- **MyThread:** es la clase encargada de tener un hilo secundario en el que cada poco tiempo hace peticiones al servidor para comprobar si tiene nuevos encuentros como receptor. Además cada cierto tiempo también consulta si hay algún amigo nuevo registrado en el sistema. Cuando le llega un nuevo encuentro notifica al usuario la llegada de éste.
- **SEND_LOCALIZATION:** es el modulo que se encarga de enviar la localización.
- **GROUPS:** es el modulo que se encarga de la gestión de los grupos, así de su creación y de su eliminación.
- **MY_MEETS:** es el modulo que se encarga de gestionar los encuentros, tanto enviados como recibidos.

La segunda parte es el módulo de envío de la localización (SEND_LOCALIZATION). A continuación se presenta el diagrama de clases que lo modula:

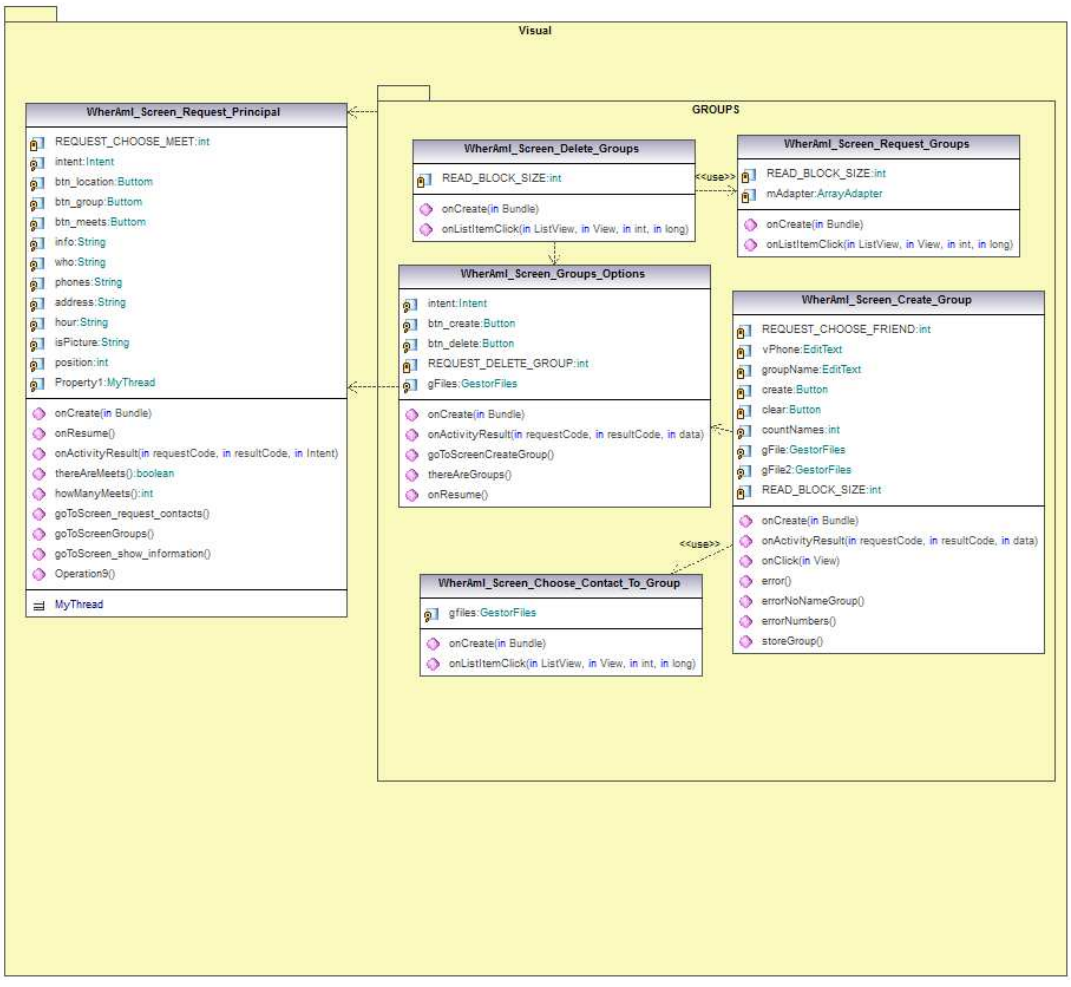


Ilustracion 10. Diagrama de clases cliente III: módulo de envío de petición

Esta parte se compone de las siguientes clases que desarrollan dicha funcionalidad:

- **WherAmI_Screen_Request_Contacts:** es la clase que se encarga de solicitar al usuario que elija a los contactos a los que se quiere enviar el encuentro. Para ello se ayuda de las listas que se generan en "WherAmI_Screen_Choose_Contact" y "WherAmI_Screen_Request_Group", las cuales son simples listas. Una vez que se eligen los contactos se pasa a la siguiente clase.
- **WherAmI_Screen_Request_Address:** es la clase que se encarga de solicitar al usuario la localización y la hora del encuentro. También usa los servicios de Android para coger la localización por GPS, 3G, 2G o Wifi. Una vez que se rellena la información se pasa a la siguiente clase.
- **WherAmI_Screen_Request_Picture:** es la clase que se encarga de solicitar la fotografía al usuario, haciendo uso del servicio de Android de la cámara, de enviar al servidor el encuentro creado, por medio del socket, y de almacenarlo en el archivo o archivos correspondientes usando la clase *GestorFiles*.

La tercera parte de la que se compone el paquete visual es el módulo de grupos, en el cual un usuario puede crear y eliminar grupos. A continuación se muestra el diagrama asociado a dicho módulo:

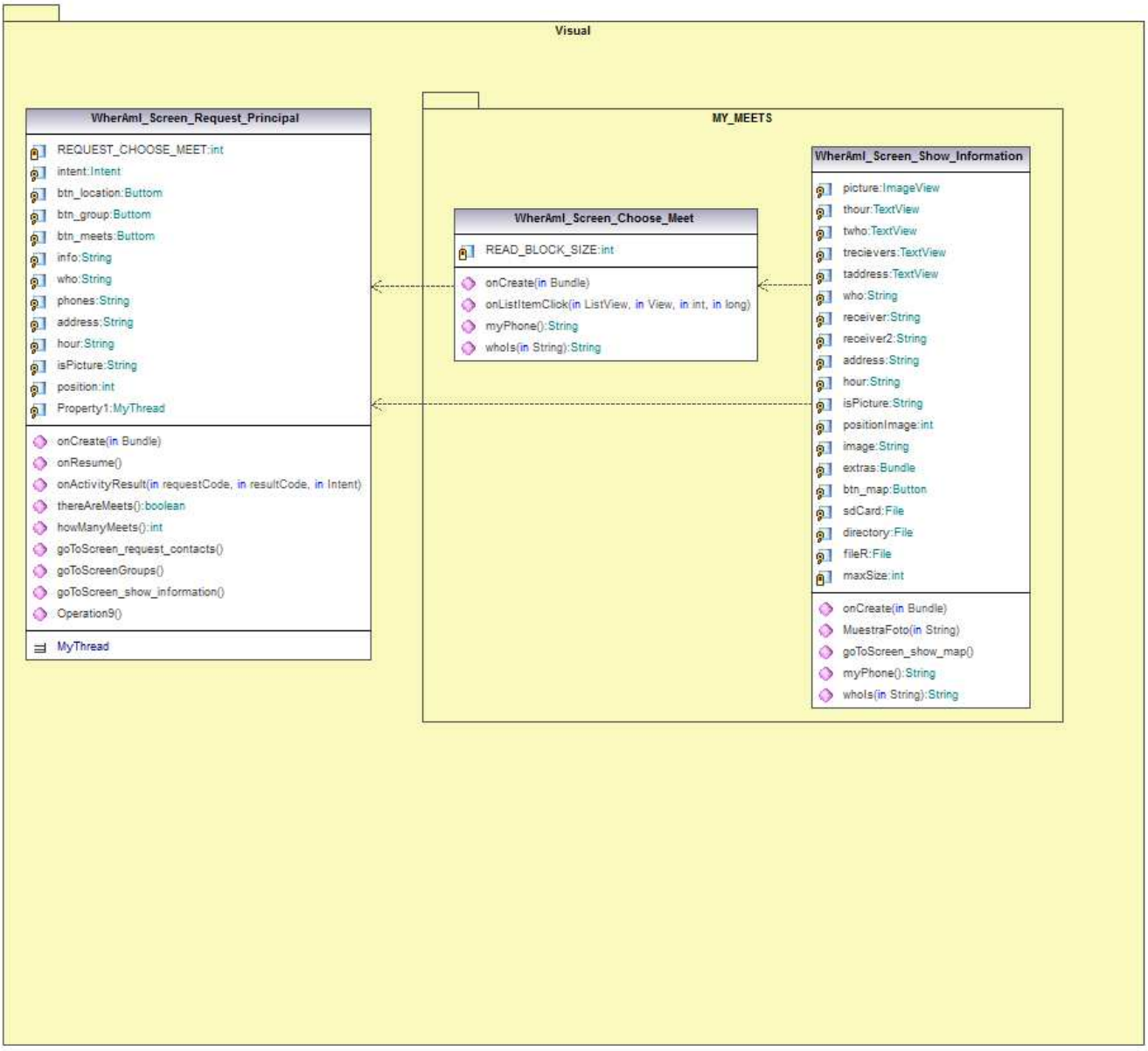


Ilustracion 11. Diagrama de clases cliente IV: módulo de grupos

Este módulo se accede desde la pantalla principal y tiene las siguientes clases:

- **WherAmI_Screen_Groups_Options:** es la clase que se encarga de gestionar los grupos. Da la opción de crearlos o de borrarlos
- **WherAmI_Screen_Create_Groups:** es la clase que se encarga de crear grupos. Para ello es necesario que inserte un nombre de grupo y selecciones los contactos por medio de la lista "WherAmI_Choose_Contact_To_Group". Una vez creados los almacena usando la clase *GestorFiles*.
- **WherAmI_Screen_Delete_Groups:** es la clase que se encarga de borrar los grupos. Se selecciona de una lista ("WherAmI_Screen_Request_Group") y una vez que se selecciona lo borra.

El cuarto y último módulo que compone la aplicación móvil es el módulo de encuentros el cual se encarga de la gestión de los encuentros, ya sean enviados o recibidos. A continuación se presenta el diagrama de clases:



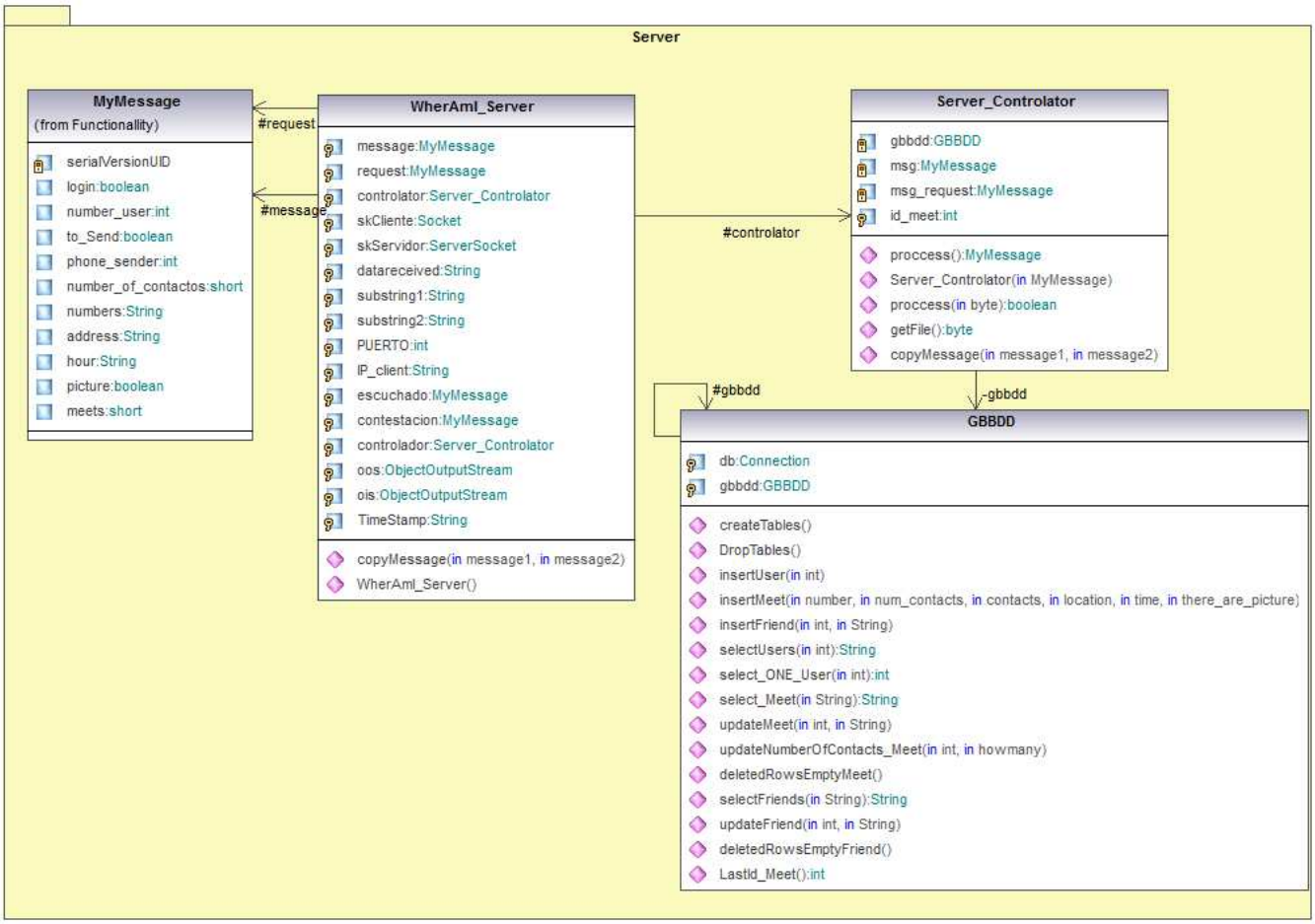
Ilustracion 12. Diagrama de clases cliente V: módulo de encuentros

Este es módulo está compuesto por las siguientes clases:

- **WherAmI_Screen_Choose_Meet:** esta es la clase encarga de seleccionar un encuentro y enviárselo a la pantalla en la que se muestra la información. Solo se accede una vez que la pantalla principal le llama.
- **WherAmI_Screen_Show_Information:** es la clase encarga de mostrar la información del encuentro, así como el emisor, receptor, hora, lugar, la fotografía y tiene un acceso para usar el servicio de Google Maps. A esta clase se llega desde la lista de encuentros o desde la notificación del hilo secundario.

4.2.1.2 Diagrama de Clases: SERVIDOR

En este apartado se muestra el diagrama de clases que modula la parte del servidor. Este paquete contiene todas las funciones propias del servidor que son recibir peticiones, hacer consultas a la Base de Datos, insertar en la Base de Datos y contestar las peticiones con la información solicitada. A continuación se muestra el diagrama de clases del servidor:



Ilustracion 13. Diagrama de clases del servidor

Este módulo está compuesto por las siguientes clases:

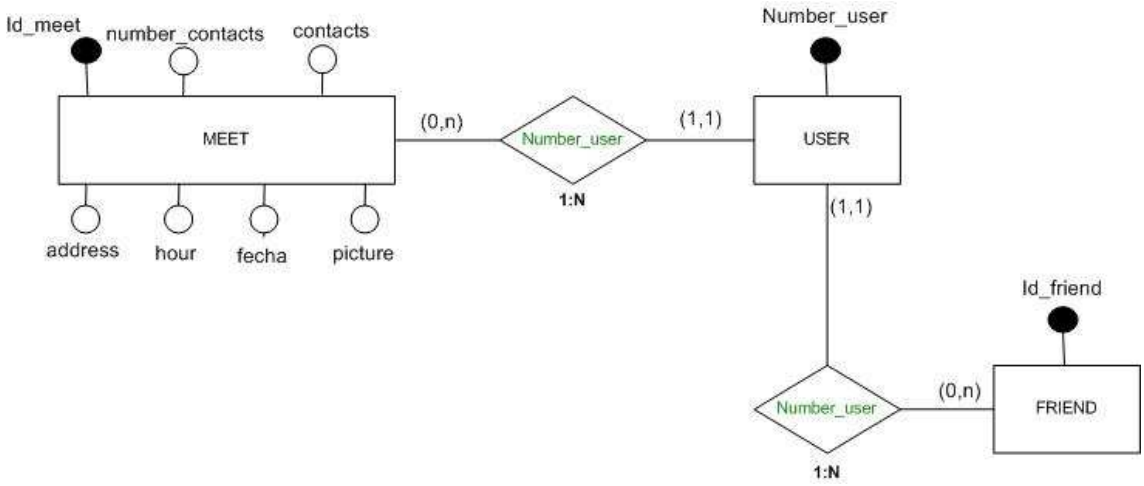
- **MyMessage:** esta es una clase que da la estructura de los mensajes que se pasan con el cliente. Esta clase es implementada en ambas aplicaciones. Gracias a ello es posible el protocolo de comunicación.

- **WherAmI_Server:** es la clase que se encarga de escuchar las peticiones que le llegan por el socket en forma de *MyMessage()*, consultar a su controlador y responder de acorde a ello por medio del socket.
- **Server_Controlator:** es la clase que se encarga de analizar los mensajes que ha recibido el servidor y actuar en consecuencia a ello. Para ello hace consultas a la Base de Datos
- **GBBDD:** es la clase que se encarga de hacer todas las gestiones con la Base de Datos, ya se inserciones, selecciones, borrados y actualizaciones. Para asegurar que es una única Base de Datos y sólo se accede desde un único sitio se ha usado el patrón Singleton.

4.2.2 Diseño de la Base de Datos del Servidor

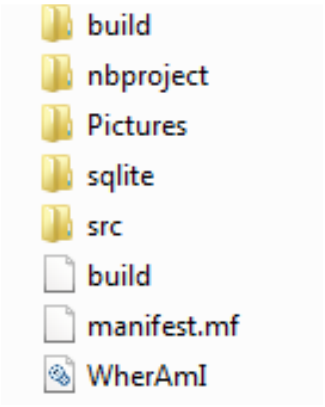
En este apartado se detalla la Base de Datos que se implementa en la parte del Servidor. Analizando lo que se pretende conseguir, los únicos datos que necesita el servidor son el registro de los usuarios, los encuentros y para que cada usuario sepa quién está registrado de sus amigos una tabla de amigos. El resto de información no es necesaria. Es más, para el registro de los usuarios sólo se guardará el número de teléfono, siendo así el usuario totalmente anónimo a la aplicación. De esta forma los encuentros y los amigos también serán anónimos. Para guardar las imágenes de los encuentros se crea una carpeta para guardar todas las imágenes y se identificarán por el número de encuentro al que pertenece, pero no será campo como tal de la tabla de encuentros.

El siguiente diagrama representa gráficamente la distribución de los datos en la Base de Datos:



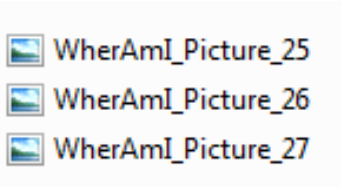
Ilustracion 14. Diagrama de la Base de Datos Relacional del servidor

En este diseño no se refleja el almacenamiento de las imágenes que se le envían a los usuarios. Todos los archivos que se envían en formato imagen y que por estándar se ha escogido el formato ".jpg" se almacenan en una carpeta llama "*Pictures*" que está en el mismo nivel que se encuentra la base de datos:



Ilustracion 15. Sistema de archivos del servidor

Para identificar las imágenes con su encuentro específico por medio del id_meet que se le inserta como parte del nombre del archivo .jpg. De tal modo que queda así:

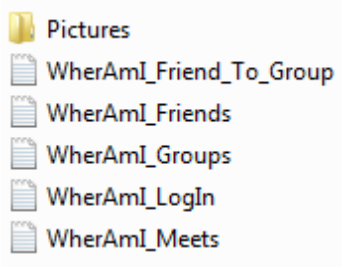


Ilustracion 16. Nombrado de las imágenes en la parte del servidor

De esta forma en caso de que en el encuentro se haya adjuntado una imagen, el servidor irá a la carpeta y cogerá la imagen identificada a dicho encuentro.

4.2.3 Diseño de la Base de Datos del Cliente

En este apartado se detalla el sistema de archivos de almacenamiento que tiene lugar en el módulo del cliente. Se ha diseñado un sistema de archivos independientes entre sí, salvo los encuentros y las imágenes asociados a ellos, que al igual que en el módulo del servidor, se identifica por el número de encuentro al que se adjunta. Los archivos diseñados para este módulo son los siguientes:

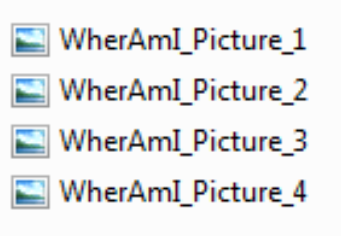


Ilustracion 17. Sistema de archivos de almacenamiento del cliente

La información que tiene cada fichero es la siguiente:

- **WherAmI_Login:** es el archivo que se encarga de almacenar el teléfono con el que está operando.
  - Número de teléfono
  - Numero IMEI

- **WherAmI_Friends:** es el archivo que se encarga de almacenar los amigos del usuario.
  - Nombre del amigo: cogido de la agenda
  - Número de teléfono del amigo.
- **WherAmI_Friend_To_Group:** es el archivo que se encarga de almacenar los amigos que se pueden meter a un grupo porque aún no han sido seleccionados para dicho grupo. Gracias a este archivo se lleva el control de que en un grupo no puede haber usuarios repetidos.
  - Nombre del amigo: cogido de la agenda
  - Número de teléfono del amigo.
- **WherAmI_Groups:** es el archivo que se encarga de almacenar los grupos creados para operar de forma conjunta:
  - Nombre del grupo.
  - Nombres de los amigos: cogidos de la agenda, que componen el grupo
  - Números de teléfono de los amigos.
- **WherAmI_Meets:** es el archivo que se encarga de almacenar los encuentros:
  - Número del emisor
  - Número de los receptores
  - Dirección del encuentro
  - Hora del encuentro
  - Un campo que indica si tiene adjuntado un archivo o no: en caso de tener un archivo adjunto le recuperaría de la carpeta “Pictures”.
- **Pictures:** es una carpeta que guarda todas las imágenes de los encuentros. La forma de identificarse con su respectivo encuentro es por el número de encuentro.



Ilustracion 18. Sistema de almacenamiento de imágenes del cliente

4.2.4 Diseño del protocolo de comunicación

En este apartado se detalla el protocolo de comunicación que se ha diseñado para establecer un contacto seguro entre los dispositivos móviles y el servidor. Para comunicar los usuarios con el servidor se ha decidido usar sockets, un enlace directo con el servidor, pero que se tiene que definir un protocolo para que los dispositivos se entiendan.

El protocolo consiste en cuatro fases diferentes:

- **Registro:** es la fase en la que el usuario se comunica con el servidor para registrarse en el sistema y los dos extremos se empiecen a entender.
- **Envío de encuentro:** es la fase en la que el cliente manda al servidor el encuentro para que lo almacene y así, cuando los usuarios especificados lo pidan, que el servidor se los entregue.
- **Petición de encuentro:** es la fase en la que el cliente solicita nuevos encuentros que le hayan mandado y el servidor se los otorga
- **Petición de amigos:** es la fase en la que el usuario solicita al servidor que le comunique cuales de las personas que conoce se encuentra registrada, para que lo actualice el propio usuario en su fichero correspondiente.

Para poder llevar a cabo toda comunicación se ha diseña un mensaje que tanto el cliente como el servidor han implementado. Este mensaje aparece en el diagrama de clases con el nombre MyMessage(). Se compone de diez campos con información relevante en ambos extremos de la comunicación, que son:

- **Login:** campo que comunica si el cliente quiere o no hacer registro.
- **Number_user:** campo que comunica el número de usuario que quiere registrar.
- **To_Send:** campo que comunica si el usuario es el que envía el encuentro.
- **Phone_sender:** campo que comunica el teléfono del usuario que envía el encuentro.
- **Number_of_contacts:** campo que comunica a cuantos usuarios se les dirige el encuentro.
- **Numbers:** campo que comunica los números de teléfono de los usuarios a los que se dirige el encuentro.
- **Address:** campo que comunica la dirección donde se ha concretado el encuentro
- **Hour:** campo que comunica la hora a la que se quiere establecer el encuentro con los usuarios receptores
- **Picture:** campo que comunica si se ha adjuntado o no una foto del lugar.
- **Meets:** campo que comunica el número de encuentros que el servidor nos quiere enviar, ya que en muchas ocasiones los receptores se pueden quedar sin línea, por diversos motivos, y se le tiene que comunicar de todos los encuentros que se le hayan enviado.

A continuación se detalla cada una de las fases en más profundidad.

4.2.4.1 Protocolo de Registro de Usuarios

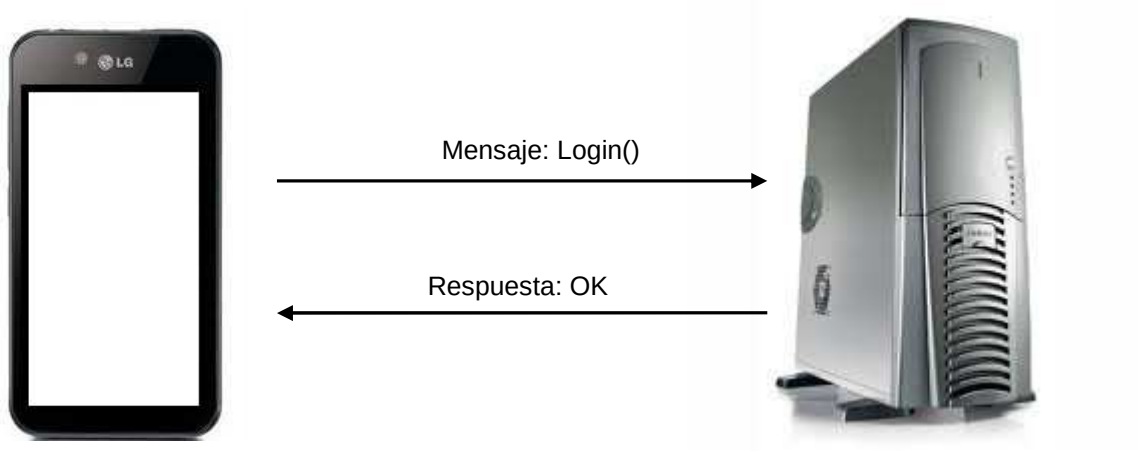
En este apartado se detalla el protocolo en el que un nuevo usuario se registra en el sistema. Para ello la aplicación le solicita su número de teléfono, para que pueda identificarle siempre que establezca nuevas comunicaciones. El protocolo es muy sencillo, el usuario envía el mensaje con la siguiente información:

- **Login:** verdadero
- **Number_user:** teléfono
- **To_Send:**.falso
- **Phone_sender:** 0
- **Number_of_contacts:** 0
- **Numbers:** vacío
- **Address:** vacío
- **Hour:** vacío
- **Picture:** falso
- **Meets:** 0

Es decir, que lo rellena todo al valor vacío excepto los campos de login, en el cual le dice que quiere registrarse con dicho número.

El servidor le contestaría con un mensaje similar, en respuesta de que el registro se ha producido con éxito.

En la siguiente ilustración se muestra el paso de mensajes:



Ilustracion 19. Protocolo de registro de usuarios

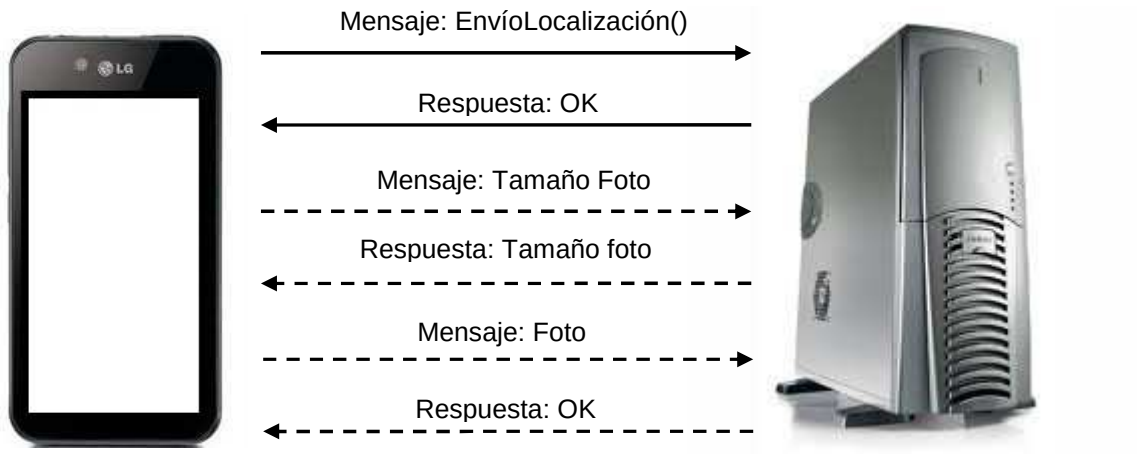
4.2.4.2 Protocolo de Envío de Encuentro

En este apartado se detalla el protocolo en el que un usuario envía un encuentro al servidor para que este lo distribuya a sus respectivos usuarios. El protocolo consiste en que el usuario le envía el siguiente mensaje.

- **Login:** falso
- **Number_user:** teléfono
- **To_Send:**.verdadero
- **Phone_sender:** teléfono
- **Number_of_contacts:** el número de contactos a los que se le quiere enviar
- **Numbers:** números de contactos
- **Address:** dirección
- **Hour:** hora
- **Picture:** verdadero o falso
- **Meets:** 0

Es decir, le envía toda la información para almacenar el encuentro. Dependiendo de sí el campo *picture* es verdadero o falso se establecen unos mensajes para enviar la foto. Si es verdadero, el dispositivo le manda primero el tamaño de la foto y el servidor le contesta con dicho tamaño. En caso de que sea diferente el tamaño se lo volvería a mandar. Después le manda un hilo de bytes que es la imagen. Cuando el servidor lo haya leído todo le manda una respuesta positiva. En caso de respuesta negativa se le volvería a mandar la imagen.

En la siguiente ilustración se muestra el paso de mensajes:



Ilustracion 20. Protocolo de envío de encuentros

Como se puede apreciar hay varios mensajes que tienen la línea discontinua. Eso quiere decir que el mensaje es opcional, dependiendo de si se ha adjuntado o no una foto se enviarán dichos mensajes.

4.2.4.3 Protocolo de Petición de Encuentro

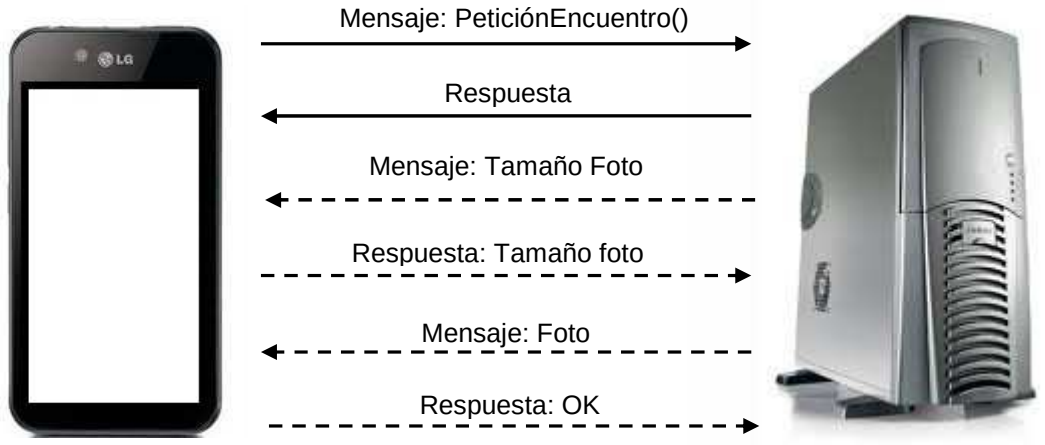
En este apartado se detalla el protocolo en el que un usuario envía un mensaje de petición de encuentros registrados como él como receptor. El protocolo consiste en que el usuario le envía el siguiente mensaje.

- **Login:** falso
- **Number_user:** teléfono
- **To_Send:**.falso
- **Phone_sender:** 0
- **Number_of_contacts:** vacío
- **Numbers:** vacío
- **Address:** vacío
- **Hour:** vacío
- **Picture:** falso
- **Meets:** 0

Es decir, le envía un mensaje vacío. El servidor lo interpreta como que quiere pedir los encuentros que hayan sido enviados para él. El servidor le contesta con un mensaje similar si no hay encuentros o con un mensaje relleno si es que existen. Si se le envía un encuentro, dependiendo de sí el campo picture es verdadero o falso, se establecen unos mensajes para enviar la foto al usuario. Si es verdadero, el servidor le manda primero el tamaño de la foto y el cliente le contesta con dicho tamaño. En caso de que sea diferente el tamaño se lo volvería a mandar. Después le manda un hilo de bytes que es la imagen. Cuando el cliente lo haya leído todo le manda una respuesta positiva. En caso de respuesta negativa se le volvería a mandar la imagen.

El protocolo se repite cada diez segundos, que es lo necesario para que los usuarios vean que todo funciona correctamente. El resto de tiempo el hilo de ejecución se encuentra dormido esperando a que el sistema del reloj de Android le despierte para volver a hacer la petición

En la siguiente ilustración se muestra el paso de mensajes:



Ilustracion 21. Protocolo de petición de encuentros

Como se puede apreciar hay varios mensajes que tienen la línea discontinua. Eso quiere decir que el mensaje es opcional, dependiendo de si se ha adjuntado o no una foto se enviarán dichos mensajes.

4.2.4.4 Protocolo de Petición de Amigos

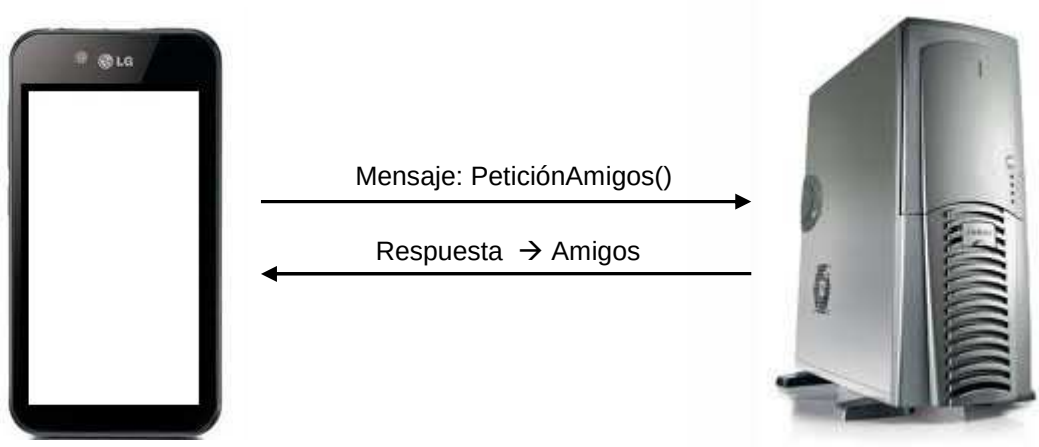
En este apartado se detalla el protocolo en el que un usuario envía un mensaje de petición de amigos registrados. El protocolo consiste en que el usuario le envía el siguiente mensaje.

- **Login:** falso
- **Number_user:** teléfono
- **To_Send:**.falso
- **Phone_sender:** teléfono
- **Number_of_contacts:** vacío
- **Numbers:** teléfonos móviles de la agenda.
- **Address:** vacío
- **Hour:** vacío
- **Picture:** falso
- **Meets:** 0

Es decir, le envía un mensaje vacío, pero con el campo phone_sender y numbers rellenos, para que el servidor sepa distinguir del otro mensaje de petición. El servidor lo interpreta como que quiere pedir los amigos registrados que hayan sido enviados para él. El servidor le contesta con un mensaje rellenando el campo de numbers, que serán los números que estén registrados en el sistema. El usuario los almacenará como sus amigos.

Este protocolo se hace cada periodo de tiempo elevado, alrededor de diez minutos, aunque el parámetro es programable.

En la siguiente ilustración se muestra el paso de mensajes:



Ilustracion 22. Protocolo de petición de amigos

5 Implementación

En este apartado se especifica cómo se ha desarrollado la implementación, así como los problemas que han ido surgiendo a la hora de desarrollar y las soluciones que se han propuesto.

Para seguir un orden primero se explica el módulo del cliente y después del servidor. La comunicación entre ambas partes ya se ha explicado en el apartado anterior en el diseño de protocolos, por lo que cuando se haga mención a dicha comunicación, solo referiré el protocolo usado.

5.1 Implementación Cliente

En este apartado se detalla los pasos seguidos en la implementación de la aplicación móvil desarrollada en Android. Para seguir un orden se explicará por pantallas según su uso normal.

Antes de empezar se tiene que decir que en todas las pantallas siguen un mismo estilo presentación para que el usuario en todo momento sepa que acciones puede realizar y por asociación de ideas la aplicación le sea más fácil de manejar. Estos estilos se pueden desarrollar en Android definiéndolos en ficheros aparte, muy parecido a los estilos .css para páginas web.

5.1.1 Módulo registro

El primer problema que se encuentra el desarrollador es que necesita un teléfono para poder operar con dicho usuario y una de las soluciones que se pensaron y se examinaron fue llamar al servicio de Android "*Mi número de teléfono*" el cual llama a "Ajustes >> Acerca del teléfono >> Estado >> Mi número de teléfono". El problema, y se puede comprobar, es que en todos los teléfonos de España dicha información es ilegal. Hay pocos países que permitan ofrecer dicha información. Por lo que la única solución propuesta es pedirle el número al propio usuario y confiar que realmente es su número de contacto.

Como hacer esta petición puede resultar algo violenta para los usuarios se le añadió un texto explicando el uso que se va a realizar con dicha información.

Una vez introducido nos comunicamos con el servidor con el protocolo de registro y guardamos el registro en el archivo "WherAmI_Login.txt". El número IMEI si está disponible desde los registros de Android, es más, este número se le tiene que ofrecer al usuario para identificar su terminal y poder hacer trámites como liberación de móviles.

5.1.2 Módulo principal

Este módulo es el que se le presenta al usuario nada más arrancar la aplicación y una vez hecho el correspondiente registro. Se necesita que sea sencilla y fácil de manejar por lo que se le ofrece tres opciones básicas. Para evitar errores del usuario las opciones que no se puedan realizar en dicho momento se quedarán totalmente deshabilitadas. En este caso la opción "*Mis encuentros*" estaría deshabilitada puesto que aún no hay encuentros registrados. Una vez que haya encuentros, tanto enviados como recibidos, es opción se habilitará al usuario.

En este apartado había un problema y era que esta pantalla realmente nunca debería terminar mientras el usuario estuviera utilizando la aplicación, ya que por cualquier error o si se

quería volver atrás, se tenía que abrir esta pantalla. Para hacer la comprobación de habilitación de botones suponía un problema y se tuvo que estudiar a fondo el sistema de aplicaciones Android. La solución fue poner en la parte OnResume() dicha comprobación, así siempre que volviese a tomar el control esta pantalla se haría dicha comprobación. Una vez que existen encuentros el botón se habilitará y se podrá acceder al módulo de encuentros.

5.1.3 Módulo grupos

Este es un módulo muy sencillo en que sólo se puede crear o borrar grupos. Al igual que antes se deshabilita la opción de borrar grupo si no hay ningún grupo creado. Para resolver este problema se tomó una solución similar al del módulo principal.

Para crear grupos se dispone de una pantalla específica en la que pide un nombre de grupo y los integrantes del grupo. Por decisiones de diseño se ha propuesto poner un máximo de diez usuarios en un grupo, pero este parámetro es configurable. Los usuarios se insertan de una lista de amigos registrados, la cual se forma a partir de los contactos de la agenda y de los amigos que el servidor nos dice que están registrados por medio del protocolo de petición de amigos. Un usuario no puede estar insertado dos veces para un mismo grupo. Una vez que se crea un grupo este es almacenado por orden alfanumérico en el archivo “WherAmI_Groups.txt”

Para borrar un grupo accede al fichero “WherAmI_Groups.txt” y carga en una lista todos los grupos. Cuando se selecciona el grupo este es eliminado del fichero manteniendo el mismo orden.

5.1.4 Módulo envío de localización

Este módulo consta de tres pantallas sencillas que están enlazadas entré sí. Como pedir información al usuario puede ser estresante para él, se le ofrece en todo momento una miga de pan, para que sepa la información que se le va a pedir y sepa cuál es la información que se le pide en ese momento.

La primera pantalla es la petición de contactos, en la cual puede elegir un contacto o un grupo. En el caso del grupo, la opción debería estar deshabilitada si no hay grupos registrados. La navegación a la siguiente pantalla no se puede permitir hasta que no haya resuelto esta información.

La segunda pantalla es la petición de la dirección y la hora. Para rellenar el campo dirección había que dejar la libertad al usuario de elegir entre escribirla él de forma manual o que eligiese la opción de que el sistema cogiese su localización exacta por medio de las líneas GPS o por redes móviles como 3G, 2G o Wifi. Para elegir la hora se le pone una ruleta en la que marca la hora actual y el usuario puede modificarlo para poner la hora que quiere realizar el encuentro. No se puede cambiar de pantalla hasta que el usuario al menos haya metido una dirección.

La tercera pantalla es la de petición de foto. En este momento el usuario podría mandar ya el encuentro, pero por si quiere adjuntar una fotografía del lugar se le presenta un botón que llama al sistema de la cámara del propio dispositivo, haciendo uso de servicio que nos ofrece Android. Una vez que envía el encuentro por medio del protocolo de envío de petición, la aplicación almacena el encuentro creado en “WherAmI_Meets.txt” y en caso de haber adjuntado una fotografía está la almacenará en la carpeta “Pictures” siguiendo el nombrado ya explicado anteriormente.

5.1.5 Módulo encuentros

Este módulo dispone de una lista de encuentros registrados, ya sea por los que el propio usuario ha enviado como por lo que ha recibido. Estos encuentros los carga del archivo “WherAmI_Meets.txt” y se presentan de más reciente a más antiguo. Una vez que se selecciona un encuentro se abre una pantalla con toda la información del encuentro, en el que si tiene una fotografía adjuntada se mostrará en la parte de abajo.

En toda la información había que poner los nombres y no los números de teléfono, para ello se han tenido que hacer varias comprobaciones a los archivos “WherAmI_LogÍn.txt” para coger el número del usuario y “WherAmI_Friends.txt” para coger el nombre de los amigos. Así de esta forma aparecerá el nombre “YO” cuando sea el número del usuario y aparecerá el nombre del amigo cuando no sea el número del usuario.

La información se presenta de forma ordenada, concisa y directa, con la opción en todo momento de abrir el mapa y ver la localización. Una vez que se pulsa el botón ver mapa había que llamar al servicio ofrecido por Android de Google Maps. Se intentó reproducir este servicio como un módulo propio, pero se descartó por varias razones, además de que el servicio de Google Maps cuenta con ciertos apoyos de indicaciones para llegar al sitio.

5.1.6 Módulo de notificaciones

Este posiblemente haya sido el módulo más costoso y más difícil de solucionar, aunque su implementación ha sido rápida puesto que los conocimientos cada vez eran más amplios. Este módulo consiste en un hilo secundario que empieza a funcionar una vez que hacemos el registro en el sistema y que cada diez segundos hace peticiones al servidor por medio del protocolo de petición de encuentro. De esta forma si el servidor indica que el usuario tiene encuentros como receptor, se carga una notificación con la información necesaria para la identificación, tanto de la aplicación como del encuentro.

Dentro de la propia notificación hay que cargar la pantalla a abrir una vez que se visualiza la notificación. En este caso abre la pantalla de información del módulo de encuentros, con toda la información correspondiente a al encuentro.

Una vez que se ha recibido un nuevo encuentro el hilo debe almacenarlo como ya se ha explicado anteriormente.

Además este hilo cada cierto tiempo manda una petición de amistad por medio del protocolo de petición de amigos. En caso de que haya nuevos amigos registrados el usuario los guardará en el fichero “WherAmI_Friends.txt”

5.2 Implementación Servidor

En este apartado se detalla los pasos seguidos en la implementación de la aplicación servidor.

Está compuesto de dos módulos que se comunican constantemente: el propio gestor y la base de datos desarrollada en SQLite.

El gestor es un módulo que está en permanente escucha por un puerto abierto a la mensajería y toma los mensajes enviados por el usuario, los analiza siguiendo los protocolos de mensajería explicados anteriormente y actúa conforme a ello. Para actuar hace las consultas y las inserciones necesarias la base de datos. Una vez que lo ha procesado crea un mensaje de respuesta y se lo envía al usuario.

La base de datos es un sistema que crea el propio servidor y que tiene que estar operativo en todo momento.

6 Pruebas

En este apartado se cada una de las pruebas unitarias que se tienen que realizar para chequear el correcto funcionamiento del sistema. Hay que resaltar que cada prueba es unitaria y por tanto no depende de ninguna de las anteriores, por lo que se tendría que partir desde el mismo punto de inicio que en este proyecto serán dos diferentes:

- **Pruebas de registro:** se parte del sistema recién arrancado según se instala, sin ficheros ni carpetas aún creados.
- **Pruebas de funcionamiento (aquellas que no son de registro):** se parte desde la primera pantalla una vez hecho el registro. Se tiene que hacer sin más información, ya que todo lo necesario se especifica en cada prueba unitaria.

Para especificar cada prueba se ha seguido la siguiente plantilla en forma de tabla que ayuda a focaliza la información:

PRU-XX	
Título	
Entrada Genérica	
Salida Genérica	
Entrada Específica	
Salida Específica	
RSF Relacionado	

Tabla 153: Diseño de tabla usada para los requisitos de Usuario

Siendo:

- **PRU-XX:** es el identificador del requisito.
  - La XX es un número de dos cifras que sigue el orden en el que se encuentra.
- **Título:** es un nombre genérico que describe de forma global en qué consiste la prueba.
- **Entrada Genérica:** son los pasos genéricos que se tienen que introducir en el programa para llevar a cabo la prueba.
- **Salida Genérica**: es la salida genérica que devuelve el sistema al ejecutar la entrada.
- **Entrada Específica**: son los pasos específicos que se le introduce al programa para ejecutar la prueba. En caso de que la entrada específica y la genérica sean iguales el valor de la entrada especifica será “---“.

- **Salida Específica**: es la salida específica que devuelve el sistema al ejecutar la entrada específica. En caso de que la salida específica y la genérica sean iguales el valor de la salida especifica será "---".
- **RSF Relacionado**: es el identificador del o de los requisitos Software funcionales que pretende testear la prueba.

A continuación se presentan las pruebas que se han diseñado para poder chequear correctamente el funcionamiento del sistema.

6.1 Pruebas de Registro

En este apartado se especifican las pruebas chequear el registro en el sistema y comprobar su correcto funcionamiento.

PRU-01	
Título	Insertar un nuevo usuario
Entrada Genérica	• Se arranca el programa por primera vez. • No existen ficheros en el móvil todavía. • Se rellena la información de la pantalla de registro. • Se pulsa el botón "ENVIAR".
Salida Genérica	• El usuario se crea correctamente. • Se abre la pantalla inicial del programa y sale un mensaje de bienvenida.
Entrada Específica	1. Se arranca el programa en el móvil. 2. Se rellena el número de teléfono, en este caso para probar se inserta el del propio usuario empezando por el número 6 (ej.: 698754321) 3. Se pulsa el botón "ENVIAR".
Salida Específica	• El usuario se crea correctamente. • Se abre la pantalla inicial del programa y sale el mensaje de bienvenida "Bienvenido a WherAmI !!".
RSF Relacionado	RSF-01

Tabla 154: PRU-01

PRU-02	
Título	Insertar un nuevo usuario con error en el campo “Teléfono”
Entrada Genérica	• Se arranca el programa por primera vez. • No existen ficheros en el móvil todavía. • Se rellena el campo del teléfono pero se ponen números incorrectos. • Se pulsa el botón “ENVIAR”.
Salida Genérica	• El usuario no se crea. • Sale un mensaje de error.
Entrada Específica	1. Se arranca el programa en el móvil. 2. Se rellena el número de teléfono de forma errónea con un: a) número que no empiece por 6 (ej.: 918754321) b) número que tenga más de 9 cifras(ej.: 61875438321) c) número que tenga menos de 9 cifras (ej.: 6184321) 3. Se pulsa el botón “ENVIAR”.
Salida Específica	a), b) y c): • El usuario no se crea. • Sale el mensaje de error “Teléfono incorrecto”.
RSF Relacionado	RSF-01, RSF-28

Tabla 155: PRU-02

6.2 Pruebas de Funcionamiento

A partir de este momento todas las pruebas se harán contando que el usuario ya está registrado correctamente y se partirá desde la primera pantalla, la pantalla principal. Las pruebas que se han de ejecutar y registrar su resultado son:

PRU-03	
Título	Crear un grupo nuevo
Entrada Genérica	• Se abre el menú de crear grupos • Se rellena el campo de “Nombre de grupo” • Se insertan usuarios en el grupo • Se pulsa “Crear”
Salida Genérica	• El grupo se crea correctamente. • Se limpia la información de la pantalla. • Sale un mensaje para saber que se ha creado correctamente.
Entrada Específica	1. Se abre la pantalla de crear grupos pulsando “*Grupos >> Crear grupo*” 2. Se rellena el campo de “Nombre de grupo” 3. Se insertan usuarios en el grupo 4. Se pulsa “Crear”
Salida Específica	• El grupo se crea correctamente. • La pantalla para crear grupos se limpia • Sale el mensaje de confirmación “El grupo se ha guardado con éxito”.
RSF Relacionado	RSF-02

Tabla 156: PRU-03

PRU-04	
Título	Crear un grupo sin nombre de grupo
Entrada Genérica	• Se abre el menú de crear grupos • No se rellena el campo de “Nombre de grupo” • Se insertan usuarios en el grupo • Se pulsa “Crear”
Salida Genérica	• El grupo no se crea. • Sale un mensaje para saber que se ha producido un error.
Entrada Específica	1. Se abre la pantalla de crear grupos pulsando “*Grupos >> Crear grupo*” 2. No se inserta nada en el campo “Nombre de grupo” 3. Se insertan usuarios en el grupo 4. Se pulsa “Crear”
Salida Específica	• El grupo no se crea. • La pantalla para crear grupos se queda igual • Sale el mensaje de error “El nombre del grupo está vacío”.
RSF Relacionado	RSF-02, RSF-28

Tabla 157: PRU-04

PRU-05	
Título	Crear un grupo con un nombre de grupo demasiado largo
Entrada Genérica	• Se abre el menú de crear grupos • Se rellena el campo de "Nombre de grupo" con más de 25 caracteres. • Se insertan usuarios en el grupo • Se pulsa "Crear"
Salida Genérica	• El grupo no se crea. • Sale un mensaje para saber que se ha producido un error.
Entrada Específica	1. Se abre la pantalla de crear grupos pulsando "*Grupos* >> *Crear grupo*" 2. Se rellena el campo "Nombre de grupo" con: "abcde12345abcde12345abcde1" 3. Se insertan usuarios en el grupo 4. Se pulsa "Crear"
Salida Específica	• El grupo no se crea. • El nombre del grupo se pone de color rojo • Sale el mensaje de error "El nombre del grupo tiene más de 25 caracteres".
RSF Relacionado	RSF-02, RSF-28

Tabla 158: PRU-05

PRU-06	
Título	Crear un grupo sin usuarios
Entrada Genérica	• Se abre el menú de crear grupos • Se rellena el campo de "Nombre de grupo" • No se inserta ningún usuario en el grupo • Se pulsa "Crear"
Salida Genérica	• El grupo no se crea. • Sale un mensaje para saber que se ha producido un error.
Entrada Específica	1. Se abre la pantalla de crear grupos pulsando "*Grupos >> Crear grupo*" 2. Se rellena el campo "Nombre de grupo" (ej.: Universidad) 3. No se inserta ningún usuarios en el grupo 4. Se pulsa "Crear"
Salida Específica	• El grupo no se crea. • La pantalla para crear grupos se queda igual • Sale el mensaje de error "No se ha insertado ningún contacto en el grupo".
RSF Relacionado	RSF-02, RSF-28

Tabla 159: PRU-06

PRU-07	
Título	Crear un grupo con más de diez usuarios
Entrada Genérica	• Se abre el menú de crear grupos • Se rellena el campo de "Nombre de grupo" • Se inserta diez usuarios en el grupo • Se pulsa añadir un usuario más
Salida Genérica	• El grupo no se crea. • Sale un mensaje para saber que se no se pueden insertar más usuarios al grupo.
Entrada Específica	1. Se abre la pantalla de crear grupos pulsando "*Grupos >> Crear grupo"* 2. Se rellena el campo *"Nombre de grupo"* (ej.: Universidad) 3. Se inserta diez usuarios en el grupo de la lista de contactos 4. Se pulsa añadir nuevo usuario
Salida Específica	• El grupo no se crea. • La pantalla para crear grupos se queda igual • Sale el mensaje de error "No se pueden añadir más usuarios".
RSF Relacionado	RSF-02, RSF-28

Tabla 160: PRU-07

PRU-08	
Título	Borrar un grupo
Entrada Genérica	• Se abre el menú de grupos • Se pulsa "Borrar Grupo" • Se elige el grupo a borrar
Salida Genérica	• El grupo se borra. • Sale un mensaje para informar el borrado.
Entrada Específica	---
Salida Específica	• El grupo se borra. • Sale el mensaje "Grupo <nombre_grupo> ha sido borrado", para saber que se ha borrado correctamente.
RSF Relacionado	RSF-03

Tabla 161: PRU-08

PRU-09	
Título	Añadir usuario a un grupo desde la lista de contactos
Entrada Genérica	• Se abre el menú de crear grupos • Se pulsa añadir contacto y se elige uno
Salida Genérica	• El usuario se añade a la lista de usuarios del grupo.
Entrada Específica	1. Se abre la pantalla de crear grupos pulsando "*Grupos >> Crear grupo*" 2. Se pulsa añadir contacto y se elige uno
Salida Específica	---
RSF Relacionado	RSF-04

Tabla 162: PRU-09

PRU-10	
Título	Comprobar que se puede elegir mandar un encuentro a un grupo o a un contacto
Entrada Genérica	• Se abre el menú mandar ubicación • Se comprueba la pantalla y las opciones que te da.
Salida Genérica	• Hay dos botones, uno para elegir un contacto y otro para elegir un grupo, y cada uno abre su lista respectiva
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localizacion*" 2. Se comprueba la pantalla y las opciones que te da.
Salida Específica	• Existen dos botones: o "Añadir un contacto": Que abre la lista de contactos o "Añadir grupo": Que abre la lista de grupos con los nombres de los componentes del grupo
RSF Relacionado	RSF-05

Tabla 163: PRU-10

PRU-11	
Título	Botón para coger la ubicación de forma automática que por defecto está desactivado
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige usuarios a mandar ubicación
Salida Genérica	• La pantalla de inserción de ubicación se abre. • Hay un botón para coger la ubicación de forma automática que se encuentra por defecto desactivado y el editor de texto vacío.
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige usuarios a mandar la ubicación. Es independiente el modo de elegirlos
Salida Específica	• La pantalla de inserción de ubicación se abre. • Hay un botón ("Mi Localización") para coger la ubicación de forma automática el cual se encuentra por defecto desactivado y el editor de texto está vacío.
RSF Relacionado	RSF-06, RSF-07, RSF-09

Tabla 164: PRU-11

PRU-12	
Título	Añadir ubicación de forma manual
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige usuarios a mandar ubicación • Se añade una ubicación de forma manual, sin pulsar el botón que coge la ubicación de forma manual
Salida Genérica	• La ubicación es recogida con éxito
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige usuarios a mandar la ubicación. Es independiente el modo de elegirlos 3. Se inserta una ubicación
Salida Específica	• La ubicación se muestra correctamente en el editor de texto.
RSF Relacionado	RSF-06, RSF-07, RSF-09

Tabla 165: PRU-12

PRU-13	
Título	Añadir ubicación de forma automática
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige usuarios a mandar ubicación • Se añade una ubicación de forma automática, pulsando el botón que coge la ubicación de forma manual
Salida Genérica	• La ubicación es recogida con éxito
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige usuarios a mandar la ubicación. Es independiente el modo de elegirlos 3. Se pulsa el botón "Mi Localización"
Salida Específica	• La ubicación recogida por el móvil se muestra correctamente en el editor de texto. • El botón Mi Localización" queda activado
RSF Relacionado	RSF-06, RSF-08, RSF-09

Tabla 166: PRU-13

PRU-14	
Título	Hora de encuentro por defecto es la hora actual
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige usuarios a mandar ubicación • Se comprueba la hora de encuentro
Salida Genérica	• La hora coincide con la que marca el reloj en la parte superior derecha
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige usuarios a mandar la ubicación. Es independiente el modo de elegirlos 3. Se comprueba la hora de encuentro
Salida Específica	---
RSF Relacionado	RSF-10

Tabla 167: PRU-14

PRU-15	
Título	Cambiar hora de quedar
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige usuarios a mandar ubicación • Se cambia la hora de quedar
Salida Genérica	• La hora del encuentro se establece a la hora elegida por el usuario
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige usuarios a mandar la ubicación. Es independiente el modo de elegirlos 3. Se cambia a la hora "17:00"
Salida Específica	• La hora del encuentro se establece a las "17:00" horas.
RSF Relacionado	RSF-10

Tabla 168: PRU-15

PRU-16	
Título	Tomar una foto
Entrada Genérica	• Se abre el menú mandar ubicación • Se da la información requerida para las dos primeras pantallas • Se pulsa el botón para tomar foto.
Salida Genérica	• Se activa el servicio del móvil para tomar fotos
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige usuarios a mandar la ubicación. Es independiente el modo de elegirlos 3. Se pide al sistema que ponga la ubicación actual y se deja la hora por defecto 4. Se pulsa el botón siguiente 5. Se pulsa el botón "Tomar foto"
Salida Específica	---
RSF Relacionado	RSF-11, RSF-12, RSF-13, RSF-14

Tabla 169: PRU-16

PRU-17	
Título	Mandar ubicación sin foto
Entrada Genérica	• Se abre el menú mandar ubicación • Se da la información requerida para las dos primeras pantallas • Se comprueba que el botón “ENVIAR” está activo • Se envía el encuentro sin foto
Salida Genérica	• La ubicación insertada se envía correctamente y se va a la pantalla principal. • Sale un mensaje para saber que se ha enviado correctamente.
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando “*Enviar Localización*” 2. Se elige usuarios a mandar la ubicación. Es independiente el modo de elegirlos 3. Se pide al sistema que ponga la ubicación actual y se deja la hora por defecto 4. Se pulsa el botón siguiente 5. Se pulsa el botón “ENVIAR”
Salida Específica	• La ubicación se manda correctamente yendo a la pantalla principal. • Sale el mensaje de confirmación “La ubicación ha sido enviada correctamente”
RSF Relacionado	RSF-15, RSF-16

Tabla 170: PRU-17

PRU-18	
Título	Mandar ubicación con foto
Entrada Genérica	• Se abre el menú mandar ubicación • Se da la información requerida para las dos primeras pantallas • Se realiza una foto • Se comprueba que el botón “ENVIAR” está activo • Se envía el encuentro
Salida Genérica	• La ubicación insertada se envía correctamente y se va a la pantalla principal. • Sale un mensaje para saber que se ha enviado correctamente.
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando “*Enviar Localización*” 2. Se elige usuarios a mandar la ubicación. Es independiente el modo de elegirlos 3. Se pide al sistema que ponga la ubicación actual y se deja la hora por defecto 4. Se pulsa el botón siguiente 5. Se pulsa el botón “Tomar foto” 6. Se toma una foto del lugar 7. Se comprueba que aparece en la pantalla la foto realizada 8. Se pulsa el botón “ENVIAR”
Salida Específica	• La ubicación se manda correctamente yendo a la pantalla principal. • Sale el mensaje de confirmación “La ubicación ha sido enviada correctamente”
RSF Relacionado	RSF-15

Tabla 171: PRU-18

PRU-19	
Título	Mandar una ubicación a un contacto
Entrada Genérica	• Se abre el menú mandar ubicación • Se rellena la información de las pantallas eligiendo la opción de mandar a un contacto • Se envía
Salida Genérica	• La ubicación insertada se envía correctamente y se va a la pantalla principal. • Sale un mensaje para saber que se ha enviado correctamente. • Al usuario elegido le llega la nueva ubicación informándole por una notificación.
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un contacto y se elige cual 3. Se inserta una ubicación "C/ Antonio Leyva, Nº35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 5. Se pulsa "ENVIAR"
Salida Específica	• La ubicación se manda correctamente yendo a la pantalla principal. • Sale el mensaje de confirmación "La ubicación ha sido enviada correctamente" • Al usuario le llega una notificación con la siguiente información separada por puntos y comas: o Nombre que tenga en su agenda mi contacto o C/ Antonio Leyva, Nº35, 28019, Madrid o 20:00h (Ej. Mario; C/ Antonio Leyva, Nº35, 28019, Madrid; 20:00h)
RSF Relacionado	RSF-17

Tabla 172: PRU-19

PRU-20	
Título	Mandar una ubicación a varios usuarios
Entrada Genérica	• Se abre el menú mandar ubicación • Se rellena la información de las pantallas eligiendo la opción de mandar a grupo • Se envía
Salida Genérica	• La ubicación insertada se envía correctamente y se va a la pantalla principal. • Sale un mensaje para saber que se ha enviado correctamente. • A todos los usuarios les llega la nueva ubicación informándoles por una notificación.
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un grupo y se elige cual 3. Se inserta una ubicación "C/ Antonio Leyva, Nº35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 5. Se pulsa "ENVIAR"
Salida Específica	• La ubicación se manda correctamente yendo a la pantalla principal. • Sale el mensaje de confirmación "La ubicación ha sido enviada correctamente" • Cada uno de los usuarios del grupo les llega una notificación con la siguiente información separada por puntos y comas: o Nombre que tenga en su agenda mi contacto o C/ Antonio Leyva, Nº35, 28019, Madrid o 20:00h (Ej. Mario; C/ Antonio Leyva, Nº35, 28019, Madrid ; 20:00h)
RSF Relacionado	RSF-17

Tabla 173: PRU-20

PRU-21	
Título	Comprobar que los grupos que se pueden elegir son los creados por el usuario
Entrada Genérica	• Se parte desde una situación sin grupos creados • Se abre el menú crear grupo • Se crea tres grupos diferentes • Se abre el menú mandar ubicación • Se elige la opción de mandar ubicación a un grupo • Se comprueba la lista
Salida Genérica	• Los grupos que aparecen son los mismos que se han creado, apareciendo por cada grupo: o el nombre del grupo en el primer renglón o los nombres de los usuarios en los siguientes
Entrada Específica	1. Se parte desde una situación sin grupos creados, borrando los grupos que existan desde la pantalla de opciones de grupos. 2. Se abre el menú crear grupo pulsando "*Grupo >> Crear*" 3. Se crea tres grupos diferentes: a. Universidad: con los tres primeros contactos (1º, 2º y 3º) b. Barrio: Con los dos siguientes contactos (4º y 5º) c. Colegio: Con los cuatro siguientes contactos (6º, 7º, 8º y 9º) 4. Se vuelve a la pantalla principal. 5. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 6. Se elige la opción mandar a un grupo 7. Se comprueba la lista
Salida Específica	• Los grupos son: o Universidad: ▪ 1º ▪ 2º ▪ 3º o Barrio: ▪ 4º ▪ 5º o Colegio: ▪ 6º ▪ 7º

	▪ 8º ▪ 9º
RSF Relacionado	RSF-02, RSF-17

Tabla 174: PRU-21

PRU-22	
Título	Miga de pan en mandar ubicación I: Contactos
Entrada Genérica	• Se abre el menú mandar ubicación • Se comprueba la miga de pan
Salida Genérica	• Existe la miga y tiene el formato correcto
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se comprueba la miga de pan
Salida Específica	• Existe miga de pan y tiene el formato: o Contacto: Fondo negro y letra blanca o Dirección: Fondo blanco y letra negra o Foto: Fondo blanco y letra negra
RSF Relacionado	RSF-18, RSF-19

Tabla 175: PRU-22

PRU-23	
Título	Miga de pan en mandar ubicación II: Dirección
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige a quien mandar y se abre la pantalla de insertado de dirección • Se comprueba la miga de pan
Salida Genérica	• Existe la miga y tiene el formato correcto
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige a quien mandar (es indiferente lo que se elija, por ejemplo se elije el primer grupo) y se abre la pantalla de insertado de dirección 3. Se comprueba la miga de pan
Salida Específica	• Existe miga de pan y tiene el formato: o Contacto: Fondo blanco y letra negra o Dirección: Fondo negro y letra blanca o Foto: Fondo blanco y letra negra
RSF Relacionado	RSF-18, RSF-19

Tabla 176: PRU-23

PRU-24	
Título	Miga de pan en mandar ubicación III: Foto
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige a quien mandar la ubicación • Se inserta una dirección • Se pulsa a siguiente • Se comprueba la miga de pan
Salida Genérica	• Existe la miga y tiene el formato correcto
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige a quien mandar (es indiferente lo que se elija, por ejemplo se elige el primer grupo) 3. Se inserta una dirección: C/ Oporto, 1037, 28026, Madrid 4. Se comprueba la miga de pan
Salida Específica	• Existe miga de pan y tiene el formato: o Contacto: Fondo blanco y letra negra o Dirección: Fondo blanco y letra negra o Foto: Fondo negro y letra blanca
RSF Relacionado	RSF-18, RSF-19

Tabla 177: PRU-24

PRU-25	
Título	Navegación I: Contactos
Entrada Genérica	• Se abre el menú mandar ubicación • Se comprueba la navegación
Salida Genérica	• Existe flecha para navegar a la siguiente pantalla y aparece deshabilitada
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se comprueba la navegación
Salida Específica	---
RSF Relacionado	RSF-20, RSF-21, RSF-29

Tabla 178: PRU-25

PRU-26	
Título	Navegación II: Dirección
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige a quien mandar y se abre la pantalla de insertado de dirección • Se comprueba la navegación
Salida Genérica	• Existen dos flechas para navegar a las pantallas contiguas y aparecen deshabilitadas
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige a quien mandar (es indiferente lo que se elija, por ejemplo se elige el primer grupo) y se abre la pantalla de insertado de dirección 3. Se comprueba la navegación
Salida Específica	---
RSF Relacionado	RSF-20, RSF-21, RSF-29

Tabla 179: PRU-26

PRU-27	
Título	Navegación III: Foto
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige a quien mandar la ubicación • Se inserta una dirección y se pulsa a siguiente • Se comprueba la navegación
Salida Genérica	• Existe una flecha para navegar a la pantalla anterior y aparece habilitada
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige a quien mandar (es indiferente lo que se elija, por ejemplo se elige el primer grupo) 3. Se inserta una dirección: C/ Oporto, 1037, 28026, Madrid 4. Se comprueba la navegación
Salida Específica	---
RSF Relacionado	RSF-20, RSF-21, RSF-29

Tabla 180: PRU-27

PRU-28	
Título	Botón de ENVIAR I: Contactos
Entrada Genérica	• Se abre el menú mandar ubicación • Se comprueba el botón de “ENVIAR”
Salida Genérica	• Existe el botón de “ENVIAR” y está deshabilitado
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando “*Enviar Localización*” 2. Se comprueba el botón de “ENVIAR”
Salida Específica	---
RSF Relacionado	RSF-22, RSF-29

Tabla 181: PRU-28

PRU-29	
Título	Botón de ENVIAR II: Dirección
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige a quien mandar y se abre la pantalla de insertado de dirección • Se comprueba el botón de “ENVIAR”
Salida Genérica	• Existe el botón de “ENVIAR” y está deshabilitado
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando “*Enviar Localización*” 2. Se elige a quien mandar (es indiferente lo que se elija, por ejemplo se elige el primer grupo) y se abre la pantalla de insertado de dirección 3. Se comprueba el botón de “ENVIAR”
Salida Específica	---
RSF Relacionado	RSF-22, RSF-29

Tabla 182: PRU-29

PRU-30	
Título	Botón de ENVIAR III: Foto
Entrada Genérica	• Se abre el menú mandar ubicación • Se elige a quien mandar la ubicación • Se inserta una dirección • Se pulsa a siguiente • Se comprueba el botón de “ENVIAR”
Salida Genérica	• Existe el botón de “ENVIAR” y está habilitado
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando “*Enviar Localización*” 2. Se elige a quien mandar (es indiferente lo que se elija, por ejemplo se elije el primer grupo) 3. Se inserta una dirección: C/ Oporto, 1037, 28026, Madrid 4. Se comprueba el botón de “ENVIAR”
Salida Específica	---
RSF Relacionado	RSF-22, RSF-29

Tabla 183: PRU-30

PRU-31	
Título	Información de la notificación
Entrada Genérica	• Se abre el menú mandar ubicación • Se rellena la información de las pantallas eligiendo la opción de mandar a un contacto • Se envía
Salida Genérica	• La ubicación insertada se envía correctamente y se va a la pantalla principal. • Sale un mensaje para saber que se ha enviado correctamente. • Al usuario elegido le llega la nueva ubicación informándole por una notificación con formato correcto.
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un contacto y se elige cual 3. Se inserta una ubicación "C/ Antonio Leyva, Nº35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 5. Se pulsa "ENVIAR"
Salida Específica	• La ubicación se manda correctamente yendo a la pantalla principal. • Sale el mensaje de confirmación "La ubicación ha sido enviada correctamente" • Al usuario le llega una notificación con la siguiente información separada por puntos y comas: o Nombre que tenga en su agenda mi contacto o C/ Antonio Leyva, Nº35, 28019, Madrid o 20:00h (Ej. Mario; C/ Antonio Leyva, Nº35, 28019, Madrid; 20:00h)
RSF Relacionado	RSF-23

Tabla 184: PRU-31

PRU-32	
Título	Lista de encuentros
Entrada Genérica	• Se abre el menú para ver las encuentros
Salida Genérica	• Aparece una lista con los encuentros independientemente de haber sido enviadas o recibidas
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Mis encuentros*" 2. Se comprueba la lista
Salida Específica	---
RSF Relacionado	RSF-24

Tabla 185: PRU-32

PRU-33	
Título	Información del encuentro en la lista
Entrada Genérica	• Se abre el menú mandar ubicación • Se rellena la información de las pantallas eligiendo la opción de mandar a un contacto • Se envía • Se abre el menú para ver los encuentros del usuario receptor • Se comprueba la información de la encuentro recién recibida
Salida Genérica	• El usuario tiene el encuentro con el formato correcto
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un contacto y se elige cual 3. Se inserta una ubicación "C/ Antonio Leyva, Nº35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 1. Se pulsa "ENVIAR" 2. Se abre la pantalla de ver los encuentros del usuario receptor pulsando éste en "*Mis encuentros*" 3. Se comprueba la información del encuentro recién recibida
Salida Específica	• El usuario tiene la siguiente información separada por puntos y comas: o Nombre que tenga en su agenda mi contacto o C/ Antonio Leyva, Nº35, 28019, Madrid o 20:00h (Ej. Mario; C/ Antonio Leyva, Nº35, 28019, Madrid; 20:00h)
RSF Relacionado	RSF-24

Tabla 186: PRU-33

PRU-34	
Título	Información del encuentro en la pantalla
Entrada Genérica	• Se abre el menú mandar ubicación • Se rellena la información de las pantallas eligiendo la opción de mandar a un contacto • Se envía • Se abre el menú para ver los encuentros del usuario receptor • Se selecciona el nuevo encuentro recibido • Se comprueba la información del encuentro
Salida Genérica	• El usuario tiene el encuentro con el formato correcto
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un contacto y se elige cual 3. Se inserta una ubicación "C/ Antonio Leyva, Nº35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 5. Se pulsa "ENVIAR" 6. Se abre la pantalla de ver los encuentros del usuario receptor pulsando éste en "*Mis encuentros*" 7. Se selecciona el nuevo encuentro recibido 8. Se comprueba la información del encuentro recién recibido
Salida Específica	• El usuario tiene la siguiente información: o Nombre que tenga en su agenda mi contacto o C/ Antonio Leyva, Nº35, 28019, Madrid o 20:00h o Foto enviada por el usuario del lugar del encuentro o Un botón para ver en el mapa el lugar del encuentro
RSF Relacionado	RSF-25

Tabla 187: PRU-34

PRU-35	
Título	Botón para ver la ubicación en el mapa
Entrada Genérica	• Se abre el menú mandar ubicación • Se rellena la información de las pantallas eligiendo la opción de mandar a un contacto • Se envía • Se abre el menú para ver los encuentros del usuario receptor • Se selecciona el nuevo encuentro recibido • Se comprueba la pantalla
Salida Genérica	• Existe un botón para ver la ubicación en el mapa
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un contacto y se elige cual 3. Se inserta una ubicación "C/ Antonio Leyva, Nº35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 5. Se pulsa "ENVIAR" 6. Se abre la pantalla de ver los encuentros del usuario receptor pulsando éste en "*Mis encuentros*" 7. Se selecciona el nuevo encuentro recibido 8. Se comprueba la pantalla
Salida Específica	• Existe el botón "Ver Mapa" para ver la ubicación en el mapa
RSF Relacionado	RSF-26

Tabla 188: PRU-35

PRU-36	
Título	Ver ubicación en el mapa
Entrada Genérica	• Se abre el menú mandar ubicación • Se rellena la información de las pantallas eligiendo la opción de mandar a un contacto • Se envía • Se abre el menú para ver los encuentros del usuario receptor • Se selecciona el nuevo encuentro recibido • Se pulsa el botón para ver el mapa
Salida Genérica	• Se abre el Google Maps con la dirección
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un contacto y se elige cual 3. Se inserta una ubicación "C/ Antonio Leyva, Nº35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 5. Se pulsa "ENVIAR" 6. Se abre la pantalla de ver los encuentros del usuario receptor pulsando éste en "*Mis encuentros*" 7. Se selecciona el nuevo encuentro recibido 8. Se pulsa el botón "Ver Mapa"
Salida Específica	• Se abre el Google Maps con la dirección C/ Antonio Leyva, Nº35, 28019, Madrid señalada
RSF Relacionado	RSF-27

Tabla 189: PRU-36

PRU-37	
Título	Control de inserción de usuarios en grupos
Entrada Genérica	• Se abre el menú crear grupo • Se añade un usuario cualquiera • Se añade otro usuario eligiendo el mismo usuario
Salida Genérica	• El sistema no muestra al usuario porque ya está insertado
Entrada Específica	1. Se abre la pantalla de crear grupo pulsando "*Grupos >> Crear*" 2. Se añade al usuario Mario Bordas 3. Se le da añadir, buscando a Mario Bordas
Salida Específica	• En la lista no se encuentra Mario Bordas y si la lista ya no hubiera más gente, el sistema muestra el mensaje de error "ya no tiene más amigos registrados"
RSF Relacionado	RSF-30

Tabla 190: PRU-37

PRU-38	
Título	Socket: envió de peticiones solo el cliente I
Entrada Genérica	• Se abre el menú mandar ubicación • Se rellena toda la información necesaria para enviar un encuentro • Se envía
Salida Genérica	• La comunicación la inicia el cliente
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un contacto y se elige cual 3. Se inserta una ubicación "C/ Antonio Leyva, Nº35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 5. Se pulsa "ENVIAR" 6. Se comprueba la comunicación
Salida Específica	• En la pantalla del servidor se comprueba que el servidor está escuchando y establece comunicación el cliente
RSF Relacionado	RSF-31, RSF-32, RSF-33

Tabla 191: PRU-38

PRU-39	
Título	Socket: envió de peticiones solo el cliente II
Entrada Genérica	• Se tiene el sistema en funcionamiento aunque no esté activo
Salida Genérica	• La comunicación la inicia el cliente
Entrada Específica	1. El sistema está instalado y el cliente registrado 2. Se comprueba la comunicación
Salida Específica	• En la pantalla del servidor se comprueba que el servidor está escuchando y establece comunicación el cliente para preguntar si tiene algún encuentro de alguien enviado a él.
RSF Relacionado	RSF-31, RSF-32, RSF-33

Tabla 192: PRU-39

PRU-40	
Título	Control de la BD I
Entrada Genérica	• Un usuario envía dos encuentros a otro usuario • Se comprueba la base de datos
Salida Genérica	• La Base de Datos tiene dos nuevos registros en la tabla de MEET
Entrada Específica	1. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 2. Se elige la opción mandar a un contacto y se elige cual 3. Se inserta la ubicación "C/ Antonio Leyva, 35, 28019, Madrid" 4. Se establece la hora de quedar a las 20:00h 5. Se pulsa "ENVIAR" 6. Se abre la pantalla de mandar ubicación pulsando "*Enviar Localización*" 7. Se elige la opción mandar a un contacto y se elige cual 8. Se inserta la ubicación "C/ Doctor Castelo 45, 28009, Madrid" 9. Se establece la hora de quedar a las 20:05h 10. Se pulsa "ENVIAR" 11. Se comprueba la BD
Salida Específica	• -
RSF Relacionado	RSF-34, RSF-35

Tabla 193: PRU-40

PRU-41	
Título	Control de la BD II
Entrada Genérica	• Se tiene el sistema en funcionamiento aunque no esté activo por parte del usuario receptor • Pasa el tiempo para solicitar encuentros
Salida Genérica	• La Base de Datos tiene dos registros menos en la tabla de MEET
Entrada Específica	1. El sistema está instalado y el cliente receptor registrado en espera 2. Pasa como mucho 10 segundos para solicitar nuevos encuentros
Salida Específica	• -
RSF Relacionado	RSF-34, RSF-35

Tabla 194: PRU-39

6.3 Trazabilidad RS – PRU

En este apartado se va a detallar una tabla con la trazabilidad que existe entre los requisitos de software funcionales y las pruebas diseñadas.

PRU \ RSF	RSF-01	RSF-02	RSF-03	RSF-04	RSF-05	RSF-06	RSF-07	RSF-08	RSF-09	RSF-10	RSF-11	RSF-12	RSF-13	RSF-14	RSF-15	RSF-16	RSF-17	RSF-18	RSF-19	RSF-20	RSF-21	RSF-22	RSF-23	RSF-24	RSF-25	RSF-26	RSF-27	RSF-28	RSF-29	RSF-30	RSF-31	RSF-32	RSF-33	RSF-34	RSF-35
PRU-01	X																																		
PRU-02	X																											X							
PRU-03		X																																	
PRU-04		X																										X							
PRU-05		X																										X							
PRU-06		X																										X							
PRU-07		X																										X							
PRU-08			X																																
PRU-09				X																															
PRU-10					X																														
PRU-11						X	X		X																										
PRU-12						X	X		X																										
PRU-13						X		X	X																										
PRU-14										X																									

PRU \ RSF	RSF-01	RSF-02	RSF-03	RSF-04	RSF-05	RSF-06	RSF-07	RSF-08	RSF-09	RSF-10	RSF-11	RSF-12	RSF-13	RSF-14	RSF-15	RSF-16	RSF-17	RSF-18	RSF-19	RSF-20	RSF-21	RSF-22	RSF-23	RSF-24	RSF-25	RSF-26	RSF-27	RSF-28	RSF-29	RSF-30	RSF-31	RSF-32	RSF-33	RSF-34	RSF-35
PRU-15										X																									
PRU-16											X	X	X	X																					
PRU-17															X	X																			
PRU-18															X																				
PRU-19																	X																		
PRU-20																	X																		
PRU-21		X															X																		
PRU-22																		X	X																
PRU-23																		X	X																
PRU-24																		X	X																
PRU-25																				X	X								X						
PRU-26																				X	X								X						
PRU-27																				X	X								X						
PRU-28																						X							X						

PRU \ RSF	RSF-01	RSF-02	RSF-03	RSF-04	RSF-05	RSF-06	RSF-07	RSF-08	RSF-09	RSF-10	RSF-11	RSF-12	RSF-13	RSF-14	RSF-15	RSF-16	RSF-17	RSF-18	RSF-19	RSF-20	RSF-21	RSF-22	RSF-23	RSF-24	RSF-25	RSF-26	RSF-27	RSF-28	RSF-29	RSF-30	RSF-31	RSF-32	RSF-33	RSF-34	RSF-35
PRU-29																						X							X						
PRU-30																						X							X						
PRU-31																							X												
PRU-32																								X											
PRU-33																								X											
PRU-34																									X										
PRU-35																										X									
PRU-36																											X								
PRU-37																														X					
PRU-38																															X	X	X		
PRU-39																															X	X	X		
PRU-40																																		X	X
PRU-41																																		X	X

Tabla 195: Matriz de trazabilidad RS - PRU

7 Conclusiones

Este apartado se va a dividir en varias partes. Las conclusiones tocan varios aspectos diferentes entre sí y el alumno ha querido diferenciarlos, ya que considera interesante enfocar cada conclusión en su contexto.

7.1 Conclusiones de Tecnología

En este apartado se van a detallar las conclusiones que el alumno ha obtenido a lo largo del proyecto acerca la tecnología usada:

- Android es un Sistema Operativo muy potente que permite mucha libertad a los usuarios, tanto usuarios propios de Android, como los usuarios programadores. La diversidad de aplicaciones que existen en Android permiten hacer cualquier acción, el problema es que se necesita de una gran variedad de ellas para obtener un servicio completo por parte del teléfono. También es cierto que gracias a esa diversidad los usuarios pueden elegir qué aplicaciones le vienen mejor para su uso diario y, todo ello, aumenta la competencia entre los programadores, por lo que la calidad de las aplicaciones aumenta considerablemente. Esta conclusión ha resultado ser muy importante, ya que gracias a entender la forma de pensar en la sociedad Android el diseño de la aplicación se ha realizado con la mayor calidad posible.
- Android imposibilita a su vez que las aplicaciones sean extensas. Esto se debe a que las pantallas de los dispositivos son pequeñas. Además la navegación que se puede tener en un software más orientado a un ordenador se pierde, haciendo que las aplicaciones no dispongan de una navegación mayor a tres pantallas. Con esto se quiere decir que una aplicación extensa hace perder el sentido de la acción que se está haciendo. Esto ha ayudado a realizar el diseño de la aplicación y en concreto el diseño de las pantallas de la aplicación.
- Android es un Sistema Operativo lento, es decir, el entorno de desarrollo de Android es Java y el control de recursos propio de Java es considerablemente peor en comparación con otros lenguajes de programación como puede ser C. Por ello la fluidez de los iPhone, iPad, etc., es mucho mejor que la de los Android. Ello hace pensar en la importancia de dar la sensación al usuario en que en todo momento el dispositivo está haciendo algo, aunque sea en segundo nivel, y no que se ha quedado bloqueado.
- Usar un servicio con conexión a un servidor desde Android consume mucho recurso ya que debe tener siempre una línea abierta el dispositivo con el propio servidor. La alternativa a ello es abrir una conexión cada cierto tiempo, lo que impide que se tenga la información actualizada por completo y para este proyecto no interesa.
- Usar una base de datos SQLite para los datos que se manejan hace que la administración de datos sea sencilla. Tener propiamente insertado los plugins necesarios para ello dentro de Android ayuda a la hora de programar y si el servidor es también programado en Java, usar SQLite hace que la transferencia de datos sea más directa, ya que la reutilización del código es fundamental en los proyectos.

7.2 Conclusiones de Ingeniería

Tras el desarrollo del proyecto se han podido extraer las siguientes conclusiones acerca de la ingeniería:

- Seguir una buena metodología ayuda a qué los proyectos salgan adelante, ya que desde el primer momento se analiza las dimensiones del proyecto y evalúa la factibilidad de éste.
- Para poder desarrollar este proyecto se han usado varios módulos ya programados por otros programadores, que normalmente ofrece el propio sistema Android, los cuales se han adaptado, y otros módulos han sido totalmente programados por el alumno. Mencionar que la reutilización de código, de prototipos, de patrones o incluso de ciertos diseños es un punto a favor a la ingeniería, ya que hace que sean más robustos y más seguros porque ha habido más usuarios que han probado la eficacia de lo reusado. A parte, poder reutilizar el trabajo de otra/s persona/s implica una investigación exhaustiva y bastante crítica que hace que el programador o el diseñador tenga que tener conocimientos avanzados sobre la materia.
- El modelo de ingeniería en Cascada con Retroalimentación es un buen modelo para cualquier proyecto, ayuda a salvar fallos y a analizarlos, pudiendo entender el trasfondo y en motivo de haberlos cometido. Ello hace que los trabajadores avancen día a día, a pesar de la experiencia que tengan.

7.3 Conclusiones acerca el prototipo

Tras el desarrollo del proyecto se han podido extraer las siguientes conclusiones acerca del prototipo diseñado puesto en marcha en unos pocos usuarios:

- La interacción de los usuarios entre la aplicación es rápida, saben a dónde dirigirse y entienden cada una de las opciones que se les presenta.
- La aplicación cumple con el propósito que se ha fijado.
- Tener la opción de crear grupos y manejarse con ellos les parece un concepto costoso en los primeros tratos con la aplicación, pero una vez que lo asimilan lo ven de gran utilidad y consiguen manejarse con más soltura y más eficacia entre la aplicación.
- Tener la posibilidad de ver una imagen real del sitio facilita mucho a los usuarios a localizar el lugar de encuentro, al igual que disponer el servicio de Google Maps.
- Los colores elegidos para la presentación de las pantallas les resultan agradables, aunque muchos de ellos piden algún tipo de organización de información por pestañas, ya que les ayudaría a su comprensión de las acciones, sobre todo en el apartado de enviar encuentro.
- El registro en el sistema por lo general les resulta incómodo aunque la explicación de la pantalla les relaja, ayudándoles a colaborar y entender que los datos no son para nada más que a modo de poder distribuir la información.

7.4 Conclusiones acerca del proyecto

Este proyecto es muy completo ya que maneja muchos conceptos dentro de la ingeniería y dentro de la informática. Estos son:

- La propia ingeniería que ha seguido el proyecto, desde el inicio hasta el final.
- Creación de interfaces y manejo de estas.
- Plataforma multihilos.
- Redes: en este caso sockets.
- Montar un servidor.
- Manejarse con nuevas tecnologías y aprender una nueva forma de programar, en este caso Android.
- Uso de servicios ofrecidos: localización GPS, 3G, 2G y Wifi; servicio de cámara; servicio Google Maps; servicio de notificaciones.
- Manejo de Bases de Datos.
- Manejo de ficheros.
- Control de errores.

Gracias a ello el conocimiento adquirido es un punto positivo para su desarrollo.

8 Futuras Líneas de Investigación

A lo largo del desarrollo del proyecto se han ido planteando muchas mejoras dentro de la aplicación y del modo de operar para obtener mejores resultados más satisfactorios para los usuarios finales. Estas líneas de Investigación que mejoraran en sí el producto son:

- En un principio el proyecto pensado fue poder tener un calculador de rutas que tuvieran en cuenta el desnivel del terreno consiguiendo rutas más cómodas para los usuarios. En caso de desarrollar dicho proyecto se le puede incluir para decir a cada usuario que recibe un encuentro, cuál es su camino más sencillo para llegar al destino.
- Otra mejora podría haber sido llamar al servicio de Google Latitude en el que dice dónde se encuentran actualmente las personas a las que les envías un encuentro, así podrías calcular el tiempo estimado que tardarán en llegar. Es cierto que ello podría asaltar ciertas políticas de privacidad de datos, por lo que se le tendría que preguntar a los usuarios si permiten el uso de dicho módulo.
- Para mejorar la interfaz, podrían hacerse ciertas encuestas y entrevistas a grupos de personas presentándoles varios prototipos. De esta forma nos ayudarían a tener una interfaz más cómoda y útil para los usuarios finales.
- Si se adaptara lo suficiente se podría conseguir también un proyecto que gestionara eventos entre los usuarios, pero quitando la dificultad y lo incómodo de tiene Facebook y Tuenti.
- Se podría hacer registros y análisis de la información que se maneja. A pesar de que este uso en muchos de los casos es ilegal si el usuario no da permiso, es cierto que el estudio puede ser muy interesante y podría ayudar a la sociedad a la hora de crear nuevos sitios de ocio, de restaurantes, de comercio, de hostelería, etc.

9 Bibliografía

- *http://es.wikipedia.org/wiki/Android*
- *http://es.wikipedia.org/wiki/Android*
- *http://es.wikipedia.org/wiki/NetBeans*
- *http://androideity.com/2011/07/03/que-es-android/*
- *http://androideity.com/2011/07/04/arquitectura-de-android/*
- *http://www.java.com/es/download/*
- *http://www.java.com/es/about/*
- *http://es.wikipedia.org/wiki/Socket_de_Internet*
- *Altova UModel*
- *http://es.wikipedia.org/wiki/Java_(lenguaje_de_programación)*

10 Anexo I: Planificación y Presupuesto

10.1 Planificación

Como podemos ver en las tablas e imágenes siguientes, el proyecto se comenzó el día 5 de Septiembre del 2011 y ha finalizado el día 21 de Agosto del 2012. Las horas totales empleadas para la realización del proyecto han sido 730. Aun así se debe señalar que durante estas fechas no ha existido una jornada de trabajo del todo estable, ya que por motivos laborales no siempre se ha podido dedicar el mismo tiempo al proyecto. En ocasiones unas semanas se empleaba una jornada de cuatro horas al día para su realización y en cambio en otras semanas se empleaban más horas o era imposible realizar tarea alguna.

En la siguiente tabla podemos ver las distintas fases en que se divide el proyecto realizado, las fechas de inicio y fin de cada una de las fases, así como las horas de trabajo destinadas a ellas.

Código	Fase	Comienzo	Fin	Número de horas
1	Investigación previa y Estado del Arte	Lunes, 05/09/2011	Viernes, 11/11/2011	210
2	Análisis	Lunes, 14/11/2011	Viernes, 25/11/2011	45
3	Diseño	Lunes, 28/11/2011	Viernes, 30/12/2011	65
4	Desarrollo, Implementación	Lunes, 02/01/2012	Viernes, 25/05/2012	220
5	Implantación y Pruebas	Lunes, 28/05/2012	Viernes, 15/06/2012	60
6	Documentación	Lunes, 12/09/2011	Viernes, 21/08/2012	130
TOTAL				730

Tabla 196: Fases en las que se subdivide el proyecto

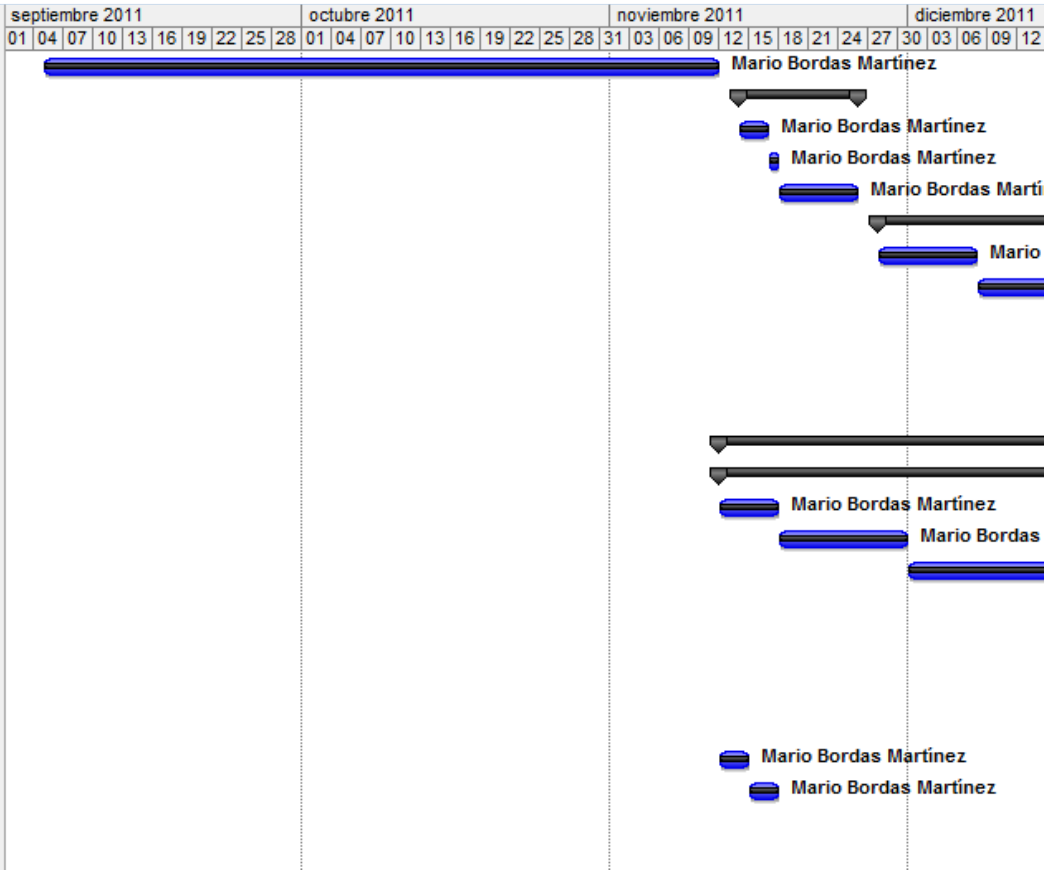
Como se ve a continuación, estas fases contienen a su vez las diferentes tareas necesarias para realizar el proyecto.

Código	Tarea	Comienzo	Fin
1	**Investigación previa y estado del arte**	**05/09/2011**	**11/11/2011**
2	**Análisis**	**14/11/2011**	**25/11/2011**
2.1	Requisitos de Usuario	14/11/2011	16/11/2011
2.2	Casos de uso	17/11/2011	17/11/2011
2.3	Requisitos de Software	18/11/2011	25/11/2011
3	**Diseño**	**28/11/2011**	**30/12/2011**
3.1	Arquitectura	28/11/2011	07/12/2011
3.2	Diseño Detallado	08/12/2011	30/12/2011
4	**Desarrollo, implementación**	**02/01/2012**	**25/05/2012**
4.1	Implementación del Servidor	02/01/2012	31/01/202
4.2	Implementación del Cliente	01/02/2012	25/05/2012
5	**Implantación y Pruebas**	**28/05/2012**	**31/07/2012**
6	**Documentación**	**12/11/2011**	**21/08/2012**
6.1	Elaboración de la memoria del proyecto	12/11/2011	17/08/2012
6.1.1	Introducción y objetivos	12/11/2011	17/11/2011
6.1.2	Estado del Arte	18/11/2011	30/11/2011
6.1.3	Análisis	01/12/2011	25/12/2011
6.1.4	Diseño	26/12/2011	26/01/2012
6.1.5	Implantación	30/05/2012	17/06/2012
6.1.6	Pruebas	18/06/2012	31/07/2012
6.1.7	Conclusiones	01/08/2012	08/08/2012
6.1.8	Futuras Líneas de Investigación	09/08/2012	11/08/2012
6.1.9	Planificación	12/11/2011	14/11/2011
6.1.10	Presupuesto	15/11/2011	17/11/2011
6.1.11	Manual de Usuario	12/08/2012	17/08/2012
6.2	Otros aspectos de la memoria a revisar y corregir	18/08/2012	21/08/2012

Tabla 197: Tareas en las que se subdivide el proyecto

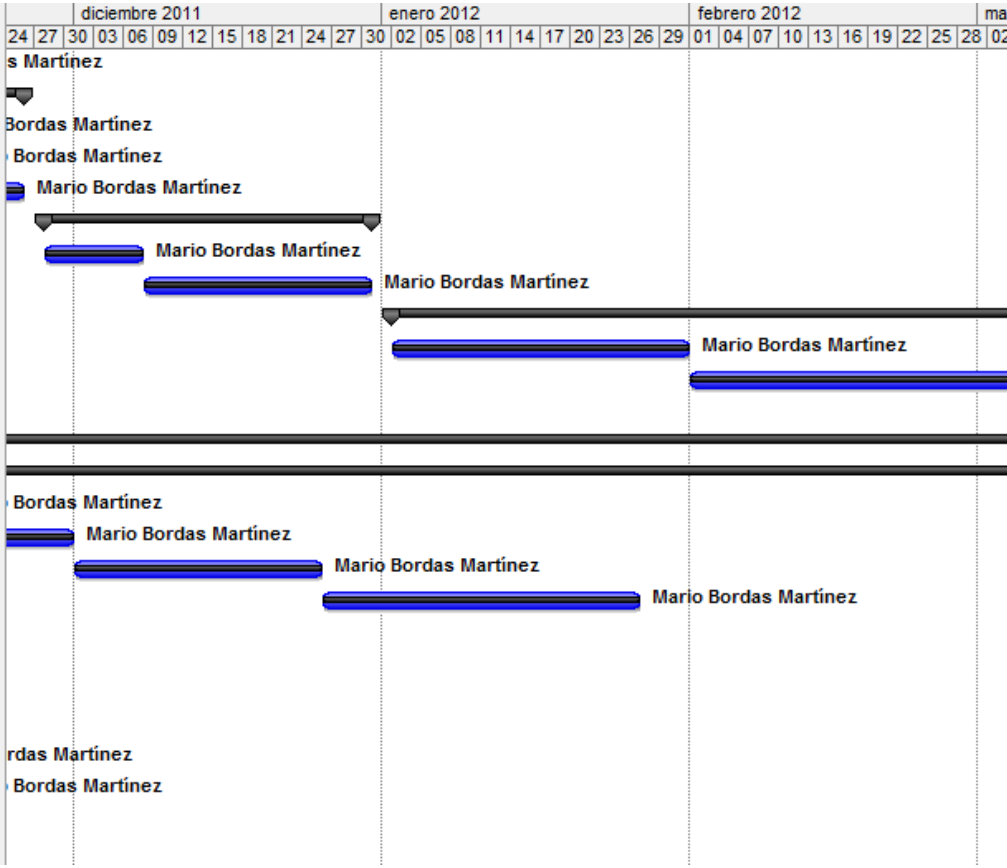
**Nótese que no se trabaja un número concreto de horas cada día, sino que realiza el número de horas previamente definido en cada fase entre la fecha de comienzo y la fecha de fin que se muestran en la tabla anterior.*

	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
1	Investigación previa y estado del arte	lun 05/09/11	vie 11/11/11	Mario Bordas Martínez
2	**Análisis**	**lun 14/11/11**	**vie 25/11/11**	**Mario Bordas Martínez**
3	Requisitos de Usuario	lun 14/11/11	mié 16/11/11	Mario Bordas Martínez
4	Casos de Uso	jue 17/11/11	jue 17/11/11	Mario Bordas Martínez
5	Requisitos de Software	vie 18/11/11	vie 25/11/11	Mario Bordas Martínez
6	**Diseño**	**lun 28/11/11**	**vie 30/12/11**	**Mario Bordas Martínez**
7	Arquitectura	lun 28/11/11	mié 07/12/11	Mario Bordas Martínez
8	Diseño Detallado	jue 08/12/11	vie 30/12/11	Mario Bordas Martínez
9	**Desarrollo, implementación**	**lun 02/01/12**	**vie 25/05/12**	**Mario Bordas Martínez**
10	Implementación del Servidor	lun 02/01/12	mar 31/01/12	Mario Bordas Martínez
11	Implementación del Cliente	mié 01/02/12	vie 25/05/12	Mario Bordas Martínez
12	Implantación y Pruebas	lun 28/05/12	mar 31/07/12	Mario Bordas Martínez
13	**Documentación**	**sáb 12/11/11**	**mar 21/08/12**	**Mario Bordas Martínez**
14	**Elaboración de la memoria del proyecto**	**sáb 12/11/11**	**vie 17/08/12**	**Mario Bordas Martínez**
15	Introducción y objetivos	sáb 12/11/11	jue 17/11/11	Mario Bordas Martínez
16	Estado del Arte	vie 18/11/11	mié 30/11/11	Mario Bordas Martínez
17	Análisis	jue 01/12/11	dom 25/12/11	Mario Bordas Martínez
18	Diseño	lun 26/12/11	jue 26/01/12	Mario Bordas Martínez
19	Implantación	mié 30/05/12	dom 17/06/12	Mario Bordas Martínez
20	Pruebas	lun 18/06/12	mar 31/07/12	Mario Bordas Martínez
21	Conclusiones	mié 01/08/12	mié 08/08/12	Mario Bordas Martínez
22	Futuras Líneas de Investigación	jue 09/08/12	sáb 11/08/12	Mario Bordas Martínez
23	Planificación	sáb 12/11/11	lun 14/11/11	Mario Bordas Martínez
24	Presupuesto	mar 15/11/11	jue 17/11/11	Mario Bordas Martínez
25	Manual de Usuario	dom 12/08/12	vie 17/08/12	Mario Bordas Martínez
26	Otros aspectos de la memoria a revisar y corregir	sáb 18/08/12	mar 21/08/12	Mario Bordas Martínez



Ilustracion 23. Diagrama de Gantt en el mes de Septiembre, Octubre y Noviembre del 2011

	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
1	Investigación previa y estado del arte	lun 05/09/11	vie 11/11/11	Mario Bordas Martínez
2	**Análisis**	**lun 14/11/11**	**vie 25/11/11**	**Mario Bordas Martínez**
3	Requisitos de Usuario	lun 14/11/11	mié 16/11/11	Mario Bordas Martínez
4	Casos de Uso	jue 17/11/11	jue 17/11/11	Mario Bordas Martínez
5	Requisitos de Software	vie 18/11/11	vie 25/11/11	Mario Bordas Martínez
6	**Diseño**	**lun 28/11/11**	**vie 30/12/11**	**Mario Bordas Martínez**
7	Arquitectura	lun 28/11/11	mié 07/12/11	Mario Bordas Martínez
8	Diseño Detallado	jue 08/12/11	vie 30/12/11	Mario Bordas Martínez
9	**Desarrollo, implementación**	**lun 02/01/12**	**vie 25/05/12**	**Mario Bordas Martínez**
10	Implementación del Servidor	lun 02/01/12	mar 31/01/12	Mario Bordas Martínez
11	Implementación del Cliente	mié 01/02/12	vie 25/05/12	Mario Bordas Martínez
12	Implantación y Pruebas	lun 28/05/12	mar 31/07/12	Mario Bordas Martínez
13	**Documentación**	**sáb 12/11/11**	**mar 21/08/12**	**Mario Bordas Martínez**
14	**Elaboración de la memoria del proyecto**	**sáb 12/11/11**	**vie 17/08/12**	**Mario Bordas Martínez**
15	Introducción y objetivos	sáb 12/11/11	jue 17/11/11	Mario Bordas Martínez
16	Estado del Arte	vie 18/11/11	mié 30/11/11	Mario Bordas Martínez
17	Análisis	jue 01/12/11	dom 25/12/11	Mario Bordas Martínez
18	Diseño	lun 26/12/11	jue 26/01/12	Mario Bordas Martínez
19	Implantación	mié 30/05/12	dom 17/06/12	Mario Bordas Martínez
20	Pruebas	lun 18/06/12	mar 31/07/12	Mario Bordas Martínez
21	Conclusiones	mié 01/08/12	mié 08/08/12	Mario Bordas Martínez
22	Futuras Líneas de Investigación	jue 09/08/12	sáb 11/08/12	Mario Bordas Martínez
23	Planificación	sáb 12/11/11	lun 14/11/11	Mario Bordas Martínez
24	Presupuesto	mar 15/11/11	jue 17/11/11	Mario Bordas Martínez
25	Manual de Usuario	dom 12/08/12	vie 17/08/12	Mario Bordas Martínez
26	Otros aspectos de la memoria a revisar y corregir	sáb 18/08/12	mar 21/08/12	Mario Bordas Martínez

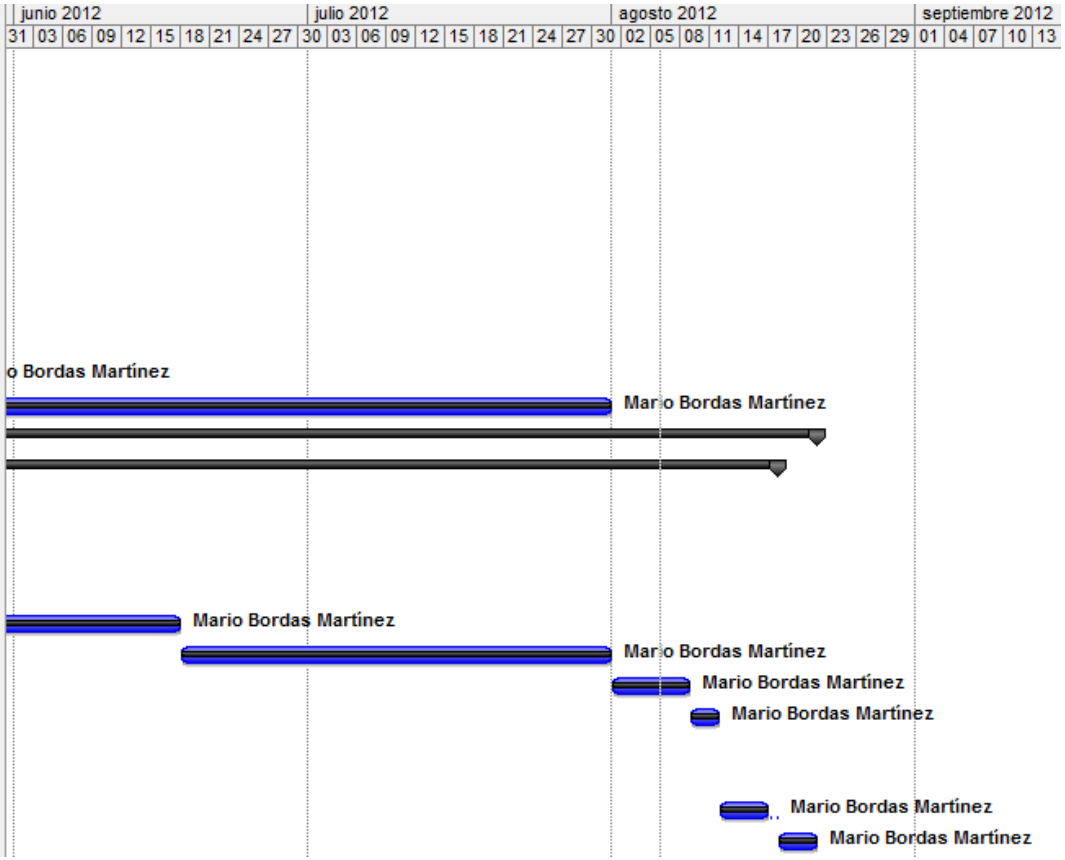


Ilustracion 24. Diagrama de Gantt en el mes de Diciembre del 2011, Enero y Febrero del 2012

	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
1	Investigación previa y estado del arte	lun 05/09/11	vie 11/11/11	Mario Bordas Martínez
2	**Análisis**	**lun 14/11/11**	**vie 25/11/11**	**Mario Bordas Martínez**
3	Requisitos de Usuario	lun 14/11/11	mié 16/11/11	Mario Bordas Martínez
4	Casos de Uso	jue 17/11/11	jue 17/11/11	Mario Bordas Martínez
5	Requisitos de Software	vie 18/11/11	vie 25/11/11	Mario Bordas Martínez
6	**Diseño**	**lun 28/11/11**	**vie 30/12/11**	**Mario Bordas Martínez**
7	Arquitectura	lun 28/11/11	mié 07/12/11	Mario Bordas Martínez
8	Diseño Detallado	jue 08/12/11	vie 30/12/11	Mario Bordas Martínez
9	**Desarrollo, implementación**	**lun 02/01/12**	**vie 25/05/12**	**Mario Bordas Martínez**
10	Implementación del Servidor	lun 02/01/12	mar 31/01/12	Mario Bordas Martínez
11	Implementación del Cliente	mié 01/02/12	vie 25/05/12	Mario Bordas Martínez
12	Implantación y Pruebas	lun 28/05/12	mar 31/07/12	Mario Bordas Martínez
13	**Documentación**	**sáb 12/11/11**	**mar 21/08/12**	**Mario Bordas Martínez**
14	**Elaboración de la memoria del proyecto**	**sáb 12/11/11**	**vie 17/08/12**	**Mario Bordas Martínez**
15	Introducción y objetivos	sáb 12/11/11	jue 17/11/11	Mario Bordas Martínez
16	Estado del Arte	vie 18/11/11	mié 30/11/11	Mario Bordas Martínez
17	Análisis	jue 01/12/11	dom 25/12/11	Mario Bordas Martínez
18	Diseño	lun 26/12/11	jue 26/01/12	Mario Bordas Martínez
19	Implantación	mié 30/05/12	dom 17/06/12	Mario Bordas Martínez
20	Pruebas	lun 18/06/12	mar 31/07/12	Mario Bordas Martínez
21	Conclusiones	mié 01/08/12	mié 08/08/12	Mario Bordas Martínez
22	Futuras Líneas de Investigación	jue 09/08/12	sáb 11/08/12	Mario Bordas Martínez
23	Planificación	sáb 12/11/11	lun 14/11/11	Mario Bordas Martínez
24	Presupuesto	mar 15/11/11	jue 17/11/11	Mario Bordas Martínez
25	Manual de Usuario	dom 12/08/12	vie 17/08/12	Mario Bordas Martínez
26	Otros aspectos de la memoria a revisar y corregir	sáb 18/08/12	mar 21/08/12	Mario Bordas Martínez

Ilustracion 25. Diagrama de Gantt en el mes de Marzo, Abril y Mayo del 2012

	Nombre de tarea	Comienzo	Fin	Nombres de los recursos
1	Investigación previa y estado del arte	lun 05/09/11	vie 11/11/11	Mario Bordas Martínez
2	**Análisis**	**lun 14/11/11**	**vie 25/11/11**	**Mario Bordas Martínez**
3	Requisitos de Usuario	lun 14/11/11	mié 16/11/11	Mario Bordas Martínez
4	Casos de Uso	jue 17/11/11	jue 17/11/11	Mario Bordas Martínez
5	Requisitos de Software	vie 18/11/11	vie 25/11/11	Mario Bordas Martínez
6	**Diseño**	**lun 28/11/11**	**vie 30/12/11**	**Mario Bordas Martínez**
7	Arquitectura	lun 28/11/11	mié 07/12/11	Mario Bordas Martínez
8	Diseño Detallado	jue 08/12/11	vie 30/12/11	Mario Bordas Martínez
9	**Desarrollo, implementación**	**lun 02/01/12**	**vie 25/05/12**	**Mario Bordas Martínez**
10	Implementación del Servidor	lun 02/01/12	mar 31/01/12	Mario Bordas Martínez
11	Implementación del Cliente	mié 01/02/12	vie 25/05/12	Mario Bordas Martínez
12	Implantación y Pruebas	lun 28/05/12	mar 31/07/12	Mario Bordas Martínez
13	**Documentación**	**sáb 12/11/11**	**mar 21/08/12**	**Mario Bordas Martínez**
14	**Elaboración de la memoria del proyecto**	**sáb 12/11/11**	**vie 17/08/12**	**Mario Bordas Martínez**
15	Introducción y objetivos	sáb 12/11/11	jue 17/11/11	Mario Bordas Martínez
16	Estado del Arte	vie 18/11/11	mié 30/11/11	Mario Bordas Martínez
17	Análisis	jue 01/12/11	dom 25/12/11	Mario Bordas Martínez
18	Diseño	lun 26/12/11	jue 26/01/12	Mario Bordas Martínez
19	Implantación	mié 30/05/12	dom 17/06/12	Mario Bordas Martínez
20	Pruebas	lun 18/06/12	mar 31/07/12	Mario Bordas Martínez
21	Conclusiones	mié 01/08/12	mié 08/08/12	Mario Bordas Martínez
22	Futuras Líneas de Investigación	jue 09/08/12	sáb 11/08/12	Mario Bordas Martínez
23	Planificación	sáb 12/11/11	lun 14/11/11	Mario Bordas Martínez
24	Presupuesto	mar 15/11/11	jue 17/11/11	Mario Bordas Martínez
25	Manual de Usuario	dom 12/08/12	vie 17/08/12	Mario Bordas Martínez
26	Otros aspectos de la memoria a revisar y corregir	sáb 18/08/12	mar 21/08/12	Mario Bordas Martínez



Ilustracion 26. Diagrama de Gantt en el mes de Junio, Julio y Agosto del 2012

10.2 Presupuesto

En este apartado se detallará un presupuesto que cubra todos los costes y las necesidades que se irán planteando a lo largo de la evolución del proyecto que están relacionados con éste.

10.2.1 Coste de Personal

En este punto se va a detallar los costes de personal que supone el proyecto a desarrollar. Respecto al coste de personal, supuestamente el trabajador que ha desarrollado el proyecto no ha cobrado nada por el dicho trabajo, pero a modo de ilustrar un presupuesto lo más real posible se ha considerado que tenga un precio dicho trabajo. En cambio el beneficio se considera nulo, ya que el proyecto es lucrativo y meter un porcentaje al beneficio no modifica el cálculo del presupuesto.

Este proyecto solo lo ha realizado una persona desde el inicio del análisis hasta las pruebas de evaluación del software. Por ello el precio de la persona será siempre el mismo, pero se detallará la parte que corresponde a cada una de las partes a modo de que quede reflejado lo más claro posible el desglose del presupuesto.

Los roles que se van a definir, dependiendo de la fase del desarrollo, serán: Analista, Diseñador, Programador, Ingeniero de Pruebas. El coste/Hora dependería del rol que desempeñara la persona que hace dicha función, pero en este caso , al ser la misma persona, se cogerá un coste igual para todos ya que en una empresa se paga el mismo salario a la misma persona independientemente de la fase que desarrolle, claro está sin contar con la responsabilidad asociada. El coste de la persona vendrá estimado en un coste bruto.

El precio que se le pagará al ingeniero por mes será lo que se suele pagar a un ingeniero informático sin experiencia, que es alrededor de 1.200 €/mes netos lo que saldría, aplicando un 25% de incremento por seguridad social e impuestos de IRPF, unos 1600 €/mes brutos.

De este modo el coste del empleado por hora será se calculará a partir de las horas mensuales trabajadas. Sabiendo que trabaja una media de 4 horas diarias y que, al ser estudiante, trabaja también fin de semana, el número de horas mensuales del ingeniero serán:

30*4 = 120 horas mensuales.

El coste por hora por tanto será:

1600 €/mes % 120 horas/mes = 13,33 €/hora.

A continuación se detalla una tabla con los roles desempeñados y su coste asociado a su trabajo a realizar.

Categoría	Coste/Hora (€)
Analista	13,33 €
Diseñador	13,33 €
Programador	13,33 €
Ingeniero de Pruebas	13,33 €

Tabla 198: Coste por puesto

El tiempo estimado para cada uno de los roles queda definido en la planificación, por lo que el sueldo por fase se calculará:

Coste_ingeniero (€/horas) * horas_fase (horas)

Las horas trabajadas en cada fase han salido de la planificación, distribuyendo el tiempo de documentación entre los roles en los que se documenta, es decir:

- **Analista:**
  - Investigación previa y Estado del Arte + Análisis + Documentación = 210 + 45 + 45 = 300 horas para el rol de analista.
- **Diseñador:**
  - Diseño + Documentación = 65 + 45 = 110 horas para el rol de diseñador.
- **Programador:**
  - Desarrollo, Implementación = 220 horas para el rol de programador.
- **Ingeniero de Pruebas:**
  - Implantación y Pruebas + Documentación = 60 + 40 = 100 horas para el rol de ingeniero de pruebas.

Los resultados obtenidos son los siguientes:

Rol	€ Brutos / hora	Horas del rol	Coste
Analista	13,33	300	3.999,00 €
Diseñador	13,33	110	1.466,30 €
Programador	13,33	220	2.932,60 €
Ingeniero de Pruebas	13,33	100	1.333,00 €
TOTAL			**9.730,90 €**

Tabla 199: Costes de cada miembro en la fase de Planificación.

10.2.2 Coste de Equipos

En este apartado detallaremos los costes asociados a los equipos informáticos que se utilizarán en el desarrollo del proyecto.

Descripción	Coste/ Unidad	Unidades	Coste total	Periodo de amortización (meses)	Duración proyecto (meses)	Coste asociado proyecto
Toshiba Satellite A660-13T	795	1	795	24	4	132,50 €
Servidor HP ProLiant ML310 G5p Server series	670,92	1	670,92	24	4	111,82 €
SAI 550 V,A Riello Pgd600 Plug Dialog	75,3	1	75,3	24	4	12,55 €
LOGITECH B110 Optical USB Mouse	9,65	1	9,65	24	4	1,61 €
Impresora HP LASERJET P1102	99,99	1	99,99	24	4	16,67 €
TOTAL						**275,14 €**

Tabla 200: Coste de equipos.

10.2.3 Coste de Software

En esta tabla se muestran los costes asociados a las licencias de los programas que se utilizarán durante todo el desarrollo del proyecto. Al igual que en el apartado anterior se tienen en cuenta los programas necesarios para llevar a cabo el análisis, el diseño y la implementación.

Descripción	Coste/ Unidad	Unidades	Coste total	Periodo de amortización (meses)	Duración proyecto (meses)	Coste asociado proyecto
Windows 7 Home Premium	0	1	0	24	4	0,00 €
Office Project Standard 2007	599,95	1	599,95	24	4	99,99 €
Office Visio Standard 2007	259,95	1	259,95	24	4	43,33 €
Altova UModel 2009 Enterprise Edition	139,8	1	139,8	24	4	23,30 €
TOTAL						**166,62 €**

Tabla 201: Coste de licencias de software

10.2.4 Coste Hardware

En este apartado se detalla los costes asociados al Hardware necesarios para el desarrollo del proyecto. En este caso no es necesario ningún hardware externo por lo que se refleja que se ha tenido en cuenta, pero que su coste es 0 €.

10.2.5 Coste Material Fungible

En este apartado se detallan los costes asociados a la utilización de material fungible que será necesario en el desarrollo. Principalmente estos costes serán sobre material de papelería básico, en el que no se puede ajustar totalmente el coste, pero se hará una estimación lo más ajustada posible.

Descripción	Coste/ Unidad	Unidades	Coste total
Marial de papelería.	N/A	N/A	120,00 €
HP Toner Black Laserjet P1505/1505N	37,54	1	37,54 €
TOTAL			**157,54 €**

Tabla 202: Coste Material fungible

10.2.6 Coste por Viajes y Dietas

En esta tabla se muestran los costes asociados a los desplazamientos para hablar con el cliente y con las personas necesarias para desarrollar el proyecto.

En este caso no existe un desplazamiento por lo que el coste será 0 €.

10.2.7 Costes Directos Totales

A continuación detallamos el coste del presupuesto total para saber cuánto cuesta cada una de las partes y lo que nos costaría el proyecto en sí:

Descripción	Coste total
Personal	9.730,90 €
Equipos	275,14 €
Software	166,62 €
Hardware	0 €
Material Fungible	157,54 €
Viajes y dietas	0 €
TOTAL	**10.172,66 €**

Tabla 203: Coste total de costes directos.

10.2.8 Costes Indirectos

En nuestro caso hemos estimado que tendremos un 5% de costes indirectos sobre los costes directos, por lo que nos da los siguientes costes:

Gastos Totales	Margen de Costes Indirectos	Costes Indirectos
10.172,66 €	5,00%	**508,63 €**

Tabla 204: Costes Indirectos

10.2.9 Beneficio

Para este proyecto no se quiere obtener beneficio, por ello el beneficio es del 0%, ya que es un proyecto lucrativo con el que no se obtienen beneficios. El motivo de ello es que es un proyecto de fin de carrera y no se puede obtener beneficio de ello. La estimación de la ganancia por consiguiente será la siguiente:

Gastos Totales	Margen de Costes Indirectos	Costes Indirectos
10.172,66 €	0,00%	**0 €**

Tabla 205: Beneficio

10.2.10 Margen de Riesgo

Para nuestro proyecto hemos calculado un 10% de riesgo, ya que este proyecto es sencillo y los costes asociados no son muy escandalosos. Normalmente el riesgo se suele situar en un 20%, pero como se acaba de comentar en este proyecto no es necesario tener un riesgo tan elevado. Por tanto, el coste de riesgo será la siguiente:

Gastos Totales	Margen riesgo	Total de riesgos
10.172,66 €	10%	**1.017,26 €**

Tabla 206: Margen de riesgo

10.2.11 Presupuesto Total

El presupuesto total sin IVA que nos da del proyecto que queremos desarrollar es el siguiente en el que se detallan cada uno de los conceptos:

Descripción	Euros
Costes Directos	10.172,66 €
Costes Indirectos	508,63 €
Beneficio	0 €
Riesgos	1.017,26 €
TOTAL	**11.698,55 €**

Tabla 207: Presupuesto total sin IVA

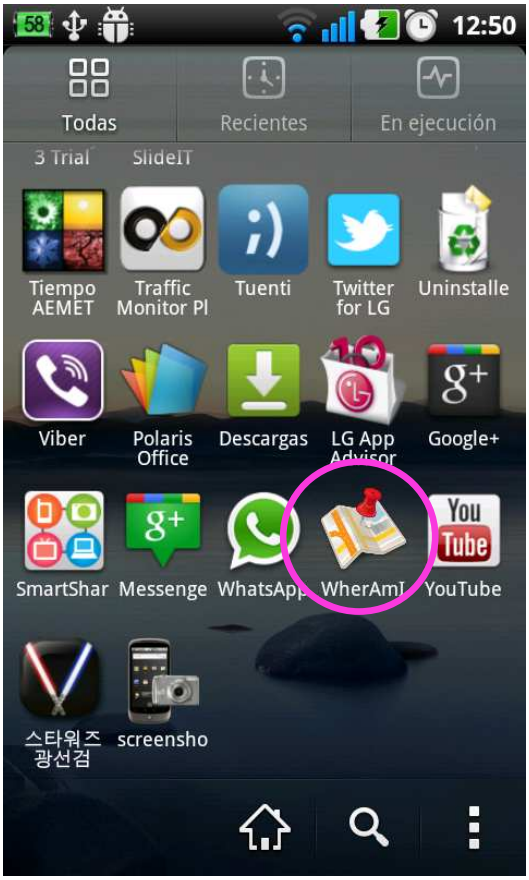
Si aplicamos el IVA que es de un 21%, el coste del proyecto real será de catorce mil ciento cincuenta y cinco euros con veinticinco céntimos (**14.155,25 €**).

11 Anexo II: Manual de Usuario

En este capítulo se detallará un manual para el usuario que muestre el uso correcto de la aplicación.

11.1 Acceso inicial

Se debe acceder al menú de aplicaciones del dispositivo y localizar el icono de la aplicación llamado *WhereAmI* tal y como se muestra en la siguiente imagen.



Ilustracion 27. Acceso inicial

Se debe pinchar sobre el icono. Tras esto se mostrará una pantalla de registro del usuario.

11.2 Registro

En la siguiente pantalla se muestra el campo correspondiente al número de teléfono del usuario que éste debe introducir para validar el registro.

Ilustracion 28. Registro

11.3 Pantalla principal – Opciones

Una vez registrado el usuario, se accede a la siguiente pantalla:

Ilustracion 29. Pantalla principal – Opciones 1

Se debe mencionar que el botón correspondiente a *Mis encuentros* aparecerá inhabilitado en caso de que el usuario no tenga ningún encuentro definido, y habilitado cuando el usuario ya tenga encuentros registrados. En un principio, tal y como se muestra en la imagen anterior, el usuario no posee encuentros; pero si por el contrario ya tuviese alguno registrado, la pantalla principal tendría el aspecto siguiente:

Ilustracion 30. Pantalla principal – Opciones 2

En ella se pueden seleccionar tres opciones diferentes que se detallarán a continuación.

11.4 Grupos

Ilustracion 31. Administración de grupos

Se pueden crear grupos de personas. Estos grupos permitirán al usuario enviar *encuentros* sin necesidad de añadir una a una cada persona al evento en creación. Más tarde, en el apartado 1.5, se mostrará claramente su funcionalidad.

Nótese una vez más que si el usuario no tiene ningún grupo creado, el botón *Borrar* permanece inactivo hasta cree alguno.

Dentro de la pantalla de administración de grupos se pueden crear o suprimir dichos grupos. Veamos a continuación cómo.

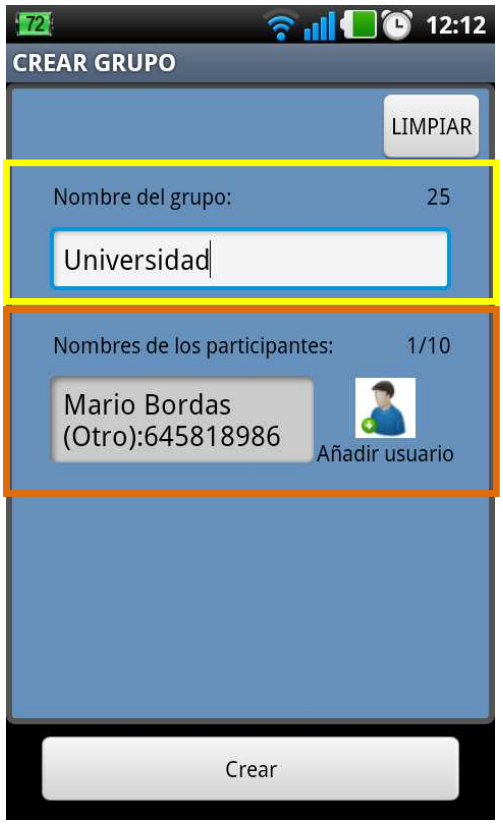
11.4.1 Crear

Ilustracion 32. Creación de grupo

Para crear un grupo el usuario debe presionar el botón pertinente y rellenar los siguientes campos:

- El nombre con el que quiere referirse al grupo, señalado con color amarillo. Dicho nombre no debe exceder los veinticinco caracteres, tal y como indica el número que aparece sobre la caja de texto.
- Los contactos que quiere que formen parte del grupo, señalado con color naranja. Se debe destacar que:
    - Tiene un máximo de diez, el cual se indica sobre el icono de *Añadir usuario* y que disminuye a medida que se añaden contactos.
    - Existe un botón que lleva a la agenda del usuario donde éste seleccionará el contacto que desea añadir al grupo.
    - Existe una caja de texto donde aparecen los contactos seleccionado previamente.

A continuación se muestra un ejemplo que ilustra lo anteriormente explicado:



Ilustracion 33. Creación de grupo (ejemplo)

11.4.2 Borrados

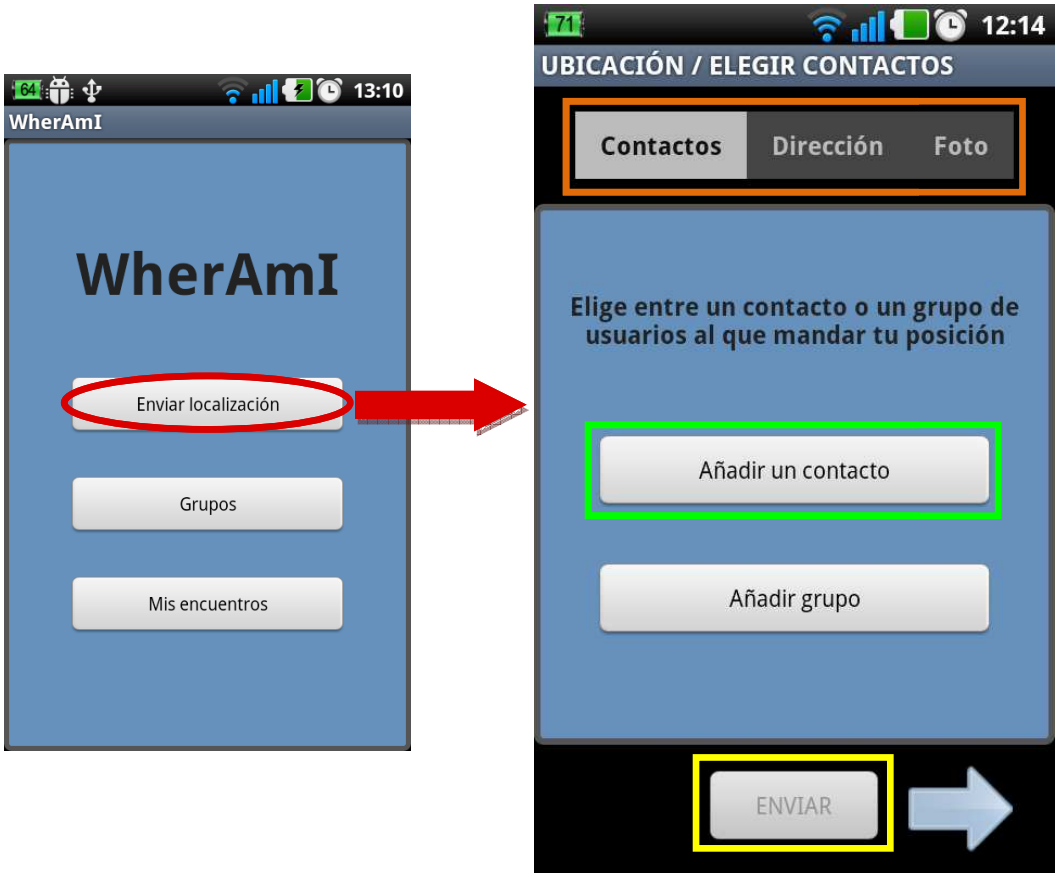
Una vez el usuario haya creado algún grupo, la aplicación habilitará, tal y como se muestra en la siguiente imagen, el botón *Borrar.*

Para borrar un grupo el usuario debe presionar el botón pertinente, tras esto la aplicación muestra al usuario la lista de grupos existentes, donde el usuario seleccionará el grupo que desea borrar.

Ilustracion 34. Supresión de grupo

Tras pulsar sobre dicho grupo, la el grupo se borra.

11.5 Enviar localización



Ilustracion 35. Envío de localización

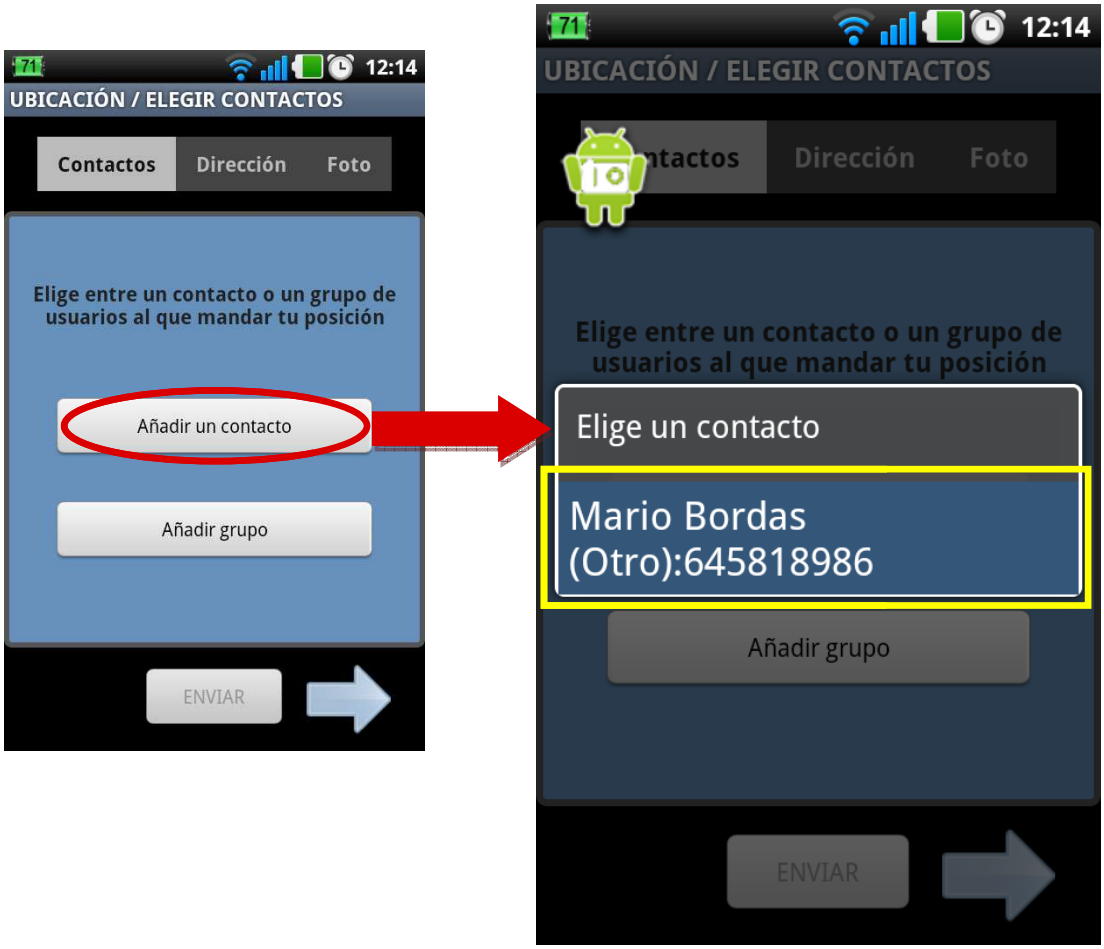
Para enviar la localización donde se quiere llevar a cabo el encuentro, el usuario deberá pulsar sobre el botón *Enviar localización*, tras esto la aplicación muestra la pantalla que se muestra en la imagen anterior. Para enviar toda la información relativa al encuentro es preciso que el usuario rellene cierta información contenida en tres pestañas diferentes: *Contactos*, *Dirección* y *Foto*.

Se comienza rellenando el/los contacto/s a los que el usuario quiere enviar la información. Para ello puede añadir un único contacto o un grupo de ellos, cosa que se detalla en los dos siguientes apartado.

Nótese que el botón *Enviar* está deshabilitado, pues aún no se ha rellenado toda la información necesaria.

11.5.1 Contactos

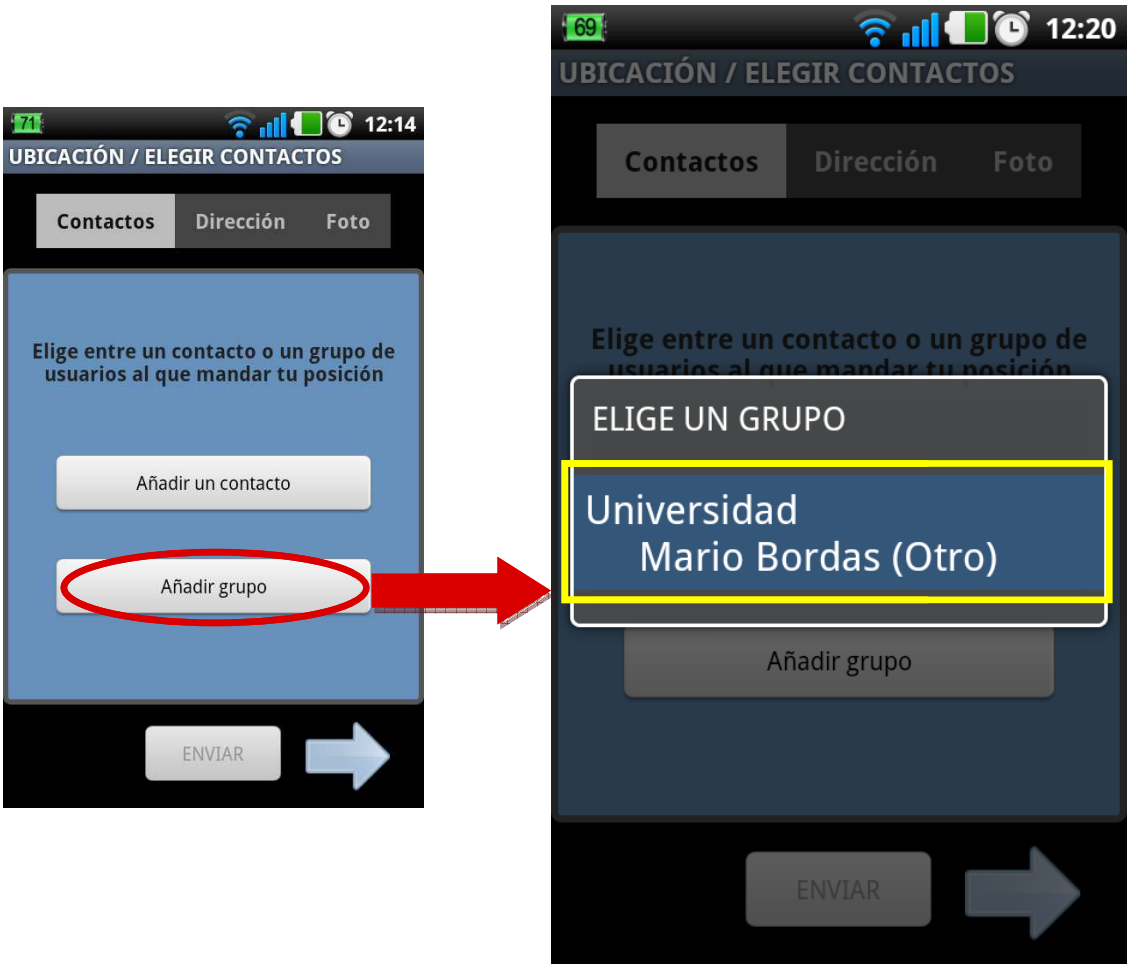
11.5.1.1 Añadir un contacto



Ilustracion 36. Adición de un contacto

Al pulsar el botón *Añadir un contacto*, se muestra la lista de contactos que el usuario tiene en el dispositivo. De dicha lista, el usuario debe seleccionar uno de ellos y pinchar en él. De esta forma, el contacto al que enviarle la información queda definido y la aplicación pasaría a la siguiente pantalla, correspondiente con la *Dirección*, sin necesidad de que el usuario haga nada más.

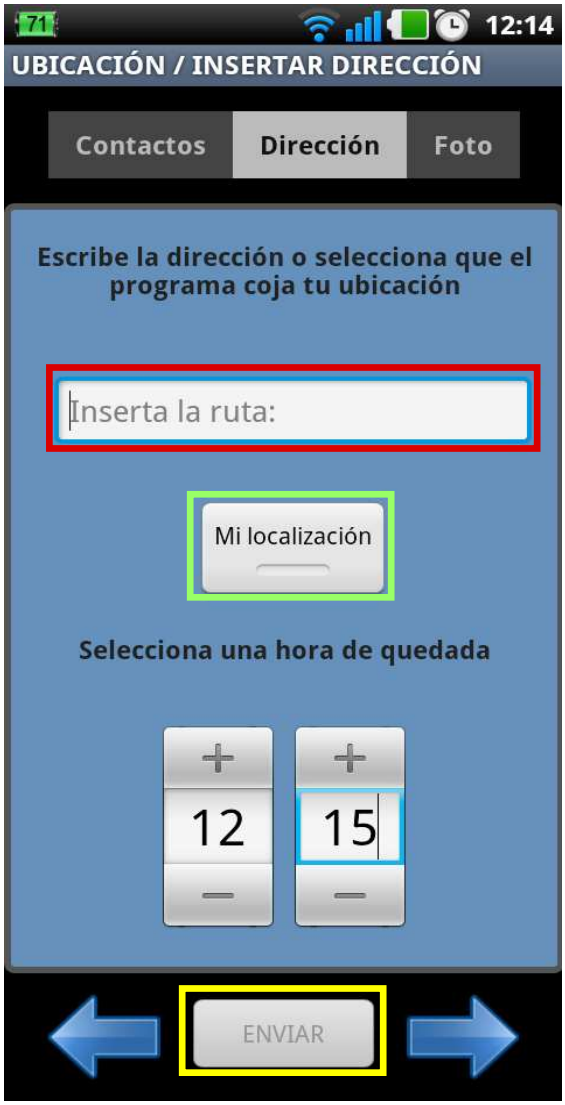
11.5.1.2 Añadir grupo



Ilustracion 37. Adición de un grupo

Al pulsar el botón *Añadir grupo*, se muestra la lista de grupos que el usuario ha definido en la aplicación. De dicha lista, el usuario debe seleccionar uno de ellos y pinchar en él. De esta forma, el grupo de contactos al que enviarles la información queda definido y la aplicación pasaría a la siguiente pantalla, correspondiente con la *Dirección*, al igual que en el caso anterior, y sin necesidad de que el usuario haga nada más.

11.5.2 Dirección



Ilustracion 38. Selección de dirección

En esta pantalla el usuario debe introducir la información relativa al tiempo y espacio donde quiere quedar con el/los contacto/s seleccionados en el paso anterior. Para ello debe rellenar la dirección donde quiere realizar el encuentro y para eso dispone de dos opciones:

- Introducir la dirección manualmente, a través del teclado. Se correspondería con la información destacada en color rojo que se muestra en la imagen anterior.

- Obtener la ubicación donde se encuentra el usuario. Esto se haría pulsando el botón remarcado en verde de *Mi localización*. Para que la aplicación halle automáticamente la ubicación en la que se encuentra el dispositivo es necesario que esté activado en el mismo el GPS y/o el 3G.

Además de esto, el usuario debe rellenar la hora de encuentro.

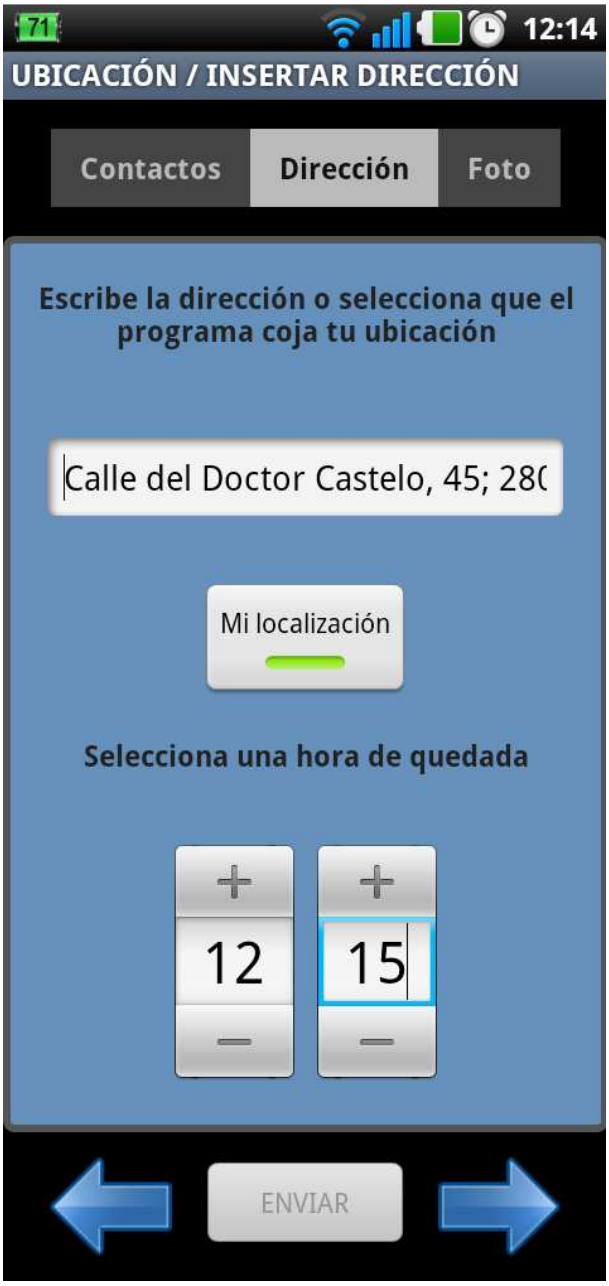
Nótese que el botón de *Enviar*, remarcado en amarillo, sigue inactivo, pues con ello el usuario está obligado a pasar a la siguiente pantalla, cerciorándose así de que es consciente de todas las posibilidades que la aplicación le brinda.

De igual forma se debe señalar que en la parte inferior existen dos flechas azules por medio de las cuales se podrá:

- En el caso de la flecha izquierda, navegar hacia la pantalla anterior para modificar el/los contacto/s seleccionados previamente.

- En el caso de la flecha derecha, navegar hacia la pantalla posterior, donde el usuario podrá adjuntar una fotografía si así lo desea.

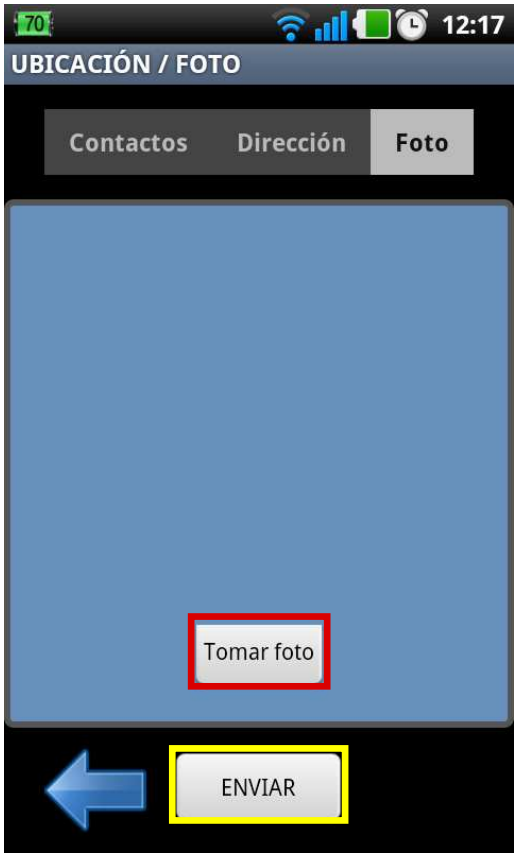
Un ejemplo de rellenado de la información necesaria en esta pantalla podría ser el siguiente:



Ilustracion 39. Selección de dirección (ejemplo)

Tras rellenar los datos pertinentes, el usuario debería pulsar la flecha que apunta a la derecha para llegar a la última pantalla, correspondiente con la *Fotografía*.

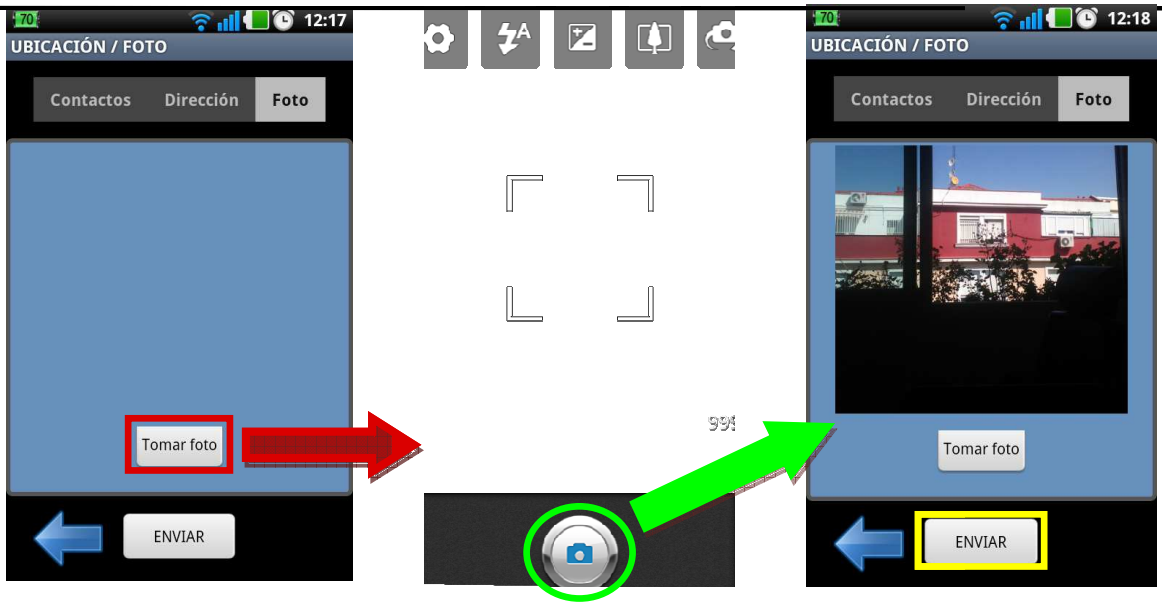
11.5.3 Fotografía



Ilustracion 40. Tomar fotografía

El tercer y último paso que el usuario debe realizar para crear un *encuentro* es decidir si quiere o no adjuntar una fotografía con el mismo. Por ello:

- Si el usuario no desea adjuntar fotografía, deberá pulsar directamente el botón *Enviar*.
- Por el contrario, si el usuario desea adjuntar una fotografía, deberá pulsar el botón *Tomar foto*. Tras esto, la aplicación accederá a la cámara que posea el dispositivo. Por medio de ella, el usuario tomará la fotografía que desee, pasando a formar parte del *encuentro*. A continuación se muestran imágenes que ilustrarán el proceso:



Ilustracion 41. Proceso de toma de fotografía

Tras esto el encuentro queda totalmente definido, enviándose al/los contactos que el usuario haya seleccionado y guardándose además dicho *encuentro* en los encuentros personales del usuario (pantalla *Mis encuentros*), donde podrá consultarlos.

11.6 Mis encuentros

Ilustracion 42. Mis encuentros

Para consultar los *encuentros* que posee el usuario, se debe pulsar el botón *Mis encuentros.* Tras esto, la aplicación le muestra al usuario la lista de encuentros que tiene programados, de donde podrá consultar la información relativa a los mismos tan sólo pulsando en ellos. A continuación se muestra un ejemplo:

Ilustracion 43. Información de encuentros

Además de la información sobre el emisor, receptor, hora y lugar del encuentro, la aplicación también muestra la ubicación de la dirección sobre el plano. Para ello, el usuario únicamente debe pulsar sobre el botón *Mostrar Mapa*. En la siguiente ilustración se muestra un ejemplo:

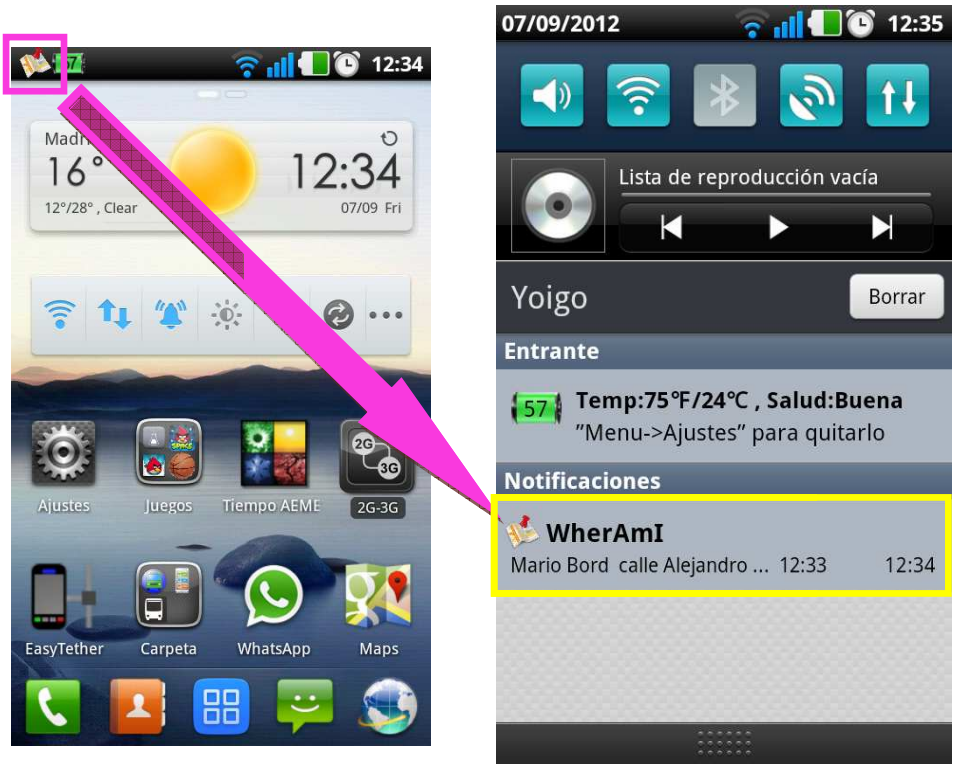
Ilustracion 44. Dirección en el mapa

11.7 Notificaciones

Además de lo explicado en los apartados anteriores, la aplicación gestiona notificaciones, es decir, cuando un usuario crea un *encuentro* y se lo manda a uno o más contactos, dichos contactos reciben la notificación, formada por su respectivo icono y mensaje, del encuentro en su dispositivo. El icono que aparece es el mismo que el que tiene la aplicación para su inicio, tal y como muestran las siguientes imágenes.

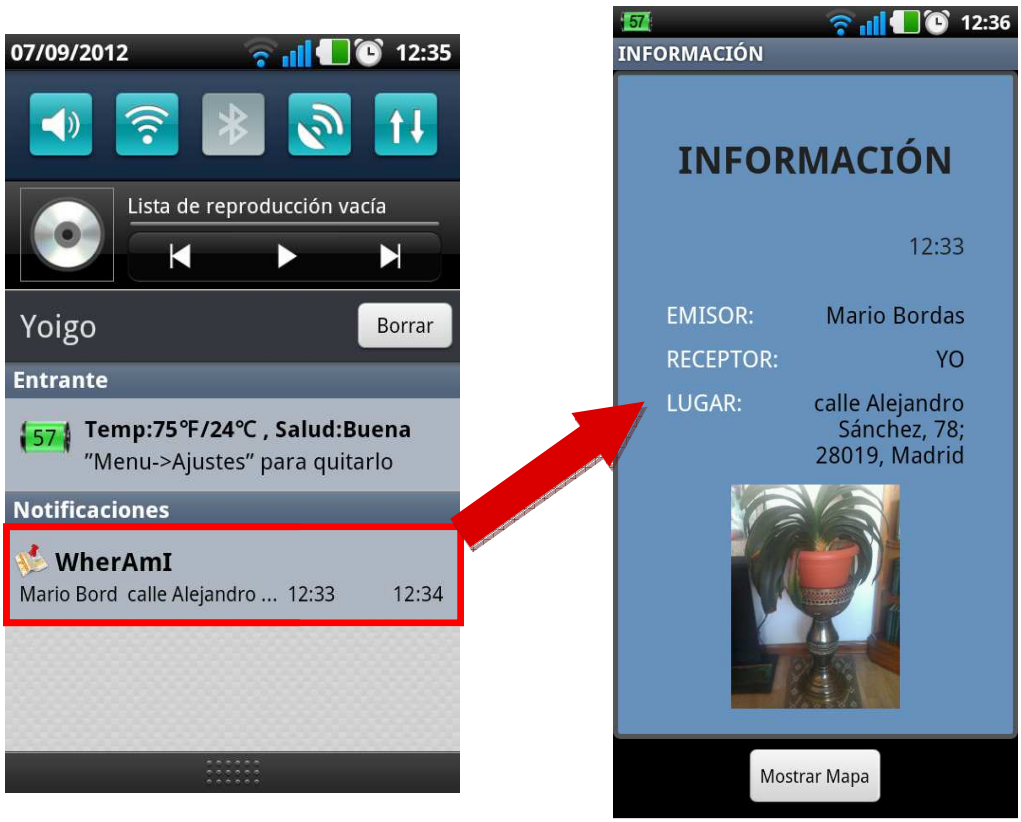
Ilustracion 45. Notificaciones (I)

Al acceder a dichas notificaciones, el dispositivo muestra información sobre el *encuentro* de la siguiente forma:



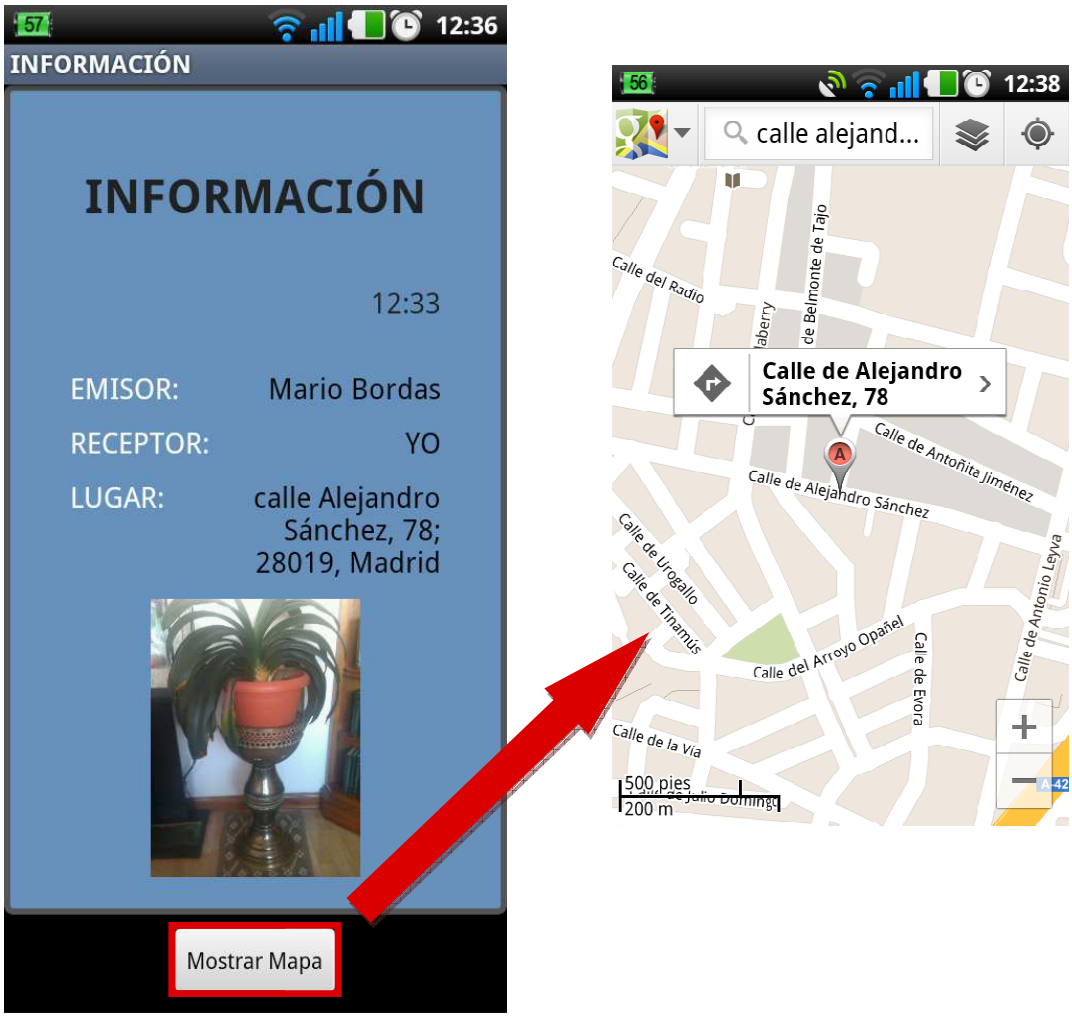
Ilustracion 46. Notificaciones (II)

Para acceder a la información completa sobre dicho *encuentro* únicamente hay que pulsar sobre la notificación de la aplicación *WherAmI* señalada en amarillo, y tras esto, el usuario podrá acceder a la información completa del encuentro.



Ilustracion 47. Notificaciones (III)

No olvidar que desde el punto de información de un *encuentro* se puede acceder a la ubicación exacta sobre el plano de la dirección de dicho *encuentro*.



Ilustracion 48. Notificaciones (IV)