UNIVERSIDAD POLITÉCNICA DE CARTAGENA

Departamento
TEORÍA DE LA SEÑAL Y LAS COMUNICACIONES

PROYECTO FIN DE CARERA
INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN

"SISTEMA DE LOCALIZACIÓN DE TAXI BASADO EN ANDROID, PHP Y MYSQL"

Dirigido por
Dr. ALEJANDRO ÁLVAREZ MELCÓN
D. PEDRO VERA CASTEJÓN

Realizado por
FRANCISCO RAMOS CASTRO

Cartagena, Septiembre de 2012

I. Índice General

1. Definición de Objetivos

Este proyecto informático trata de realizar un sistema de localización y visionado en tiempo real de taxis cercanos al punto en el que se encuentra el usuario de la aplicación. A su misma vez, permite al taxista la recepción de peticiones así como la visualización de la localización del cliente

Para ello, se han realizado dos aplicaciones Android, por un lado la "Aplicación cliente", que muestra los taxis libres más cercanos que hay desde el punto geográfico en el que se encuentra el usuario. Por otro lado, la "Aplicación Taxista", que envía los datos de localización periódicamente a un servidor MYSQL y muestra la localización del cliente que solicita ese taxi.

El sistema operativo para dispositivos móviles creado por Google (Android) se está extendiendo cada vez más; además los dispositivos móviles actuales incorporan todo tipo de sensores y chips con los que podemos realizar aplicaciones interesantes, por ejemplo, en este proyecto se aprovecha la funcionalidad del chip receptor de GPS.

La aplicación Usuario obtiene la localización actual del usuario y su identificador de terminal IMEI, después estos datos se envían a un servidor de base de datos MYSQL en el que se encuentran almacenadas las localizaciones e identificadores de taxis libres. Una vez realizada la consulta se hace uso de la tecnología Google Maps para representar la localización actual del usuario y la de los taxis cercanos en un mismo mapa.

La aplicación Taxista obtiene la localización actual del usuario y su identificador de terminal IMEI, después estos datos se envían a un servidor de base de datos MYSQL en el que se encuentran almacenadas las localizaciones e identificadores de clientes cercanos. Una vez realizada la consulta se hace uso de la tecnología Google Maps para representar la localización actual del taxi y la del cliente que solicita taxi en un mismo mapa.

Además de las dos aplicaciones, para que el sistema pueda funcionar se requiere un servidor de base de datos MYSQL en el que se almacenarán las localizaciones e identificadores de todos los clientes y taxistas y un servicio web encargado de intercambiar datos entre la aplicación Android y la base de datos.

Este servicio web va a ser por así decirlo el "controlador del sistema" ya que debe registrar nuevos usuarios, borrar usuarios desconectados y devolver a cada usuario los datos de acuerdo a su localización.

Para conseguir todo esto, se han mezclado distintos periféricos del teléfono móvil como son el receptor GPS y conexión de de datos tanto WIFI como GSM; además se ha hecho uso de herramientas como el SDK de Android para Eclipse, Google Maps y se ha contratado un servidor de base de datos MYSQL con la Empresa 1and1.

En primer lugar, se explican las características básicas del sistema operativo Android; tras esto, se desarrolla una completa guía de instalación del SDK Android para Eclipse para empezar a programar las primeras aplicaciones, también se darán unas nociones básicas de manejo de la API de Google Maps.

A continuación se verán las características del dispositivo móvil que se ha utilizado para el proyecto así como el funcionamiento de los periféricos usados, en este caso, el

GPS. Para finalizar con las herramientas utilizadas para el proyecto se detalla las características y funcionamiento del motor de base de datos MYSQL.

Una vez que se han explicado las herramientas y tecnologías implicadas en esto proyecto se da paso a la explicación del desarrollo del sistema, el cual está formado por una aplicación para clientes, otra aplicación para taxistas, una base de datos MYSQL y un servicio web programado en lenguaje PHP.

2. Introducción a Android

2.1 ¿Qué es una plataforma de software?

Una plataforma de software es un elemento crucial en el desarrollo del mismo. Mediante el marco de trabajo que proporcionan las plataformas se puede crear nuevo software y que este se pueda ejecutar sobre ellas (*ver Fig 2.1*). Las plataformas de desarrollo típicas incluyen una arquitectura de computadores, un sistema operativo (S.O.), lenguajes de programación y sus correspondientes librerías o interfaces gráficas (user interface o UI).

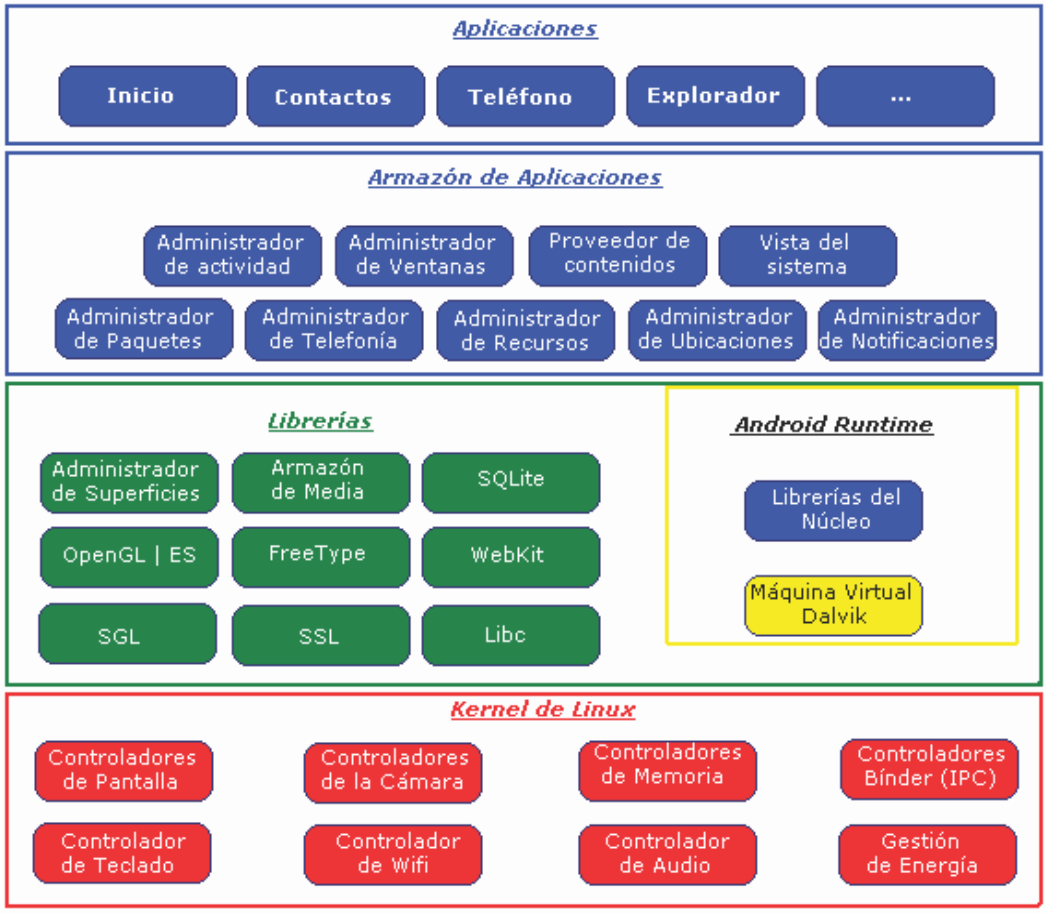


Figura 2.1 Sistema de capas de Android

Las plataformas son comúnmente mencionadas con las APIs (funciones y métodos que ofrece una biblioteca para ser utilizada por otro software como una capa de abstracción). Un conjunto completo de APIs constituye una plataforma software. Estas plataformas son normalmente dependientes de los sistemas operativos aunque existen otras en las que no es así.

Java es un ejemplo de plataforma no dependiente del S.O. debido a que Java es tanto el lenguaje como la plataforma de desarrollo, la cual incluye la Java Virtual Machine (JVM) cuya función es interpretar el bytecode resultante de la compilación del código fuente del programa en Java.

2.2 ¿Qué es Android?

Android es una plataforma de software y un sistema operativo para dispositivos móviles basada en un kernel Linux, desarrollada por Google y más tarde por la Open Handset Alliance. Esta plataforma permite a los desarrolladores escribir código en Java que se ejecute en móviles mediante las librerías Java desarrolladas por Google. También se pueden escribir aplicaciones en otros lenguajes, como por ejemplo C, para posteriormente ser compiladas en código nativo ARM y ejecutarlas, aunque este proceso de desarrollo no está soportado oficialmente por Google. La mayor parte de la plataforma de Android está disponible bajo licencia de software libre de Apache y otras licencias de código abierto.

2.2.1 Breve Historia

En Julio de 2005, Google adquirió Android, Inc , una pequeña Startup de California. En esos momentos, la compañía se dedicaba a la creación de software para teléfonos móviles. Una vez en Google, el equipo desarrolló un S.O. basado en Linux para dispositivos móviles. Más adelante, Google adaptó su buscador y sus aplicaciones para su uso en móviles.

En septiembre del 2007, Google tenía varias patentes de aplicaciones sobre el área de la telefonía móvil. El 5 de noviembre del mismo año, se anunció la fundación de la Open Handset Alliance al mismo tiempo que la creación de la plataforma Android. La Open Handset Alliance está formada por un consorcio de 34 compañías de hardware, software y telecomunicaciones, entre las cuales se incluyen Google, HTC , Intel y Motorola entre otras, dedicadas a investigar estándares abiertos para dispositivos móviles.

El primer teléfono en el mercado que posee Android es el T-Mobile G1 (también conocido como Dream), lanzado el día 22 de octubre de 2008 que viene con la versión Android 1.0 preinstalada. Este móvil es el resultado conjunto de T-Mobile, HTC y Google. Por último, desde el 21 de octubre de 2008, Android está disponible como código abierto. Gracias a esto, cualquiera puede añadir extensiones, nuevas aplicaciones o reemplazar las existentes por otras dentro del dispositivo móvil.

2.2.2 Características de Android

- Amplia variedad de diseños (VGA, librerías de gráficos 2D y 3D…)
- Almacenamiento de datos en BBDD SQLite [9]
- Conectividad (GSM/EDGE, CDMA, EV-DO, UMTS, Bluetooth y Wi-Fi)
- Mensajería (SMS y MMS)
- Navegador Web
- Máquina virtual de Java
- Las aplicaciones escritas en Java pueden ser compiladas y ejecutadas en la maquina virtual de Dalvik, la cual es una especializada máquina virtual diseñada para uso en dispositivos móviles.
- Soporte de formatos (MPEG-4, H.264, MP3, AAC, OGG, AMR, JPEG, PNG, GIF)
- Soporte para hardware adicional (cámaras de video, pantallas táctiles, GPS, acelerómetros….)
- Entorno de desarrollo (emulador, herramientas de depuración, perfiles de memoria y funcionamiento, plugin para Eclipse IDE).

2.2.3 Arquitectura de Android

En la Fig. 2.2 se muestra la composición de la arquitectura de Android.

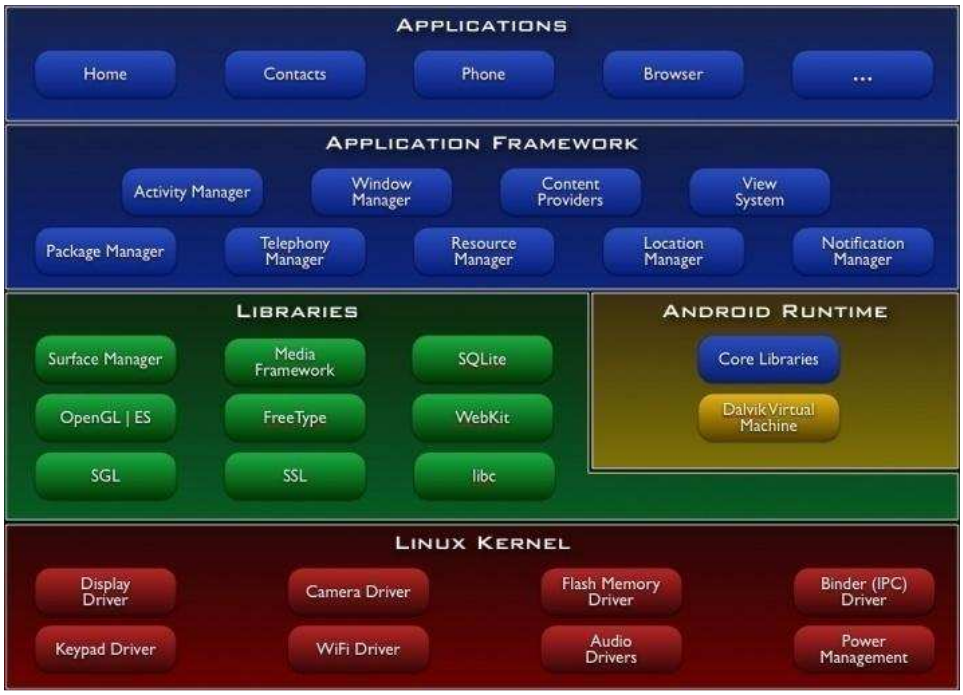


Figura 2.2 Arquitectura de Android

A continuación, se explican los diferentes componentes de la arquitectura de Android:

- Aplicaciones: ejemplos de aplicaciones base son un cliente de email, programa de SMS, calendario, mapas, navegador, contactos, y otros. Todas las aplicaciones están escritas en el lenguaje de programación Java.

- Framework de aplicaciones: los desarrolladores tienen acceso completo a las APIs del framework usado por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades (sujeto a reglas de seguridad del framework). Éste mismo mecanismo permite que los componentes sean reemplazados por el usuario. Este framework está formado por un extenso conjunto de vistas tales como listas, cajas de texto, botones…

- Content Providers, permiten a las aplicaciones acceder a información de otras aplicaciones o compartir su propia información.

- Resource Manager, proporciona acceso a recursos que no son código como pueden ser gráficos, cadenas de texto...

- Notification Manager, permite a las aplicaciones mostrar alarmas personalizadas en la barra de estado.

- Activity Manager, gestiona el ciclo de vida de las aplicaciones.

- Librerías: Android incluye un set de librerías C/C++ usadas por varios componentes del sistema. Estas capacidades se exponen a los desarrolladores a través del framework de aplicaciones de Android, el cual interactúa con las librerías mediante JNI (Java Native Interface). Algunas son:

System C Library (implementación librería C estándar), librerías de medios, librerías de gráficos, 3D, SQLite, entre otras.

- Runtime de Android: Android incluye un set de librerías base que proveen la mayor parte de las funcionalidades disponibles en las librerías base del lenguaje de programación Java. Cada aplicación Android corre su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente. Dalvik ejecuta archivos en el formato Dalvik Executable (.dex), el cual está optimizado para memoria mínima.

- Núcleo Linux: Android depende de un Linux versión 2.6 para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red, y modelo de drivers. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila.

2.3 Android SDK

El kit de desarrollo de software (Software Development Kit o SDK) incluye un conjunto de herramientas de desarrollo, tales como un debugger, librerías, un emulador (basado en QEMU), documentación, código de ejemplo y tutoriales. Está soportado en S.O. Windows, Linux y Mac. El entorno de desarrollo (Integrated Development Environment o IDE) oficialmente soportado es Eclipse conjuntamente con el plugin ADT proporcionado por Google.

2.4 Anatomía de una aplicación de Android

Dentro de una aplicación de Android hay cuatro componentes principales: Activitity, Listeners, Servicios y Content Provider. Todas las aplicaciones de Android están formadas por algunos de estos elementos o combinaciones de ellos.

2.4.1 Activity

Las Activities (o Actividades) son el elemento constituyente de Android más común. Para implementarlas se utiliza una clase por cada Actividad que extiende de la clase base Activity. Cada clase mostrará una interfaz de usuario, compuesta por Views (o Vistas). Cada vez que se cambie de Vista, se cambiará de Actividad, como por ejemplo en una aplicación de mensajería que se tiene una Vista que muestra la lista de contactos y otra Vista para escribir los mensajes. Cuando cambiamos de Vista, la anterior queda pausada y puesta dentro de una pila de historial para poder retornar en caso necesario. También se pueden eliminar las Vistas del historial en caso de que no se necesiten más. Para pasar de vista en vista, Android utiliza una clase especial llamada Intent.

2.4.1.1 Intents

Un Intent es un objeto mensaje y que, en general, describe que quiere hacer una aplicación. Las dos partes más importantes de un Intent son la acción que se quiere realizar y la información necesaria que se proporciona para poder realizarla, la cual se expresa en formato URI [12]. Un ejemplo sería ver la información de contacto de

una persona, la cual mediante un Intent con la acción ver y la URI que representa a esa persona se podría obtener.

Relacionado con los Intents hay una clase llamada IntentFilter que es una descripción de qué Intents puede una Actividad gestionar. Mediante los IntentFilters, el sistema puede resolver Intents, buscando cuales posee cada actividad y escogiendo aquel que mejor se ajuste a sus necesidades. El proceso de resolver Intents se realiza en tiempo real, lo cual ofrece dos beneficios:

- Las actividades pueden reutilizar funcionalidades de otros componentes simplemente haciendo peticiones mediante un Intent.
- Las actividades pueden ser remplazadas por nuevas actividades con IntentFilters equivalentes.

2.4.2 Listeners

Los Listeneres se utilizan para reaccionar a eventos externos (por ejemplo, una llamada). Los Listeners no tienen UI pero pueden utilizar el servicio Notification Manager para avisar al usuario.

Para lanzar un aviso no hace falta que la aplicación se esté ejecutando, en caso necesario, Android la iniciará si se activa el Listeners por algún evento.

2.4.3 Servicios

Un Servicio es básicamente un código que se ejecuta durante largo tiempo y sin necesidad de UI, como puede ser un gestor de descarga en el cual se indican los contenidos a descargar y posteriormente el usuario puede acceder a una nueva Vista sin que el gestor se interrumpa.

En caso de que haya múltiples servicios a la vez, se les puede indicar diferentes prioridades según las necesidades.

2.4.4 Content Provider

En Android, las aplicaciones pueden guardar su información en ficheros, BBDD SQLite… Pero en caso de que lo que se quiera sea compartir dicha información con otras aplicaciones, lo necesario es un Content Provider. Un Content Provider es una clase que implementa un conjunto estándar de métodos que permite a otras aplicaciones guardar y obtener la información que maneja dicho Content Provider.

2.4.5 Android Manifiest

En Android existe un archivo XML llamado AndroidManifest que, aunque no forme parte del código principal de la aplicación, es necesario para su correcto funcionamiento. Este archivo es el fichero de control que le dice al sistema qué tiene que hacer con todos los componentes anteriormente mencionados en este apartado que pertenecen a una aplicación en concreto.

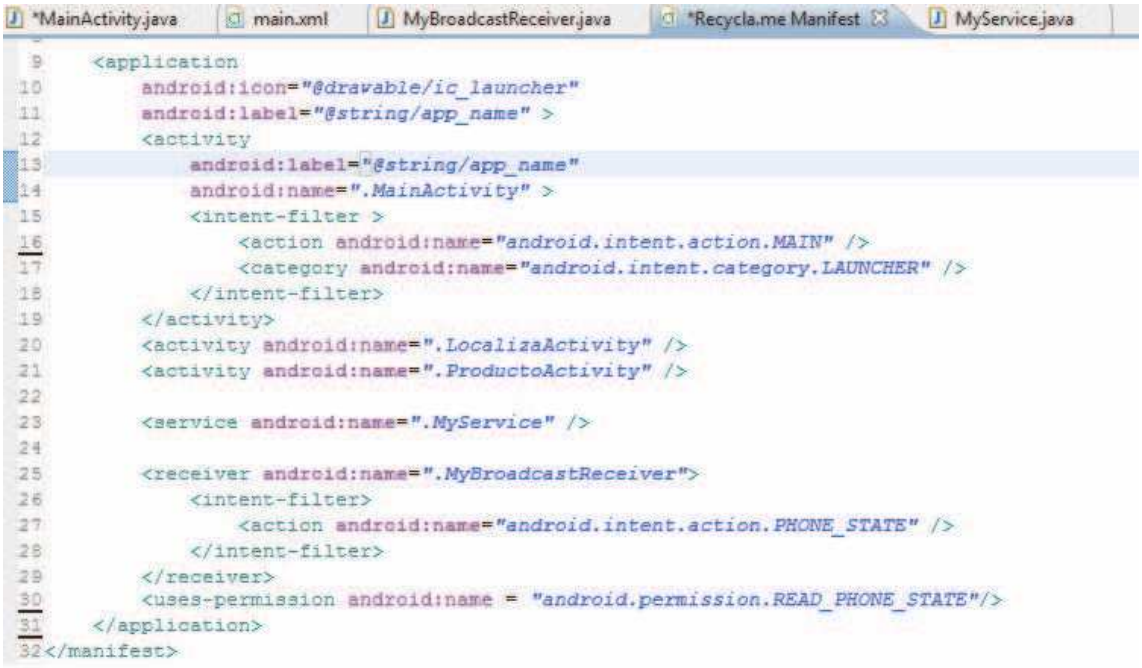


Imagen 2.1 Ejemplo de AndroidManifest

2.5 Tipos de aplicación de Android

Un Android Package (.apk) es el fichero que contiene el código de la aplicación y sus recursos, y que posteriormente se instala en el dispositivo para poder ejecutar la aplicación.

2.5.1 Tareas

Una tarea en Android es lo que el usuario ve como una aplicación y el desarrollador ve como una o más actividades donde el usuario interacciona y va pasando de Vista en Vista.

Dentro de las tareas, una actividad toma el papel de punto de entrada (será la primera en mostrarse cuando se ejecute la aplicación) y las demás, si hay, formarán parte de la misma tarea, a la espera de ser instanciadas.

2.5.2 Procesos

En Android, los procesos se ejecutan a nivel de kernel y el usuario normalmente no tiene constancia de ellos. Todo el código de la aplicación se suele ejecutar en un proceso dedicado pero también se puede especificar si sólo se quiere que se ejecute en el proceso una determinada clase o componente de la aplicación.

Los principales usos de los procesos son:

- Mejorar la estabilidad o seguridad de las aplicaciones.
- Reducir la sobrecarga de proceso ejecutando el código de múltiples aplicaciones en el mismo proceso.
- Ayudar al sistema a gestionar los recursos separando partes de código pesado en un proceso separado que puede ser eliminado independientemente de otras partes de la aplicación.

2.5.3 Threads

En lo referente a threads, Android evita la creación de threads adicionales por parte de un proceso, manteniendo la aplicación en un sólo thread a menos que no los cree la propia aplicación. Esto repercute de manera importante en las llamadas a instancias de Actividades, Listeners y Servicios, ya que sólo pueden ser hechas por el thread principal del proceso en el que están corriendo.

Por otro lado, al no crearse un thread por cada instancia, dichas instancias no deben realizar operaciones largas o bloqueantes cuando son llamadas, de lo contrario, bloquearían todos los demás componentes del proceso.

2.6 Ciclo de vida de una aplicación de Android

Cada aplicación de Android corre en su propio proceso, el cual es creado por la aplicación cuando se ejecuta y permanece hasta que la aplicación deja de trabajar o el sistema necesita memoria para otras aplicaciones.

Una característica fundamental de Android es que el ciclo de vida de una aplicación no está controlado por la misma aplicación sino que lo determina el sistema a partir de una combinación de estados como pueden ser qué aplicaciones están funcionando, qué prioridad tienen para el usuario y cuánta memoria queda disponible en el sistema. De esta manera, Android sitúa cada
proceso en una jerarquía de importancia basada en los estados comentados,
como se puede ver a continuación.

1. Un proceso en primer plano es uno que se requiere para lo que el usuario está actualmente haciendo. Se considera en primer plano si:

- Está ejecutándose una Actividad perteneciente a la pantalla con la que el usuario está interactuando.
- Está ejecutando un BroadcastReceiver
- Esta ejecutándose un servicio

2. Un proceso visible es aquel que contiene una Actividad que es visible al usuario mediante la pantalla pero no en primer plano (está pausada). Este proceso solo se eliminará en caso de que sea necesario para mantener ejecutándose los procesos en primer plano.

3. Un proceso de servicio es aquel que contiene un servicio que ha sido inicializado. No son directamente visibles al usuario y el sistema los mantendrá a no ser que no pueda servir los dos anteriores.

4. Un proceso en background es aquel que acoge una actividad que no está actualmente visible al usuario. Mientras que dichos procesos implementen bien su propio ciclo de vida, el sistema puede eliminarlos para dar memoria a cualquiera de los 3 servicios anteriores.

5. Un proceso vacío es aquel que no contiene ningún componente activo de ninguna aplicación. La única razón para mantener dicho proceso es para mejorar sus inicializaciones posteriores a modo de caché.

Para comprender mejor el ciclo de vida de una aplicación de Android, en la Fig.4 se muestra el diagrama de flujo de dicho ciclo, mencionando también los métodos que se llaman durante el transcurso del mismo.

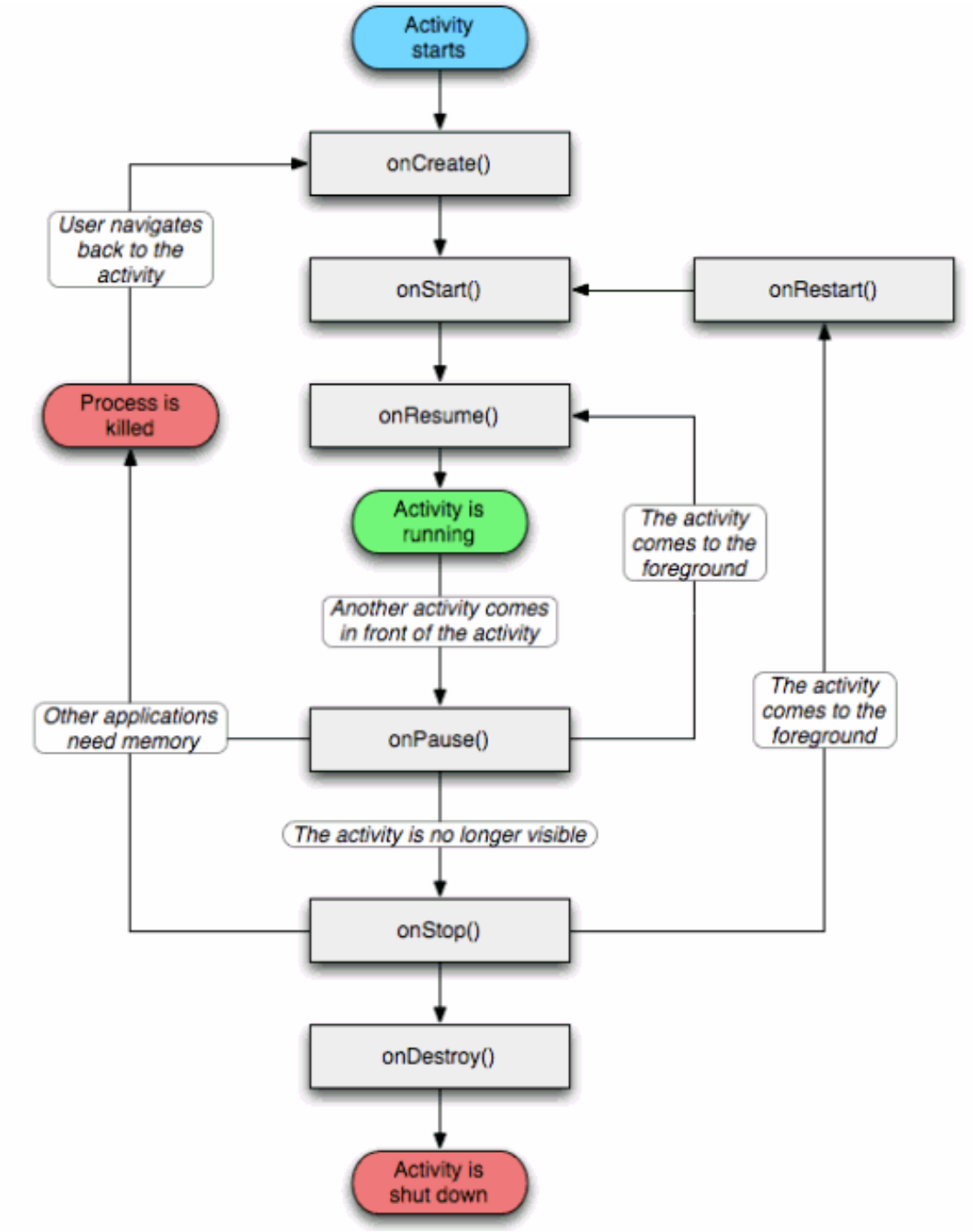


Figura 2.3. Ciclo de vida de una aplicación en Android

2.7 Almacenamiento

Android presenta varias formas de almacenamiento de la información, es importante notar que en Android todos los datos de la aplicación son privados a ésta.

2.7.1 Imágenes de disco

Dentro de la categoría de ficheros, encontramos las imágenes de disco, que Android usa para poder almacenar los datos y las diferentes configuraciones, las cuales simulan particiones flash de disco, por ejemplo, una imagen que contenga el kernel del emulador, una imagen para la memoria ram, datos de usuario y una imagen para simular una tarjeta SD.

Por defecto, Android guarda localizaciones predeterminadas para estas imágenes. Durante el arranque, el emulador busca y lee estos ficheros de imágenes. Esta ubicación se puede especificar mediante las opciones de arranque del emulador; si no encuentra ninguna imagen, utiliza las direcciones por defecto.

El emulador usa tres tipos de imagen: de sistema, de runtime, y temporales.

- Las imágenes de sistema contienen datos del sistema y configuraciones por defecto necesarias para que el emulador arranque. Estas imágenes son de sólo lectura, ya que no se deben modificar.

- Las imágenes de runtime leen y escriben datos. Son dos: la de datos y la de tarjetas SD emuladas. Cada emulador usa una imagen de datos de usuario para guardar los datos específicos de sesión y usuario, como por ejemplo, las aplicaciones instaladas. Si se usan varias instancias de emulador a la vez, solo la primera puede usar datos persistentes si no se especifican ficheros de imagen diferentes en el arranque. Opcionalmente, se pueden crear imágenes de disco que emulen tarjetas SD. Solamente se puede cargar una imagen de disco en la carga del emulador. Además una vez cargada una tarjeta SD, no se puede extraer de un emulador en funcionamiento, pero si navegar por ella y añadir o remover archivos. El emulador soporta tarjetas SDHC emuladas por lo que soporta tamaños de hasta 128 gigabytes.

Por último están las imágenes temporales, las cuales se eliminan al apagarse el dispositivo. Estas imágenes son una copia de la imagen de sistema de Android y la imagen de la memoria caché.

2.8 Gráficos

Para implementar gráficos 2D y 3D (ver Fig. 5), Android utiliza OpenGL ES (OpenGL for Embedded Systems) [47], una variante simplificada de la API gráfica OpenGL [48] diseñada para dispositivos integrados tales como teléfonos móviles, PDAs y consolas de videojuegos. La define y promueve el Grupo Khronos [49], un consorcio de empresas dedicadas a hardware y software gráfico interesadas en APIs gráficas y multimedia. Existen varias versiones, de entre las cuales la versión 1.0 ha sido seleccionada como la API oficial para gráficos 3D de Android.

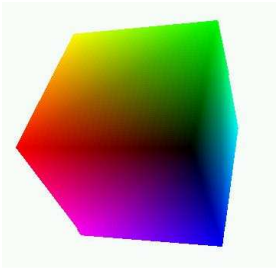


Imagen 2.2. Ejemplo de gráfico 3D generado con OpenGL ES

La API original, OpenGL, es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. La interfaz consiste en más de 250 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples, tales como puntos, líneas y triángulos. Fue desarrollada originalmente por Silicon Graphics Inc. (SGI) en 1992 y se usa ampliamente en CAD, realidad virtual, representación científica, visualización de información y simulación de vuelo. También se usa en desarrollo de videojuegos, donde compite con Direct3D [13] en plataformas Microsoft Windows.

OpenGL tiene dos propósitos esenciales:

- Ocultar la complejidad de la interfaz con las diferentes tarjetas gráficas, presentando al programador una API única y uniforme.

- Ocultar las diferentes capacidades de las diversas plataformas hardware, requiriendo que todas las implementaciones soporten la funcionalidad completa de OpenGL (utilizando emulación software si fuese necesario).

2.9 Direccionamiento en Android

Cada instancia del emulador corre detrás de un router virtual que aísla el emulador de las interfaces de red del PC. Un dispositivo emulado no puede ver el PC u otro emulador de la red. En su lugar, solo ve que está conectado a través de una E thernet a un router.

Cabe destacar que las mismas direcciones son usadas en todos los emuladores que se están ejecutando, reforzando el aislamiento. Para consultar el espacio de direcciones de Android. También destacar que este direccionamiento es específico al emulador y que probablemente sea muy diferente en dispositivos reales. Actualmente, el emulador no soporta IGMP [14] o multicast [15].

2.10 Recursos

Los recursos en Android son ficheros externos que no son código y que son usados y compilados dentro de la aplicación. Android soporta diferentes tipos de recursos, incluido XML, PNG y JPEG. Los ficheros XML tienen diferente formato dependiendo de qué describan.

Los recursos se exteriorizan respecto al código. Los ficheros XML se compilan dentro un archivo binario por razones de eficiencia y los strings se comprimen en una forma más eficiente de almacenamiento. Es por esta razón que hay diferentes tipos de recursos en Android. En general, hay recursos de tres tipos: ficheros XML, Bitmaps y ficheros Raw (por ejemplo, ficheros de sonido).

2.11 Versiones disponibles

El sistema operativo Android, al igual que los propios teléfonos móviles, ha evolucionado rápidamente, acumulando una gran cantidad de versiones, desde la 1.0 para el QWERTY HTC G1, hasta la 4.1 que acaba de salir al mercado.

- **Cupcake: Android Version 1.5**

Características: Widgets, teclado QWERTY virtual, copy & paste, captura de vídeos y poder subirlos a Youtube directamente.

- **Donut: Android Version 1.6**

Características: Añade a la anterior la mejoría de la interfaz de la cámara, búsqueda por voz, y navegación en Google Maps.

- **Eclair: Android Version 2.0/2.1**

Características: Mejoras en Google Maps, salvapantallas animado, incluye zoom digital para la cámara, y un nuevo navegador de internet.

- **Froyo: Android Version 2.2**

Características: Incluye hotspot Wifi, mejora de la memoria, más veloz, Microsoft Exchange y video-llamada.

- **GingerBread: Android Version 2.3**

Características: Mejoras del consumo de batería, el soporte de vídeo online y el teclado virtual, e incluye soporte para pagos mediante NFC.

- **Honey Comb: Android Version 3.0/3.4**

Características: Mejoras para tablets, soporte Flash y Divx, integra Dolphin, multitarea pudiendo cambiar de aplicación dejando las demás en espera en una columna, widgets y homepage personalizable.

- **Ice Cream Sandwich: Android Version 4.0**

Características: Multiplataforma (tablets, teléfonos móviles y netbooks), barras de estado, pantalla principal con soporte para 3D, widgets redimensionables, soporte usb para teclados, reconocimiento facial y controles para PS3.

- **Jelly Bean: Android Version 4.1**

Características: Multiplataforma (tablets, teléfonos móviles y netbooks), barras de estado, pantalla principal con soporte para 3D, widgets redimensionables, soporte usb para teclados, reconocimiento facial y controles para PS3.

3. Eclipse como entorno de trabajo. Manual de instalación

3.1 Requisitos de software

Para desarrollar aplicaciones en Android usando las herramientas que aporta su SDK, es necesario disponer del siguiente software en nuestro PC:

Sistema operativo:
- Windows XP o Vista
- Mac OS X 10.4.8 o superior (solo x86)
- Linux (probado en Linux Ubuntu)

Entornos de desarrollo de aplicaciones:
- Eclipse 3.3 (Europe), Eclipse 3.4 (Ganymede), Eclipse 3.5 (Galileo), Eclipse Juno.
- JDK 5 o superior
- Android Development Tools plugin (Opcional)

Nota: En esta guía se explica detalladamente la instalación del SDK de Android para el sistema operativo Windows XP, usando el entorno de desarrollo de aplicaciones Eclipse 3.4 (Ganymede).

3.2 Instalación de Eclipse IDE

Si nuestro PC no dispone de un entorno de desarrollo compatible con Android SDK, recomendamos descargar Eclipse 3.4 (Ganymede), en la web oficial de Eclipse: http://www.eclipse.org/downloads.

A continuación se van a explicar los pasos básicos para tenerlo funcionando en nuestro equipo.

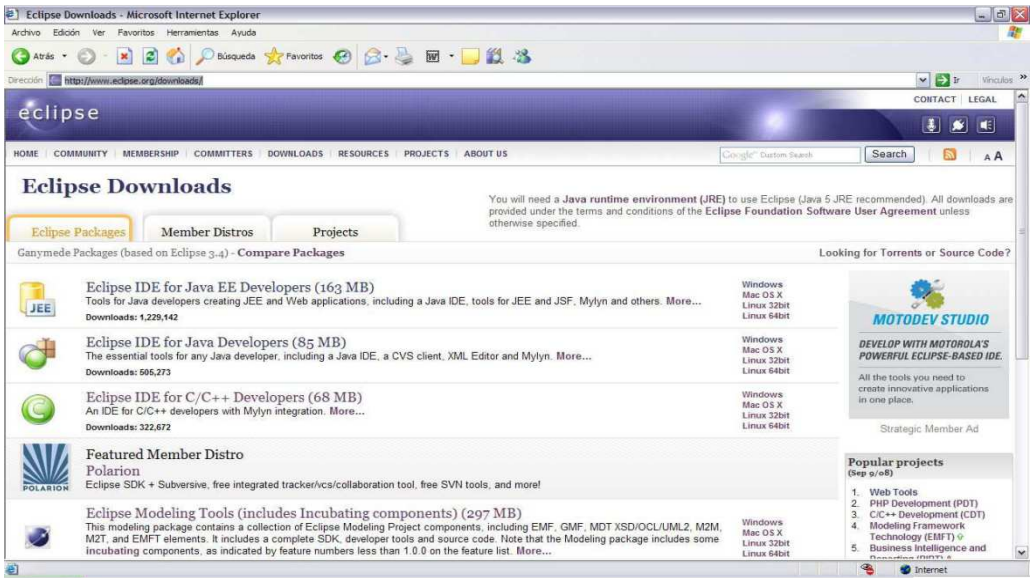


Imagen 3.1 Página Web de Eclipse

Seleccionamos **Eclipse Classic 3.4** para Windows y nos los descargamos en nuestro equipo. Descomprimimos el archivo .zip en el directorio que hemos seleccionado y ejecutamos el archivo **eclipse.exe** que se encuentra en la carpeta eclipse descomprimida. Finalmente, elegimos un espacio de trabajo donde elaborar nuestros proyectos.

3.3 Instalación del SDK de Android

Para descargar el SDK de Android de su página oficial, iremos al enlace mencionado en el apartado anterior, "http://developer.android.com/sdk/1.5_r1/index.html", y más tarde pulsamos sobre el enlace para iniciar la descarga, en nuestro caso para el sistema operativo Windows. A continuación, aceptamos los términos de licencia del SDK e iniciamos la descarga.

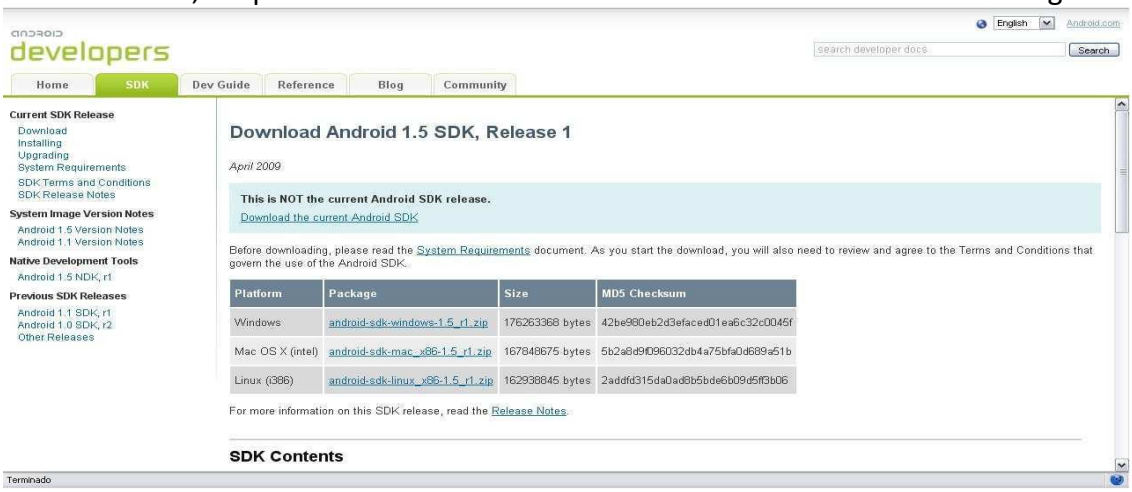


Imagen 3.2 Descarga del SDK de Android

Después de descargar el SDK, descomprimimos el archivo en el equipo. Es importante recordar el directorio donde se ha guardado y el nombre de la carpeta, para usar las herramientas del SDK. Opcionalmente, podemos añadir la ruta del directorio 'tools' del SDK al 'PATH' de nuestro equipo. Para ello, seguimos los siguientes pasos:

- *En el icono de Mi PC, pulsamos encima el botón derecho del ratón y seleccionamos **Propiedades**.*

- *En la pestaña de **Opciones avanzadas** pulsamos el botón de **Variables de entorno**.*

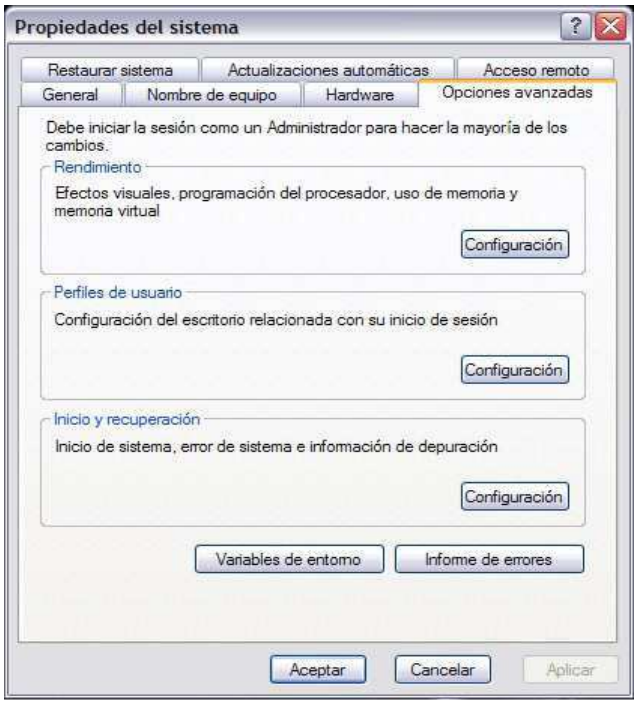


Imagen 3.3 Propiedades del sistema

- Dentro de las variables del sistema, buscamos la variable Path y hacemos doble clic sobre ella.

- En la ventana que se abrirá para modificar la variable del sistema, añadimos en el valor de la variable, la ruta donde hemos guardado la carpeta de Android SDK, incluyendo al final del directorio la carpeta 'tools' que se encuentra dentro de la misma, tal y como se muestra en la imagen.

Figura 3.4 Modificación de la variable de sistema Path

- Finalmente, aceptamos las modificaciones.

El hecho de haber añadido esta ruta a la variable PATH de nuestro equipo sirve para poder controlar el Puente de Ajuste de Android (adb) y otras herramientas de línea de comando, sin necesidad de escribir el camino entero al directorio de las herramientas. De esta forma, si tenemos que introducir comandos desde la ventana de símbolo del sistema para realizar pruebas a nuestras aplicaciones, nos va a resultar mucho más cómodo trabajar con ello.

Nota: Si alguna vez actualizamos el SDK de Android, deberemos ajustar también la nueva ruta como hemos hecho anteriormente, para indicar la nueva ubicación en caso que se haya modificado.

3.4 Instalación del Plugin ADT de Eclipse

Para descargar el Plugin ADT de Eclipse, vamos a seguir los pasos que nos dicen en la web http://developer.android.com/sdk/1.5_r2/installing.html.

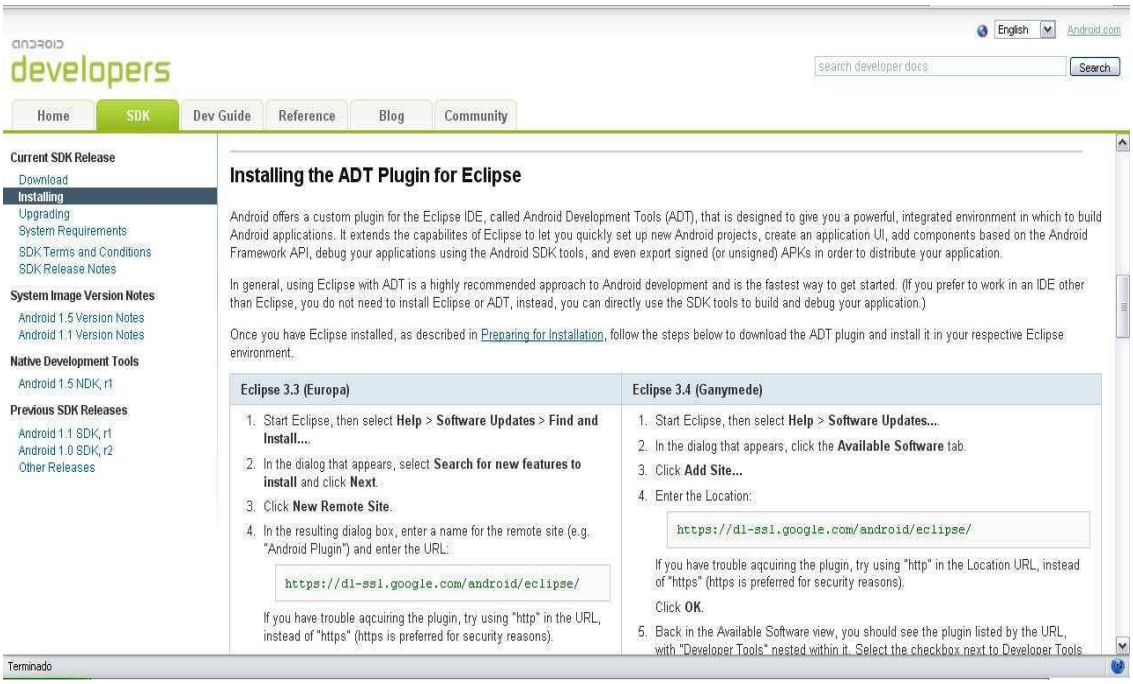


Imagen 3.5 Instalación del plugin ADT

Este Plugin incluye una variedad de extensiones que hacen más rápida y sencilla la creación, ejecución y corrección de errores de Android.

Éstos son los pasos a seguir para los usuarios de Eclipse 3.4 (Ganymede):

- Ejecutamos Eclipse, y seleccionamos **Help ->** Software Updates....

- En la ventana que aparece seleccionamos la pestaña de **Available Software** y pulsamos el botón para añadir un nuevo sitio (Add Site).

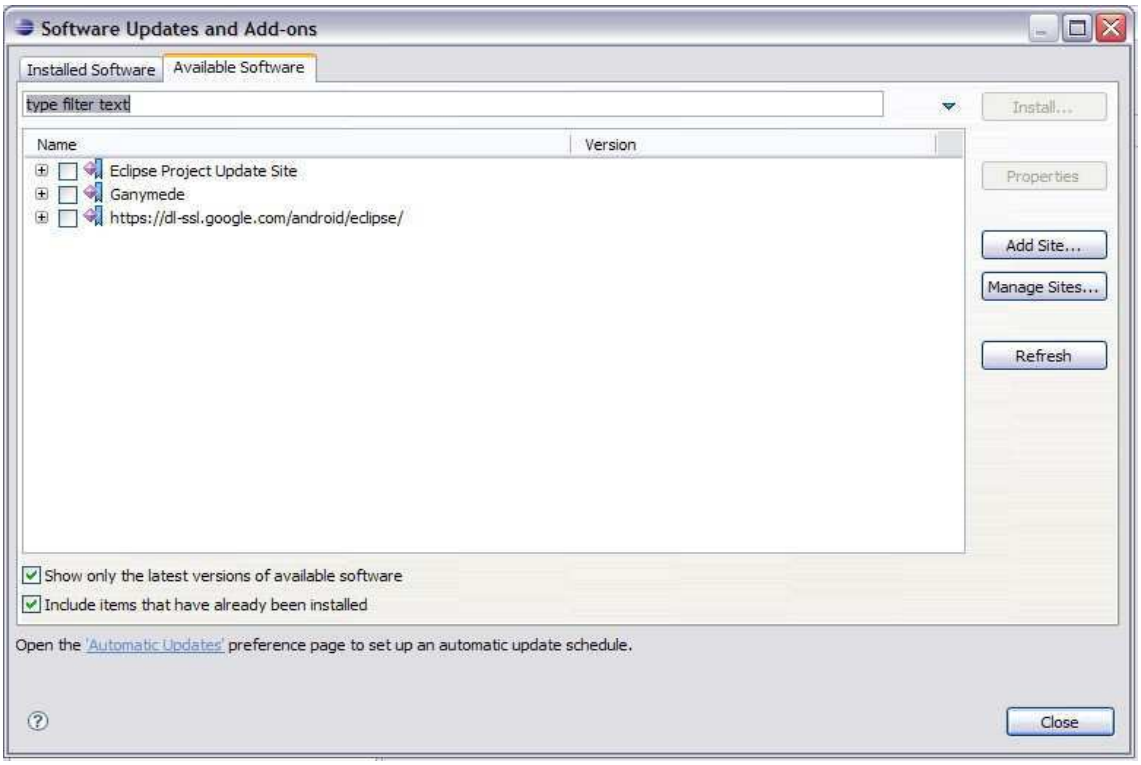


Imagen 3.6 Búsqueda de actualizaciones de Software disponible

- En el cuadro de diálogo escribimos la siguiente dirección: https://dl-ssl.google.com/android/eclipse/ y pulsamos **ok**

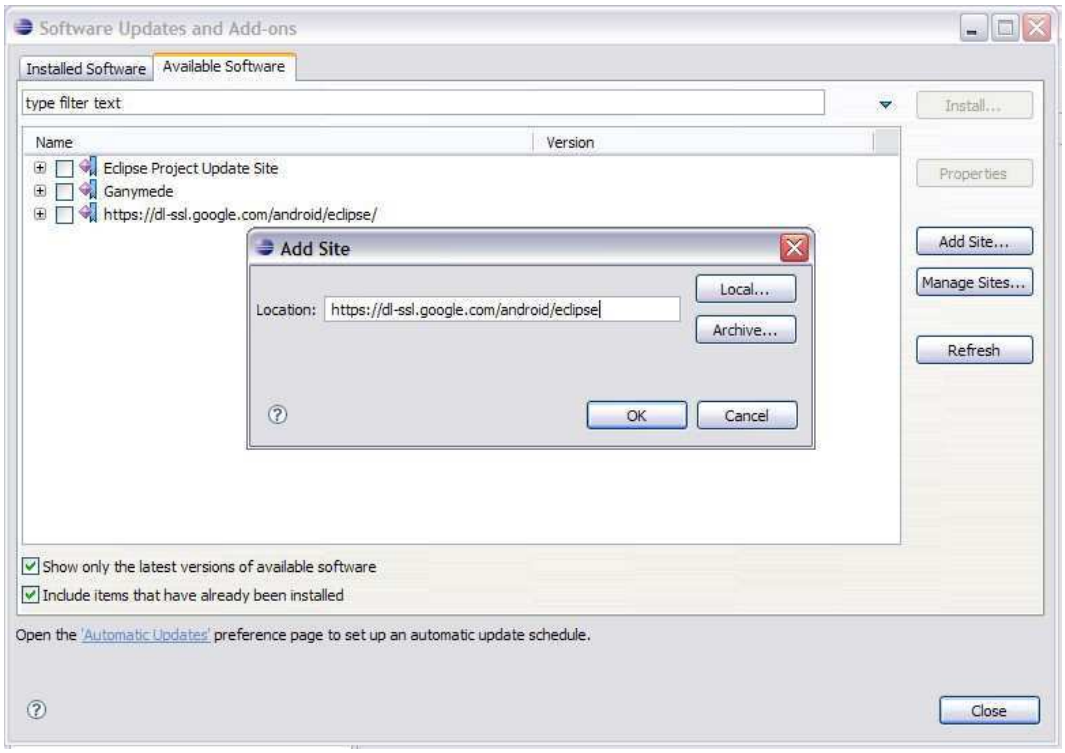


Imagen 3.7 Actualización de Software disponible

- De vuelta a la pestaña de Available Software, seleccionamos el Plugin que acabamos de añadir y pulsamos sobre **Install**

- En la ventana de instalación se van a cargar los componentes "Android Developer Tools" y "Android Editors", este último es opcional, aunque recomendado y es necesario tener el Plugin **WST**.

- Reiniciamos Eclipse.

Una vez reiniciado, actualizamos las preferencias de Eclipse siguiendo estos pasos:

- Seleccionamos Window -> Preferences
- En el panel de la izquierda, seleccionamos **Android**.

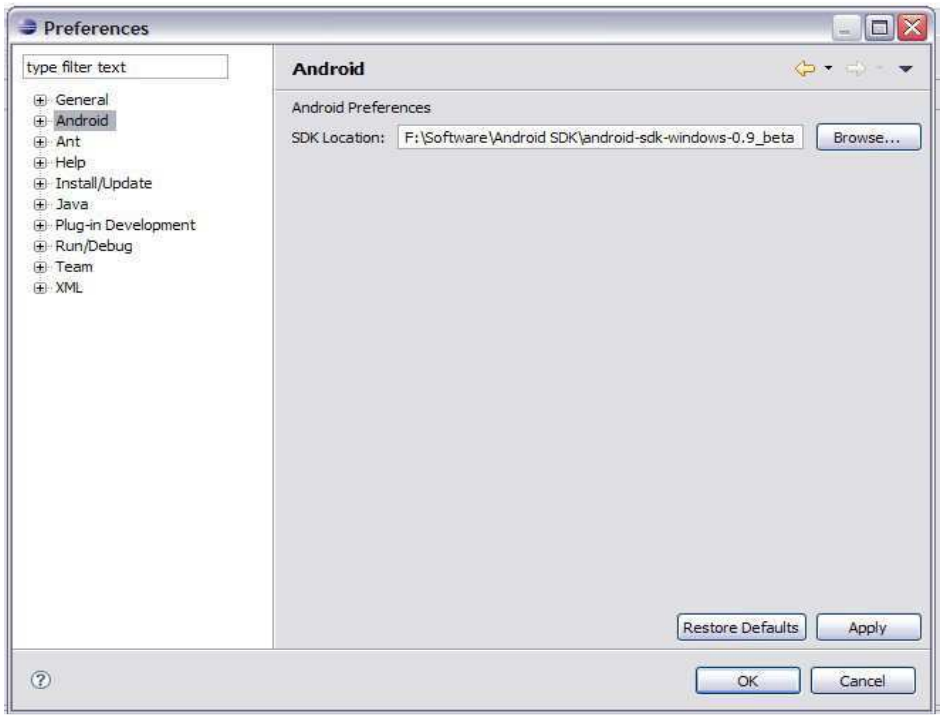


Imagen 3.8 Actualización de las preferencias de Android

- Pulsamos sobre el botón Browse para localizar el directorio donde hemos ubicado el SDK de Android.

- Aplicamos los cambios y luego pulsamos **ok**.

En algunos casos el nuevo Plugin de herramientas de desarrollo de Android (ADT) necesita actualizarse, por la posibilidad de que existan nuevas actualizaciones del SDK.

- Seleccionamos Help -> Software Update

- Seleccionamos la pestaña de "Installed Software" y pulse en Update...

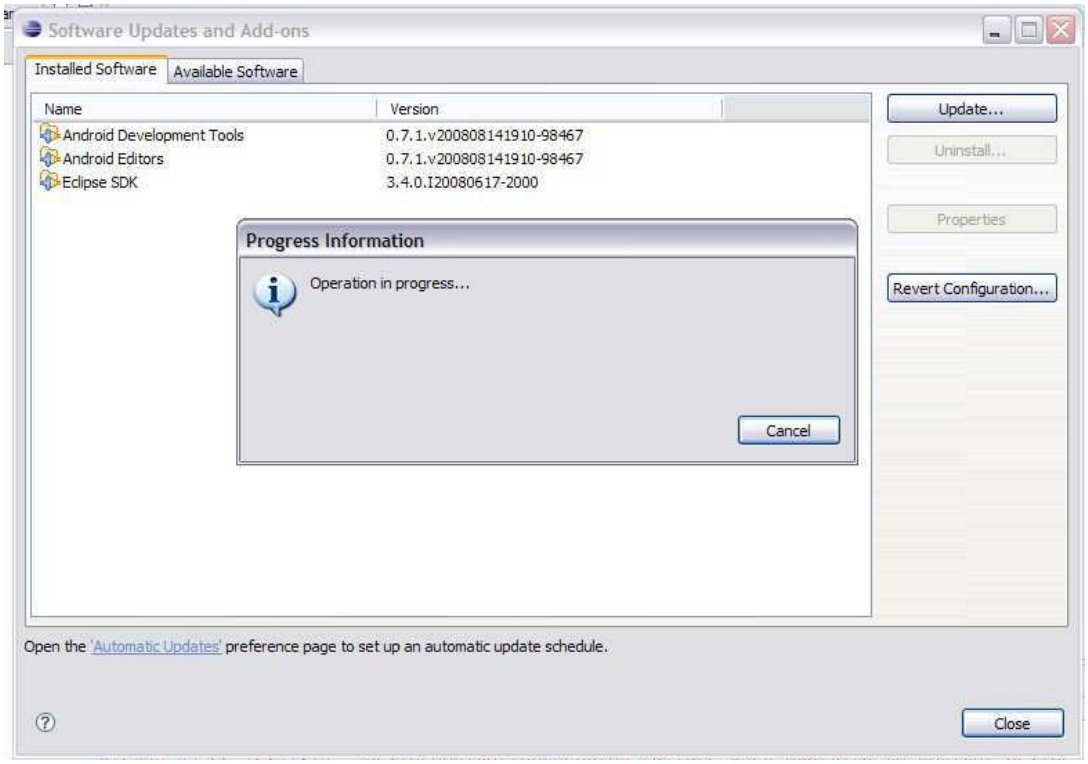


Imagen 3.9 Instalación de software actualizado

Si hay alguna actualización, la seleccionamos y finalizamos.

Ya estamos preparados para empezar a desarrollar aplicaciones con el nuevo SDK de Android.

3.5 Emulador

Una vez tengamos el proyecto listo para ejecutar, entra en escena el emulador de Android. Éste proporciona una vista especial para comprobar si la aplicación hace lo que se desea. A continuación se muestra la vista del emulador para la versión 2.2 de Android:

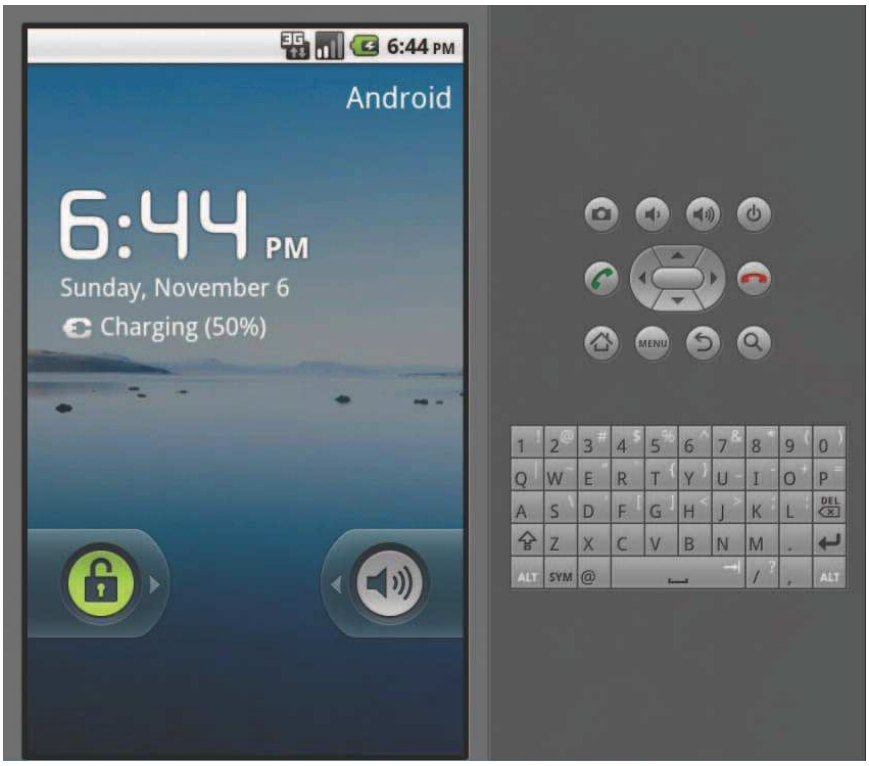


Imagen 3.10. Emulador para Android 2.2

Lo primero que hay que hacer cuando se quiere ejecutar una aplicación, es pinchar sobre el proyecto con el botón derecho, y en “Run as” seleccionar “Android Application”, entonces se lanzará el emulador más apropiado siempre que esté creado (más adelante, se explicará cómo generar los emuladores).

No se debe parar la ejecución del emulador, dado que cada vez que se ejecuta el mismo, necesita de muchos recursos del computador, por lo que tarda bastante en lanzarse, y realmente no es necesario cerrarlo, puesto que cada vez que se lleva a cabo una ejecución del proyecto, la aplicación se reinstala en el emulador.

4. Herramientas utilizadas

En este apartado describiremos todo las herramientas que se han usado para la realización del sistema.

A nivel lógico sólo habría que especificar los dos lenguajes de programación usados, en nuestro caso: XML y Java. XML, han sido utilizados en archivos como "AndroidManifest.xml". También se usa en otros archivos de definición de datos que harán falta especificar ("strings.xml", etc.). Igualmente, se ha usado Java para el desarrollo de la implementación de nuestra aplicación (todas las clases, paquetes, etc.). Ya dentro de la propia aplicación se ha utilizado el estándar JSON para el intercambio de datos entre la aplicación Android y el servicio Web.

Finalmente a estas tecnologías habría que añadir el uso de una base de datos MYSQL y de un servicio web escrito en lenguaje PHP.

A nivel físico, a continuación se muestran las características del dispositivo móvil (Samsung Galaxy Mini), y las características de los elementos indispensables que se han usado y que están incorporados en el dispositivo, como son el chip receptor GPS y la comunicación por datos.

4.1 Dispositivo Móvil Usado (Samsung Galaxy Mini S5570)

Este dispositivo tiene un procesadorQualcomm® MSM7227™, 600 MHz y una plataforma Android™ 2.3.6 (Gingerbread), posee una memoria ROM de 512 Mb y una RAM de 384 Mb.

El dispositivo posee una pantalla táctil TFT-LCD de 3,14 pulgadas con resolución QVGA 240 x 320 píxeles. Tiene cámara en color de3.15 megapíxeles y sistema de localización GPS con soporte A-GPS perfecto para complementar el servicio de GoogleMaps.

Es muy ligero, con unas dimensiones de 110,4 x 60,6 x 12,1 mm, y con un peso de 105 gramos, incluida la batería recargable de iones de litio.

Tiene GPRS o servicio general de paquetes vía radio para utilizar los servicios tales como Wireless Application Protocol(WAP), servicio de mensajes cortos (SMS), servicio de mensajería multimedia (MMS), Internet y para los servicios de comunicación, como el correo electrónico y la World Wide Web (WWW).

En lo que se refiere a la conectividad, se dispone de tecnología 3G, que junto con la tecnología HSDPA (High Speed Downlink Packet Access) mejoran notablemente la velocidad de transmisión de la red, y WiFi 802.11b/g para conectarse a cualquier red inalámbrica que esté al alcance. También tiene Bluetooth® 2.0 con EDR y A2DP para auriculares inalámbricos estéreo y HTC ExtUSB™ (conector mini USB 2.0 de 11 patillas).

En la parte superior del móvil, posee un led de notificaciones en que, según el tipo de aviso, tiene una forma distinta de iluminación. En la parte inferior se ubican los botones de llamada y de acceso al menú, junto con una bola de navegación, y en la esquina inferior izquierda se encuentra la ranura para almacenar la tarjeta de memoria microSD. Debajo del móvil se encuentra un conector USB para poder cargar la batería y poder conectarlo al PC y realizar las gestiones de almacenamiento de datos y aplicaciones.

Imagen 4.1 Dispositivo Móvil Samsung Galaxy Mini S5570

En la pantalla principal tenemos un área de notificaciones, donde nos aparecerán todos los avisos que vayamos recibiendo, ya sean tipos de llamadas, SMS/MMS o e-mails recibidos, descargas de aplicaciones, conexión USB, detección de redes, aviso de eventos, etc. La pantalla principal está ampliada y podemos verla por partes haciendo movimientos hacia izquierda y derecha con el dedo.

Podemos configurar dicha pantalla agregando accesos directos a aplicaciones, widgets (agregar un reloj, un marco de la imagen, una barra de búsqueda de Google...) o cambiando el fondo de pantalla. La orientación de la pantalla cambia de horizontal a vertical según como tengamos colocado el dispositivo.

En lo que se refiere al software que viene instalado de serie, mediante Gmail podemos crear o configurar nuestra propia cuenta, si ya la tenemos creada, y poder gestionarla como si de un ordenador se tratara, recibiendo, enviando y editando la configuración de nuestra cuenta de correo. Gmail nos ofrece acceso a los mismos correos electrónicos que creamos, vemos y editamos en nuestro ordenador utilizando Gmail en la Web.

Esto significa que cuando cambiemos o eliminamos cualquier dato en Gmail en la Web, la información actualizada aparecerá en nuestro dispositivo, y viceversa. Esto mantiene nuestro correos electrónicos intactos de manera que si perdemos nuestro dispositivo o si éste se destruye, nuestros correos electrónicos aparecerán como

previamente, en un dispositivo de sustitución. Esta información a modo de copia se puede realizar mediante la sincronización de datos a través del aire. La sincronización de datos tiene lugar en un segundo plano y no interfiere en nuestra actividad. Nos daremos cuenta de que los datos se están sincronizando cuando veamos el correspondiente icono de sincronización de datos en la barra de estado.

También posee un navegador para visitar los sitios web deseados, cumpliendo la función de un navegador de cualquier PC normal.

En cuanto a mapas se refiere, dispone del servicio Google Maps con todas sus funcionalidades para encontrar cualquier ubicación, mapas vía satélite, callejero, mapas de tráfico, indicaciones para llegar a cualquier parte, modalidad Street View y Latitude, que permite compartir nuestra ubicación y la de nuestros amigos con el fin de estar localizado para quien tu desees (pudiéndose configurar tu privacidad en todo momento). Todo eso haciendo uso del servicio GPS que tan importante va a ser para nuestro proyecto.

También disponemos del servicio YouTube, que gracias a la asociación con Google, podemos disfrutar de todos los vídeos que allí disponen y podemos buscarlos en nuestro móvil y verlos en la mejor calidad posible.

Aparte de todas estas novedades, el dispositivo incorpora los servicios típicos que ofrece todo teléfono móvil, como agenda, alarma, calendario, calculadora, cámara de fotos y de vídeo, mensajería SMS/MMS y gestión de imágenes y música que se pueden descargar en nuestra tarjeta microSD.

Este dispositivo también tiene otras características especiales como radio FM, brújula digital y acelerómetro. A estos dos últimos les dedicaremos un apartado especial debido a la importancia que tienen para nuestro proyecto y también por la importancia de conocer detalladamente cómo se usarán en el localizador.

Además, hay otro servicio importantísimo que es Android Market, donde podemos encontrar todas las aplicaciones que los usuarios han creado y han decidido compartir para que todo el mundo pueda usarlas, aprovechando la definición de software libre y haciendo aún más grande la revolución que ya está causando Android. Entre las aplicaciones, podemos encontrar muchísimas gratuitas, como un escaneador de códigos de barras, servicio móvil deeBay, servicios de comunicación como Skype, Live Chat, Hi MSN, herramientas de finanzas, bloc de notas, aplicaciones multimedia, noticias y meteorología, ocio, productividad, viajes, bibliotecas de software, multitud de juegos y cualquier aplicación que nos pueda pasar por nuestra imaginación y queramos subirla a Android Market para que todos los usuarios puedan disfrutar de ella.

No podemos olvidarnos de la tecnología de comunicación inalámbrica de corto alcance, Bluetooth, que permite intercambiar información a una distancia de aproximadamente 8 metros sin precisar una conexión física.

En definitiva, la tecnología Android va en camino de convertirse en una herramienta de máxima importancia para todos los dispositivos móviles, y cada vez son más las empresas dedicadas al diseño de estos dispositivos (Nokia, Sony Ericsson, Samsung, HTC) que están preparando sus dispositivos con Android. Incluso se habla de que los ordenadores personales también trabajarán en un futuro próximo bajo el sistema operativo Android. Así que la competencia va cada vez más lejos de la mano de Google.

4.1.1 Sensores soportados

Cada vez es más frecuente encontrar dispositivos móviles con distintos tipos de sensores. La razón fundamental es que nos proporcionan varios tipos de medidas de nuestro entorno, pudiendo adaptar las aplicaciones del terminal al mismo. Esto significa que el usuario puede adentrarse en una nueva experiencia nunca antes vivida con, por ejemplo, su telefono móvil, descubriendo que en su entorno puede haber mucha más información, diversión o mucho más entretenimiento de lo que ve a simple vista.

Android destaca por su soporte para múltiples sensores. En este apartado nos centraremos en aquellos que son relevantes para la localización, es decir, el GPS.

4.1.1.1 GPS

Actualmente, es bastante común encontrar un dispositivo GPS integrado en los terminales móviles de nueva generación. Este sensor nos proporcionará nuestra Información GPS, como las coordenadas de nuestra posición, su precisión, la altitud sobre el nivel del mar, los satélites que estamos escuchando...Es, por tanto, el principal elemento para proporcionar servicios basados en datos de localización, como navegadores, redes sociales móviles o buscadores de recursos según su posición.

El Sistema de Posicionamiento Global (NAVSTAR-GPS o más comúnmente GPS) es el más común y conocido a la hora de enfrentarse al problema de localizar un dispositivo móvil. Se trata de una constelación de 24 satélites que giran alrededor de la Tierra a una distancia de 20200 Km, con trayectorias sincronizadas para cubrir toda la superficie del planeta.

Para su funcionamiento, el terminal del usuario debe disponer básicamente de una antena y un módulo GPS. Con estos elementos, el dispositivo es capaz de recibir los datos que envían los satélites que tenga en línea de visión, que son fundamentalmente la posición del satélite y una marca de tiempo. Sincronizando esas marcas de tiempo se puede ver el retardo que han sufrido las señales hasta llegar al terminal, y por tanto la distancia a la que se encuentran los satélites. Después, triangulando según esas distancias se puede hallar la posición del dispositivo. El número de satélites necesarios para conseguir el punto en el que nos encontramos es variable, según la calidad del módulo GPS del terminal: teóricamente

son necesarios 3, pero en la práctica pueden ser necesarios hasta 9 satélites para obtener una buena aproximación.

En cuanto a los datos que proporciona el sistema son varios, entre los que se encuentran las coordenadas geográficas (latitud y longitud) de nuestra posición, la altura sobre el nivel del mar a la que nos hallamos, marcas de tiempo, etc. La precisión de las coordenadas geográficas en condiciones óptimas tiene un error de aproximadamente 15 metros, siendo la tecnología comercial más precisa que podemos encontrar, funcionando por sí sola.

El sistema GPS es, por tanto, ideal para establecer la posición de un dispositivo, ya que prácticamente en cualquier espacio abierto tenemos cobertura. Sin embargo, el sistema GPS adolece de un problema importante en núcleos urbanos y lugares montañosos, y es que en situaciones en las que permanecemos rodeados de edificios u obstáculos altos en general, no disponemos de línea de visión directa con el número suficiente de satélites necesarios para fijar nuestra posición. Por tanto, el sistema no es suficiente para ello. Esto puede solucionarse con el sistema A-GPS, como veremos más adelante.

Por último, otro problema al que nos enfrentamos con esta tecnología es el del consumo de batería. Los dispositivos móviles se caracterizan por alimentarse de una batería. Los requisitos de ligereza hacen que las mismas deban ser del mínimo tamaño posible con un rendimiento adecuado. El contratiempo se produce porque el modulo GPS tiene un consumo de batería elevado, limitando la duración de operatividad del terminal, que, en algunos casos como en los teléfonos, es un factor crítico: Podemos quedarnos incomunicados. Para paliar esto se han desarrollado algoritmos que ahorran batería, conectando el GPS sólo cuando sea estrictamente necesario.

4.2 Base de datos MYSQL

Se ha decidido usar este motor de base de datos dada su rapidez y la facilidad de programación.

Para la realización del proyecto se ha contratado un alojamiento con un proveedor de servicio este alojamiento incluye una base de datos MYSQL y espacio web para depositar los archivos que conforman el servicio web.

En la siguiente imagen se muestra el panel de control del alojamiento desde el cual se accede a la base de datos, se gestionan las cuentas de correo, las cuentas ftp..

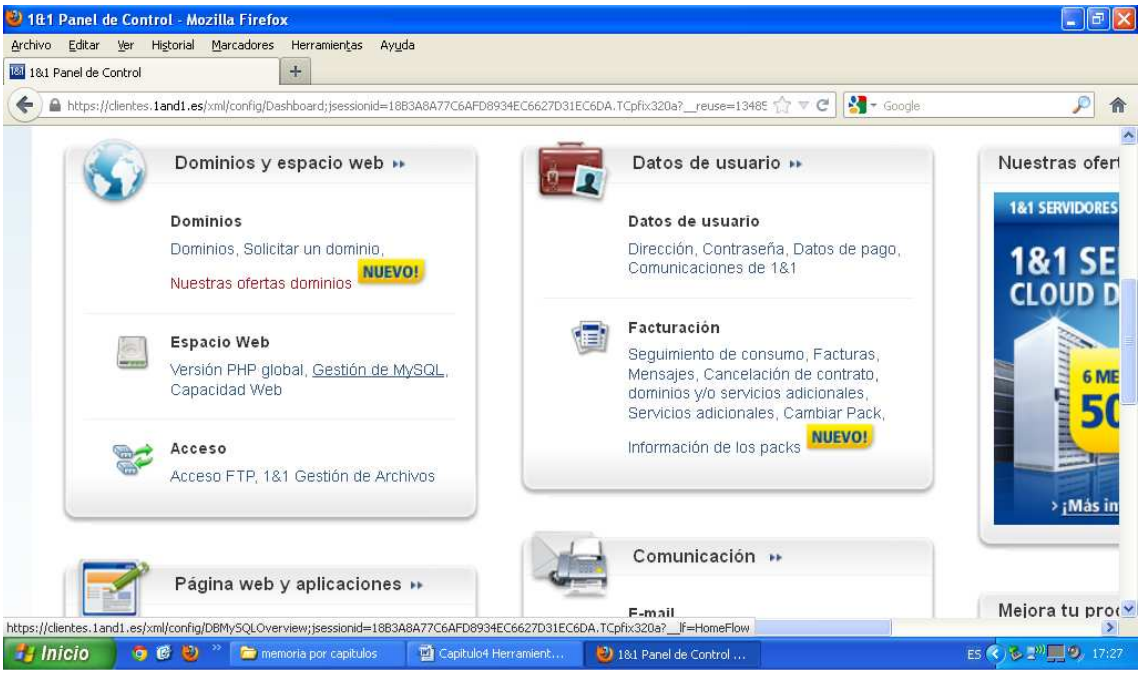


Imagen 4.2 Panel de control principal del alojamiento

Como toda base de datos MYSQL disponemos del conocido panel de administración phpMyAdmin:

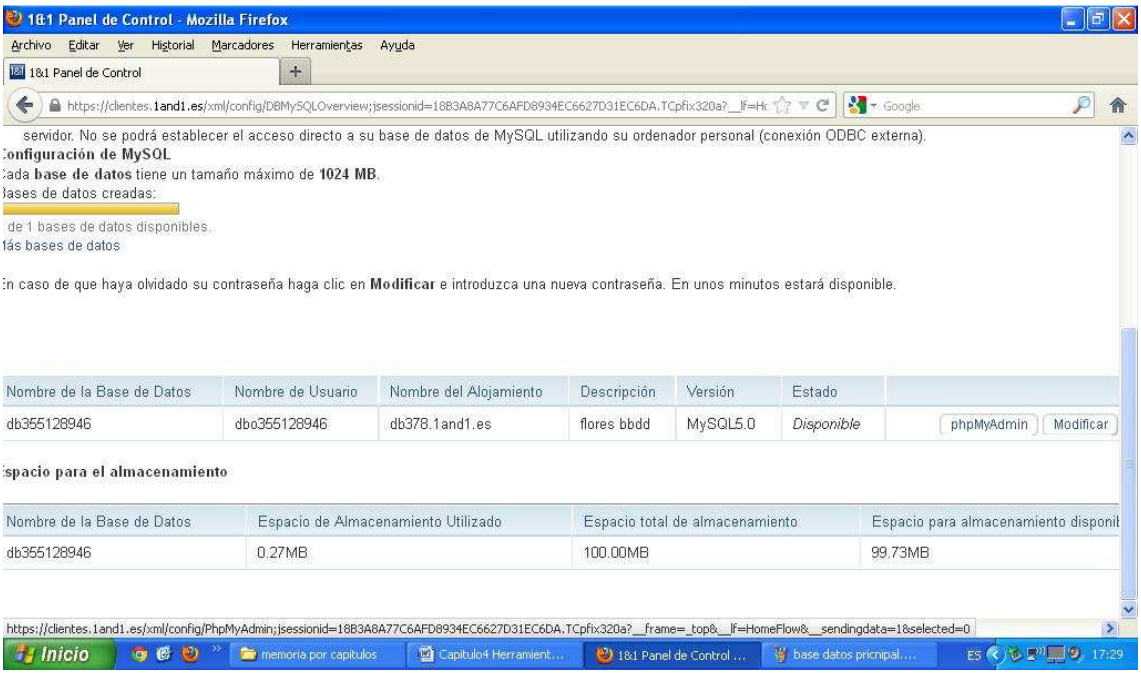


Imagen 4.3 Zona de administración principal de la base de datos

Una vez que pulsamos el botón phpMyAdmin accedemos a la administración de la base de datos:

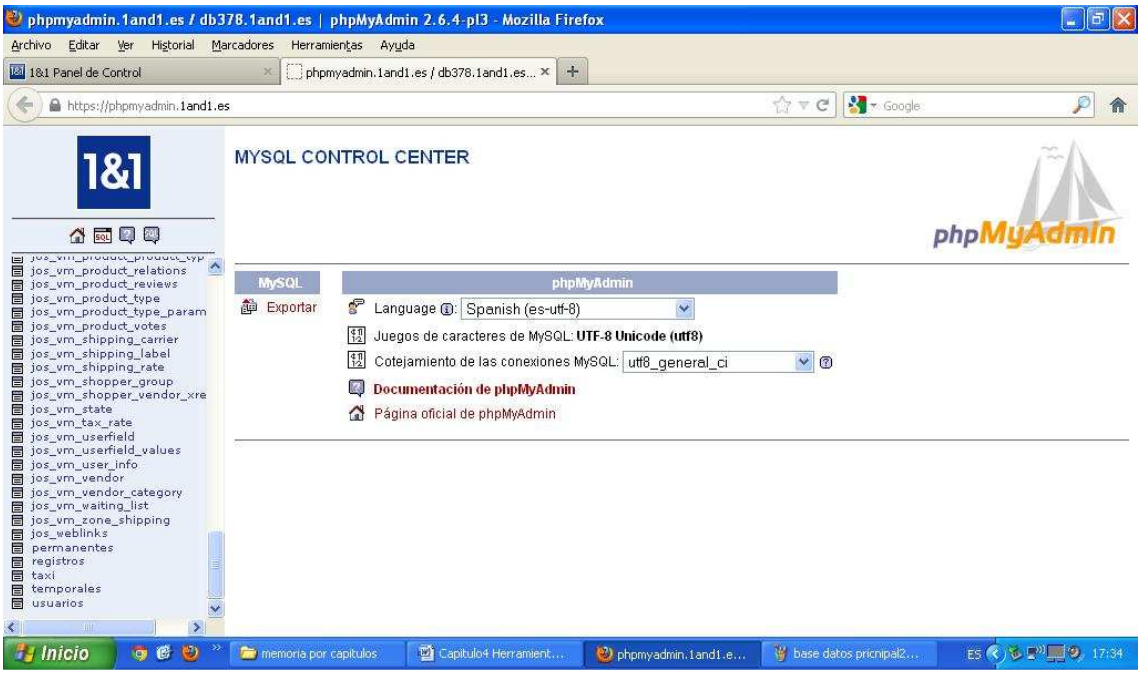


Imagen 4.4 phpMyadmin

4.3 Espacio Web

Como se ha comentado anteriormente este alojamiento incluye 1 GB de espacio web en el que se han introducidos los archivos que forman la aplicación web "Centro de Control", además de una serie de archivos con extensión PHP que conforman el llamado Servicio Web mediante el cual se insertar, actualizan, consultan o se borran datos en la base de dato.

Para subir los archivos a este espacio web es necesario dar de alta una cuenta ftp en el panel principal mostrado en la figura 4.2. Una vez que se ha dado de alta una cuenta FTP se hace uso de un programa cliente FTP en este caso se ha utilizado Filezilla, para subir los archivos al servidor.

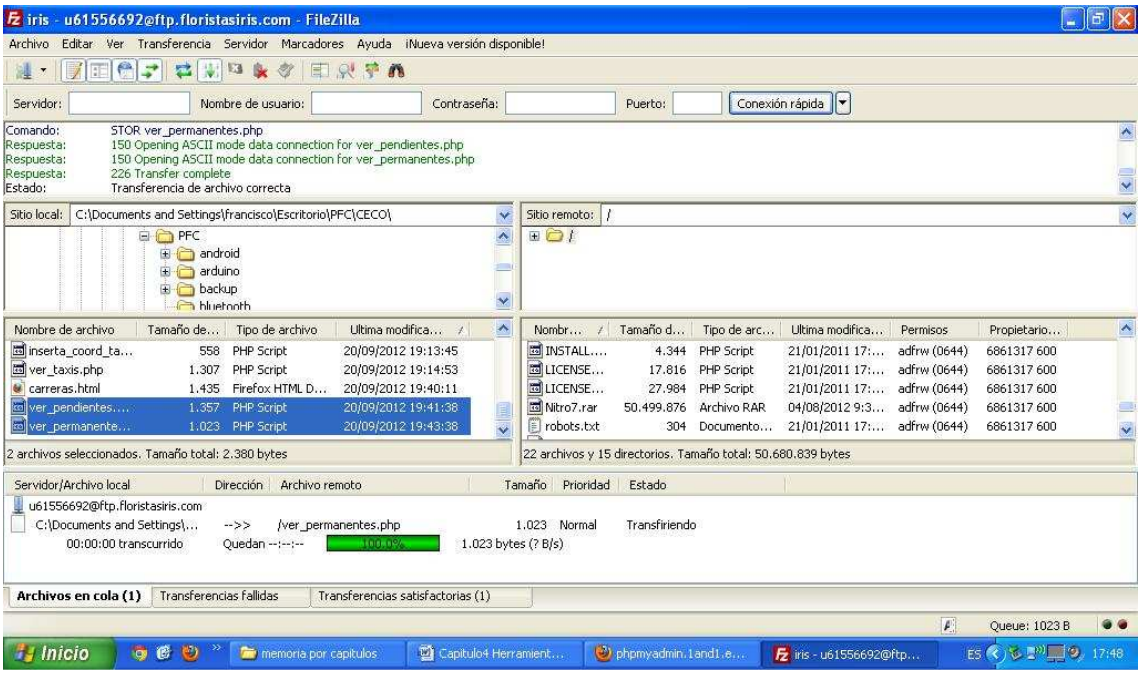


Imagen 4.5 Subida de archivos al servidor con Filezilla

5. Pruebas realizadas

Para llevar a cabo la realización de este sistema se han implementado varias aplicaciones cada una de las cuales busca el desarrollo y estudio de una funcionalidad del sistema.

En la siguiente figura se muestra una captura del software eclipse; a la izquierda se puede ver el “explorador de proyectos” en el que se muestran algunas de las aplicaciones realizadas.

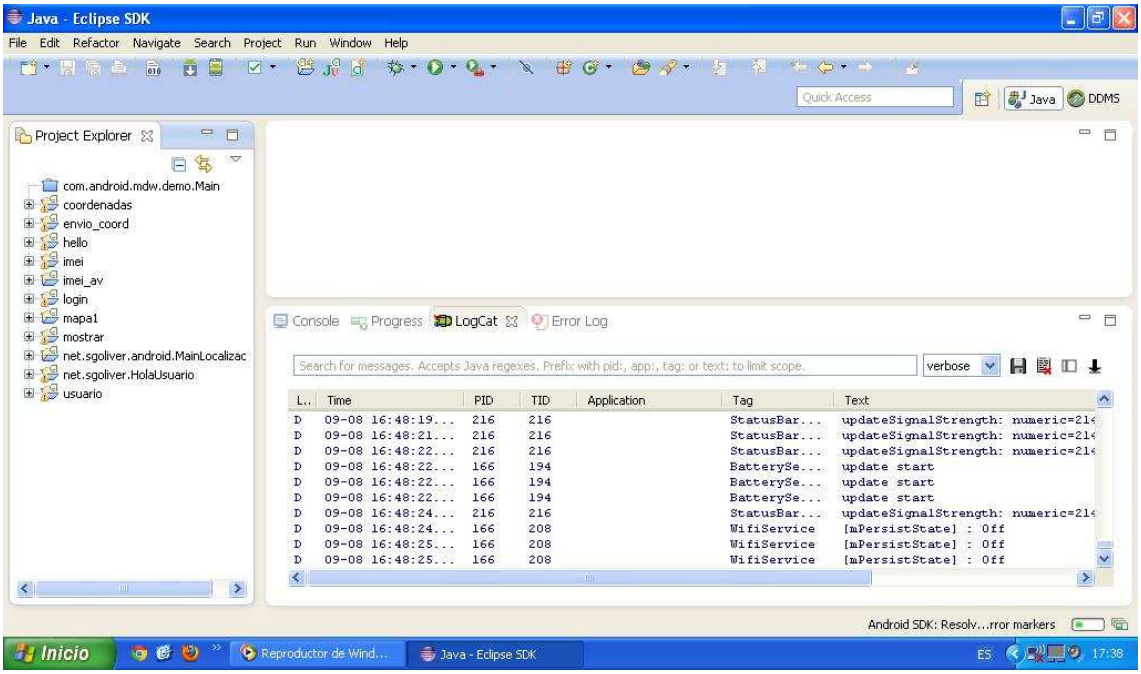


Imagen 5.1 Explorador de proyectos de Eclipse

5.1 Extracción e impresión por pantalla de datos del GPS.

Para que este sistema funcione necesitamos saber las coordenadas GPS del usuario. En esta aplicación se ha experimentado con la obtención de estas coordenadas obteniendo resultados muy a tener en cuenta ya que van a definir el funcionamiento de la aplicación cliente como mas adelantará se explicará en el apartado conclusiones

5.1.1 Código fuente de la aplicación "MainLocalizacion.java"

```
package net.localizacion.android;

import android.app.Activity;
import android.content.Context;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;

public class MainLocalizacion extends Activity {

        private Button btnActualizar;
        private Button btnDesactivar;
        private TextView lblLatitud;
        private TextView lblLongitud;
        private TextView lblPrecision;
        private TextView lblEstado;

        private LocationManager locManager;
        private LocationListener locListener;

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    btnActualizar = (Button)findViewById(R.id.BtnActualizar);
    btnDesactivar = (Button)findViewById(R.id.BtnDesactivar);
    lblLatitud = (TextView)findViewById(R.id.LblPosLatitud);
    lblLongitud = (TextView)findViewById(R.id.LblPosLongitud);
    lblPrecision = (TextView)findViewById(R.id.LblPosPrecision);
    lblEstado = (TextView)findViewById(R.id.LblEstado);

    btnActualizar.setOnClickListener(new OnClickListener() {
                        @Override
                        public void onClick(View v) {
                                comenzarLocalizacion();
                        }
                });
```

```
   btnDesactivar.setOnClickListener(new OnClickListener() {
                        @Override
                        public void onClick(View v) {
                        locManager.removeUpdates(locListener);
                        }
                });
}

private void comenzarLocalizacion()
{
        //Obtenemos una referencia al LocationManager
        locManager =
                (LocationManager)getSystemService(Context.LOCATION_SERVICE);

        //Obtenemos la última posición conocida
        Location loc =
                locManager.getLastKnownLocation(LocationManager.STRING_PROVEEDOR);

        //Mostramos la última posición conocida
        mostrarPosicion(loc);

        //Nos registramos para recibir actualizaciones de la posición
        locListener = new LocationListener() {
                public void onLocationChanged(Location location) {
                        mostrarPosicion(location);
                }
                public void onProviderDisabled(String provider){
                        lblEstado.setText("Provider OFF");
                }
                public void onProviderEnabled(String provider){
                        lblEstado.setText("Provider ON ");
                }
                public void onStatusChanged(String provider, int status, Bundle extras){
                        Log.i("", "Provider Status: " + status);
                        lblEstado.setText("Provider Status: " + status);
                }
        };

        locManager.requestLocationUpdates(
                        LocationManager.STRING_PROVEEDOR, 30000, 0, locListener);
}

private void mostrarPosicion(Location loc) {
        if(loc != null)
        {
                lblLatitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
                lblLongitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));
                lblPrecision.setText("Precision: " + String.valueOf(loc.getAccuracy()));
                Log.i("", String.valueOf(loc.getLatitude() + " - " + String.valueOf(loc.getLongitude())));
        }
        else
        {
                lblLatitud.setText("Latitud: (sin_datos)");
                lblLongitud.setText("Longitud: (sin_datos)");
                lblPrecision.setText("Precision: (sin_datos)");
        }
}
```

```
}
```

*La cadena STRING_PROVEEDOR puede tomar los siguientes valores:

NETWORK_PROVIDER: localización por red GSM o WIFI

GPS_PROVIDER: localización mediante satélites GPS

5.1.2 Archivo de configuración de la aplicación "Androidmanifest.xml"

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

  <uses-sdk android:minSdkVersion="7" />

  <application android:icon="@drawable/icon" android:label="@string/app_name">
    <activity android:name=".MainLocalizacion"
          android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>

  </application>

  <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
  <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

</manifest>
```

5.1.3 Archivo de interfaz de la aplicación "main.xml"

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent" >

  <TextView android:id="@+id/LblPosicion"
          android:layout_width="fill_parent"
          android:layout_height="wrap_content"
          android:layout_margin="5dip"
          android:text="Posición Actual:" />

        <TextView android:id="@+id/LblPosLatitud"
          android:layout_width="fill_parent"
          android:layout_height="wrap_content"
          android:layout_margin="5dip"
          android:background="#aaaaaa"
```

```
        android:textColor="#000000" />

    <TextView android:id="@+id/LblPosLongitud"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dip"
        android:background="#aaaaaa"
        android:textColor="#000000" />

    <TextView android:id="@+id/LblPosPrecision"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dip"
        android:background="#aaaaaa"
        android:textColor="#000000" />

    <Button android:id="@+id/BtnActualizar"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Activar" />

    <Button android:id="@+id/BtnDesactivar"
            android:layout_width="fill_parent"
            android:layout_height="wrap_content"
            android:text="Desactivar" />

    <TextView android:id="@+id/LblEst"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dip"
        android:text="Estado proveedor:" />

    <TextView android:id="@+id/LblEstado"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dip"
        android:background="#aaaaaa"
        android:textColor="#000000" />

</LinearLayout>
```

5.1.4 Captura de pantalla aplicación en marcha

Imagen 5.2 Obtención de coordenadas

Como puede apreciarse en la figura, con esta aplicación se obtienen las coordenadas y además hay un tercer campo llamado precisión que mide la calidad de la medida efectuada, tras los botones de activar-desactivar se encuentra el último campo "Estado proveedor" que nos indicará que el receptor GPS está apagado o bien no hay disponible ninguna conexión de datos.

5.1.5 Estudio y análisis del GPS

Con la realización de esta aplicación se ha aprendido que hay dos formas posibles de geolocalización; la primera mediante el chip receptor GPS incorporado en el dispositivo móvil y la segunda, mediante una conexión de datos bien sea WIFI o mediante la red GSM

5.1.5.1 Características del Chip GPS en Geolocalización

Para hablar del funcionamiento del este circuito vamos a distinguir dos situaciones posibles; la primera es estar cuando el dispositivos se encuentra al aire libre y la segunda cuando el dispositivo está en el interior de un edificio o similar. La cobertura de los satélites GPS en el interior de edificios es escasa y en muchos casos nula como se ha comprobado experimentalmente con esta aplicación. Al aire libre la cobertura es máxima y el grado de precisión mayor que a través de localización por conexión de datos.

5.1.5.2 Características de la red GSM y conexión WIFI en Geolocalización

Tanto una como otra resuelven el problema del chip receptor GPS en interiores ya que sí hay cobertura.

En interior mediante conexión WIFI se consigue una grado de precisión mucho mayor que a través de la red GSM que en muchos casos tienes errores te hasta varias decenas de metros.

5.1.6 Conclusiones

En vista de los resultados de las pruebas realizadas con una y otra forma de obtención de coordenadas se determina la siguiente configuración:

-La aplicación taxista obtendrá sus coordenadas mediante el Chip GPS con el fin de obtener unas coordenadas muy precisas ya que la mayor parte del tiempo el taxi se encuentra al aire libre o en el peor de los casos rodeado de edificios muy altos.

-La aplicación da la opción al usuario a elegir si usar satélites GPS o redes de datos para ser localizados en función de si este se encuentra en un interior o al aire libre.

5.2 Obtención del número IMEI del terminal.

La forma de identificación de cada usuario se hará a través del identificador IMEI del terminal mediante el cual se puede controlar que usuarios siguen en el sistema y también es útil para gestionar el intercambio de mensajes entre aplicaciones. En el sistema se hará uso del identificador IMEI para las siguientes tareas:

- Identificación de clientes
- Identificación de taxista
- Gestionar petición de cliente a taxi
- Gestionar respuesta de taxi a cliente
- Gestionar desconexión de clientes y taxistas.

5.2.1 Código fuente de la aplicación “ImeiActivity.java”

```
package net.imei.android;

import android.os.Bundle;
import android.app.Activity;
import android.telephony.TelephonyManager;
import android.widget.TextView;
import android.widget.Toast;
```

```
public class imeiActivity extends Activity {
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_imei);
     TelephonyManager tm = (TelephonyManager)
getSystemService(imeiActivity.this.TELEPHONY_SERVICE);
      String imei = tm.getDeviceId();
      Toast.makeText(getApplicationContext(),imei, Toast.LENGTH_LONG).show();
   }


}
```

5.2.2 Archivo de configuración de la aplicación "AndroidManifest.xml"

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="net.imei.android"
   android:versionCode="1"
   android:versionName="1.0" >
   <uses-sdk
      android:minSdkVersion="7"
      android:targetSdkVersion="15" />
   <application
      android:icon="@drawable/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >
      <activity
         android:name=".imeiActivity"
         android:label="@string/title_activity_imei" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"></uses-permission>
</manifest>
```

5.2.3 Archivo de interfaz de la aplicación "activity_imei.xml"

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent" >

   <TextView
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_centerHorizontal="true"
      android:layout_centerVertical="true"
      android:padding="@dimen/padding_medium"
      android:text="@string/hello_world"
```

```
    tools:context=".imeiActivity" />

</RelativeLayout>
```

5.2.4 Captura de pantalla aplicación en marcha

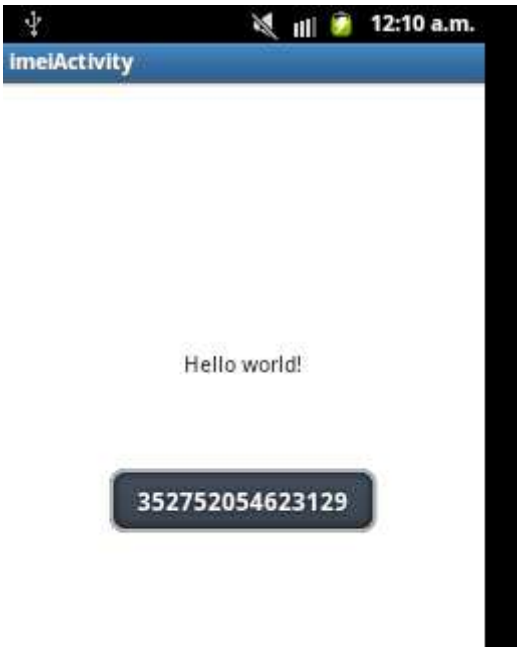


Imagen 5.3 Obtención del identificador IMEI

5.2.5 Conclusiones

¿Qué es el IMEI?

El IMEI es un identificador único de dispositivo asignado por el proveedor con el fin de identificar unívocamente al l terminal.

¿Por qué usar el IMEI y no el número de teléfono?

1. El IMEI es un identificador a nivel mundial.

2. Algunas compañías no registran el número de teléfono en la tarjeta SIM de manera que se hace imposible acceder a este número mediante programación en Android

5.3 Aplicación inicial en el manejo de Google Maps

Una vez que se tiene el IMEI y las coordenadas el siguiente tema a investigar será el manejo de la API de Google Maps. En esta primera aplicación se ha cargado un mapa con tres puntos introducidos como constantes en el código de la aplicación.

Para llevar a cabo esta tarea es necesario registrarse como desarrollador de aplicaciones Google Maps para obtener así la firma digital o API Google Key con la que firmaremos nuestra aplicación de manera que esta sea reconocida por Google y nos permita hacer uso de sus mapas. Para más información acerca de cómo obtener una API Google Key revise el anexo I "Obtener una API Key de Google".

5.3.1 Código fuente de la aplicación "ImeiActivity.java"

```
package net.mapa1.android;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapView;
import android.graphics.drawable.Drawable;
import android.os.Bundle;

public class mapa1Activity extends MapActivity {
 /** Called when the activity is first created. */
 @Override
 public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_mapa1);
    MapView mapView = (MapView) findViewById(R.id.mapa);
    mapView.setBuiltInZoomControls(true);


    Drawable marker=getResources().getDrawable(android.R.drawable.star_big_on);
    int markerWidth = marker.getIntrinsicWidth();
    int markerHeight = marker.getIntrinsicHeight();
    marker.setBounds(0, markerHeight, markerWidth, 0);


    MyItemizedOverlay myItemizedOverlay = new MyItemizedOverlay(marker);
    mapView.getOverlays().add(myItemizedOverlay);

    GeoPoint myPoint1 = new GeoPoint(37986348,-1132332);
    myItemizedOverlay.addItem(myPoint1, "taxi1", "taxi1");
    GeoPoint myPoint2 = new GeoPoint(39020000, 2290000);
    myItemizedOverlay.addItem(myPoint2, "taxi2", "taxi2");
    GeoPoint myPoint3 = new GeoPoint(37942176,-1138174);
    myItemizedOverlay.addItem(myPoint3, "taxi3", "taxi3");

    }

@Override
protected boolean isLocationDisplayed() {
// TODO Auto-generated method stub
return false;
}
```

```
@Override
protected boolean isRouteDisplayed() {
// TODO Auto-generated method stub
return false;
}

}
```

5.3.2 Código fuente de la clase "myItemizedOverlay"

```
 package net.mapa1.android;
import java.util.ArrayList;

import android.graphics.Canvas;
import android.graphics.drawable.Drawable;

import com.google.android.maps.GeoPoint;
import com.google.android.maps.ItemizedOverlay;
import com.google.android.maps.MapView;
import com.google.android.maps.OverlayItem;

public class MyItemizedOverlay extends ItemizedOverlay<OverlayItem>{

private ArrayList<OverlayItem> overlayItemList = new ArrayList<OverlayItem>();

public MyItemizedOverlay(Drawable marker) {
super(boundCenterBottom(marker));
// TODO Auto-generated constructor stub

populate();
}

public void addItem(GeoPoint p, String title, String snippet){
OverlayItem newItem = new OverlayItem(p, title, snippet);
overlayItemList.add(newItem);
  populate();
}

@Override
protected OverlayItem createItem(int i) {
// TODO Auto-generated method stub
return overlayItemList.get(i);
}

@Override
public int size() {
// TODO Auto-generated method stub
return overlayItemList.size();
}

@Override
public void draw(Canvas canvas, MapView mapView, boolean shadow) {
// TODO Auto-generated method stub
super.draw(canvas, mapView, shadow);
```

```
//boundCenterBottom(marker);
}

}
```

5.3.3 Archivo de configuración de la aplicación "AndroidManifest.xml"

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="net.mapa1.android"
   android:versionCode="1"
   android:versionName="1.0" >

   <uses-sdk
      android:minSdkVersion="8"
      android:targetSdkVersion="15" />
    <uses-permission android:name="android.permission.INTERNET"/>
            <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
            <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
   <application
      android:icon="@drawable/ic_launcher"
      android:label="@string/app_name"
      android:theme="@style/AppTheme" >

      <activity
         android:name=".mapa1Activity"
         android:label="@string/title_activity_mapa1" >
         <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
         <uses-library android:name="com.google.android.maps" />

   </application
</manifest>
```

5.3.4 Archivo de interfaz de la aplicación "activity_mapa1.xml"

```
<?xml version="1.0" encoding="utf-8"?>
 <com.google.android.maps.MapView xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/mapa"
 android:layout_width="fill_parent"
 android:layout_height="fill_parent"
 android:clickable="true"
 android:apiKey="0qJd4FoFDCEN-v9cLbSbvWp9iOxh0GXSxOJQXqw" />
```

5.3.5 Captura de pantalla aplicación en marcha

Imagen 5.4 Inicio en el manejo de GoogleMaps

5.3.6 Conclusiones

Con esta aplicación se ha entendido el uso de API Google Maps así como la división en capas del mapa la cual permite la representación de cuantos puntos se quieran.

Además se ha experimentado con los controles del mapa; tamaño, zoom, modo de vista, centrar el mapa en una posición determinada…

5.4 Creación de varias "Activities" y paso de parámetros entre ellas. Probar el uso simultáneo de un mapa y campos de texto en una misma Activity.

En esta aplicación se han creado dos Activities, en la primera se pide al usuario introducir una serie de datos manualmente, en la segunda se extrae el identificador IMEI del terminal se muestra por pantalla junto con los datos procedentes de la otra actividad. Además en esta segunda actividad se muestra un mapa como prueba de uso de campos de texto y mapa dentro de una misma Activity.

5.4.1 Captura de pantalla de la aplicación

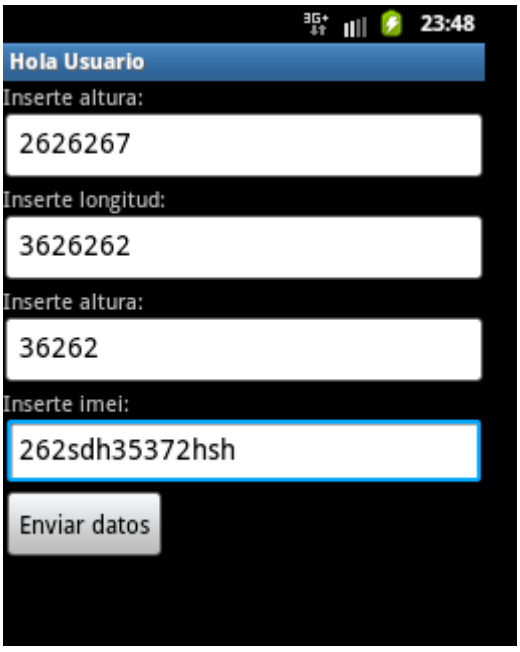


Imagen 5.5 Envío de parámetros a una activity

Imagen 5.6 Recepción de parámetros y representación de mapa

5.4.2 Código fuente de la aplicación "TaxiMap"

Activity 1

```
package net.taximap.android;

import android.app.Activity;

import android.content.Intent;
import android.os.Bundle;
```

```
import android.view.MenuInflater;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;


public class HolaUsuario extends Activity {

   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.main);

      /*Localizar los controles
      final EditText latitud = (EditText)findViewById(R.id.latitud);
      final EditText longitud = (EditText)findViewById(R.id.longitud);*/

      final EditText numero = (EditText)findViewById(R.id.numero);

      //final EditText imei = (EditText)findViewById(R.id.imei);


      final Button btnHola = (Button)findViewById(R.id.BtnHola);
      btnHola.setOnClickListener(new OnClickListener() {
                  @Override
                  public void onClick(View v) {
                     //Creamos el Intent
                     Intent intent = new Intent(HolaUsuario.this, FrmSaludo.class);

                     //Creamos la información a pasar entre actividades
                     Bundle b = new Bundle();
                     /*
                     b.putString("latitud", latitud.getText().toString());
                     b.putString("longitud", longitud.getText().toString());
                     //b.putString("altura", altura.getText().toString()); */
                     b.putString("numero", numero.getText().toString());


                     //Añadimos la información al intent
                     intent.putExtras(b);

                     //Iniciamos la nueva actividad
                     startActivity(intent);
                  }
               });
   }
}
```

Activity 2

```
package net.taximap.android;
```

```
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.widget.TextView;

//import com.android.mapa1.MyItemizedOverlay;
//import com.android.mapa1.R;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapView;

import android.content.Context;
import android.graphics.drawable.Drawable;
import android.telephony.TelephonyManager;
import android.util.Log;



import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;



import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.Toast;

public class FrmSaludo extends MapActivity implements LocationListener{


   class MyOverlay extends Overlay {
          GeoPoint point;

          public MyOverlay(GeoPoint point) {
                super();
                this.point = point;
          }

      @Override
      public boolean draw(Canvas canvas, MapView mapView, boolean shadow, long when) {
         super.draw(canvas, mapView, shadow);

         Point scrnPoint = new Point();
         mapView.getProjection().toPixels(this.point, scrnPoint);
```

```
        Bitmap marker = BitmapFactory.decodeResource(getResources(), R.drawable.icon);
        canvas.drawBitmap(marker,
                    scrnPoint.x - marker.getWidth() / 2,
                    scrnPoint.y - marker.getHeight() / 2, null);
        return true;
      }
    }


    @Override
    public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.frmsaludo);

     String textDeviceID=null;

      //Primero obtene mos el imei del telefono
      TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
      textDeviceID=getDeviceID(telephonyManager);


      //Localizar los controles
      /* TextView latitud = (TextView)findViewById(R.id.latitud);
      TextView longitud = (TextView)findViewById(R.id.longitud);
      TextView altura = (TextView)findViewById(R.id.altura); */
      TextView numero = (TextView)findViewById(R.id.numero);
      TextView imei = (TextView)findViewById(R.id.imei);
     //TextView imei=null;
     imei.setText("Identificador: " + textDeviceID);
      //Recuperamos la información pasada en el intent
      Bundle bundle = this.getIntent().getExtras();

      //Construimos el mensaje a mostrar
      /*latitud.setText("Latitud" + bundle.getString("latitud"));
      longitud.setText("Longitud" + bundle.getString("longitud"));
      altura.setText("Altura" + bundle.getString("altura")); */

      //El numero de taxi se obtiene de la activity anterior es introducido manualmente por el taxista

      numero.setText("Numero de taxi: " +bundle.getString("numero"));

      //imei.setText("Imei" + bundle.getString("imei"));
     // imei.setText("Imei: " +   textDeviceID);


      LocationManager locationManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
      updateLocation(locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER));
      locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 6000, 50, this);


      MapView mapView = (MapView) findViewById(R.id.mapa);

     //new MyLocationListener(mapView.getController(), locationManager, getBaseContext());
      mapView.setBuiltInZoomControls(true);
```

```
    Drawable marker=getResources().getDrawable(android.R.drawable.star_big_on);
    int markerWidth = marker.getIntrinsicWidth();
    int markerHeight = marker.getIntrinsicHeight();
    marker.setBounds(0, markerHeight, markerWidth, 0);


    MyItemizedOverlay myItemizedOverlay = new MyItemizedOverlay(marker);
    mapView.getOverlays().add(myItemizedOverlay);

        Location loc =
                locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

    GeoPoint myPoint1 = new GeoPoint((int) (loc.getLatitude() * 1E6), (int) (loc.getLongitude() * 1E6));
    myItemizedOverlay.addItem(myPoint1, "taxi1", "taxi1");

    /*GeoPoint myPoint2 = new GeoPoint((int) (loc.getLatitude() * 1E6-3000), (int) (loc.getLongitude() *
1E6+3000));
    myItemizedOverlay.addItem(myPoint2, "taxi2", "taxi2");

    GeoPoint myPoint3 = new GeoPoint((int) (loc.getLatitude() * 1E6+6000), (int) (loc.getLongitude() *
1E6+6000));
    myItemizedOverlay.addItem(myPoint3, "taxi3", "taxi3");

    GeoPoint myPoint4 = new GeoPoint((int) (loc.getLatitude() * 1E6+9000), (int) (loc.getLongitude() *
1E6-9000));
    myItemizedOverlay.addItem(myPoint4, "taxi4", "taxi4");

    GeoPoint myPoint5 = new GeoPoint((int) (loc.getLatitude() * 1E6-12000), (int) (loc.getLongitude() *
1E6+12000));
    myItemizedOverlay.addItem(myPoint5, "taxi5", "taxi5");*/

 // updateLocation(loc);

  }

        @Override
        protected boolean isRouteDisplayed() {
                return false;
        }

        @Override
        public void onLocationChanged(Location location) {
                updateLocation(location);
        }

        @Override
        public void onProviderDisabled(String provider) {
                Intent intent = new Intent(
android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                startActivity(intent);
        }

        @Override
        public void onStatusChanged(String provider, int status, Bundle extras) {}

        @Override
```

```
        public void onProviderEnabled(String provider) {}

        protected void updateLocation(Location location){
                MapView mapView = (MapView) findViewById(R.id.mapa);
    MapController mapController = mapView.getController();
    GeoPoint point = new GeoPoint((int) (location.getLatitude() * 1E6), (int) (location.getLongitude() *
1E6));
    mapController.animateTo(point);
    mapController.setZoom(12);
        }

  String getDeviceID(TelephonyManager phonyManager){

        String id = phonyManager.getDeviceId();
        if (id == null){
         id = "not available";
        }

        int phoneType = phonyManager.getPhoneType();
        switch(phoneType){
        case TelephonyManager.PHONE_TYPE_NONE:
         return "NONE: " + id;

        case TelephonyManager.PHONE_TYPE_GSM:
         //return "GSM: IMEI=" + id;
                return id;


        case TelephonyManager.PHONE_TYPE_CDMA:
         return "CDMA: MEID/ESN=" + id;

        /*
         *  for API Level 11 or above
         *  case TelephonyManager.PHONE_TYPE_SIP:
         *   return "SIP";
         */

        default:}
         return "UNKNOWN: ID=" + id;
        }

}
```

5.4.3 Archivo de interfaz de la aplicación “activity1.xml”

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
        xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical"
  android:layout_width="fill_parent"
  android:layout_height="wrap_content" >
```

```
        <TextView android:id="@+id/numero1"
                android:layout_height="wrap_content"
                android:layout_width="fill_parent"
           android:text="@string/numero1" />

        <EditText android:id="@+id/numero"
                android:layout_height="wrap_content"
                android:layout_width="fill_parent" />

        <Button android:id="@+id/BtnHola"
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:text="@string/envio" />

</LinearLayout>
```

5.4.4 Archivo de interfaz de la aplicación “activity2.xml”

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout
 xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
 android:layout_width="wrap_content"
 android:layout_height="wrap_content">



<TextView android:id="@+id/numero"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

<TextView android:id="@+id/imei"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />


<com.google.android.maps.MapView
   xmlns:android="http://schemas.android.com/apk/res/android"
   android:id="@+id/mapa"
   android:layout_width="wrap_content"
   android:layout_height="wrap_content"
   android:apiKey="0qJd4FoFDCEN-v9cLbSbvWp9iOxh0GXSxOJQXqw"
   android:clickable="true" >
</com.google.android.maps.MapView>


        <Button android:id="@+id/Btnenvio"
                android:layout_height="wrap_content"
                android:layout_width="wrap_content"
                android:text="@string/envio" />
```

```
</LinearLayout>
```

5.4.5 Archivo de configuración de la aplicación "AndroidManifest.xml"

```
<?xml version="1.0" encoding="utf-8"?>


<manifest
    xmlns:android="http://schemas.android.com/apk/res/android"
    package="net.sgoliver"
    android:versionCode="1"
    android:versionName="1.0">

    <uses-sdk
     android:minSdkVersion="8"
     android:targetSdkVersion="15" />
     <uses-permission android:name="android.permission.READ_PHONE_STATE"></uses-permission>


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

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

  <application android:icon="@drawable/icon"
                    android:label="@string/app_name">

    <activity android:name=".HolaUsuario"
          android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>

    <activity android:name=".FrmSaludo" />
    <uses-library android:name="com.google.android.maps" />

  </application>

</manifest>
```

5.5 Prueba de comunicación en red con servidor MYSQL y servicio web PHP remotos.

Para adquirir soltura en el manejo de servicios web y base de datos se ha decidido realizar una aplicación que se encarga de autenticar al usuario en base a los datos guardados en MYSQl y a través de un servicio web PHP

Lo primero que se necesita es una base de datos MYSQl en la cual creamos una tabla con los campo ID de usuario y contraseña. Para ello hemos contratado un alojamiento web con

la empresa 1AND1 Internet SLU. En la siguiente imagen se muestra la base de datos que se ha creado y la tabla usuarios

5.5.1 Código fuente de la aplicación "login.java"

```
package net.login.android;

import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.MenuItem;
import android.support.v4.app.NavUtils;

import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;

import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;

import android.app.Activity;
import android.content.Context;
import android.content.Intent;

import android.os.Bundle;
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnKeyListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class MainActivity extends Activity {
        EditText user;
        EditText pass;
        Button validar;
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
```

```
user = (EditText) findViewById(R.id.txtUsuario);
pass = (EditText) findViewById(R.id.txtPass);
validar = (Button) findViewById(R.id.btnValidar);

validar.setOnClickListener(new OnClickListener() {
    @Override
    public void onClick(View v) {
    ArrayList parametros = new ArrayList();
    parametros.add("Usuario");
    parametros.add(user.getText().toString());
    parametros.add("Contrasena");
    parametros.add(pass.getText().toString());



    // Llamada a Servidor Web PHP
    try {
                Post post = new Post();
                JSONArray datos = post.getServerData(parametros,
                                "http://floristasiris.com/taxi/login.php");
    // No se puede poner localhost, carga la consola de Windows
    // y escribe ipconfig/all para ver tu IP
    if (datos != null && datos.length() > 0) {
                JSONObject json_data = datos.getJSONObject(0);
                int numRegistrados = json_data.getInt("imei");
    if (numRegistrados > 0) {
                Toast.makeText(getBaseContext(),
                                "Usuario correcto. ", Toast.LENGTH_SHORT)
                                        .show();
                }
    } else {
                Toast.makeText(getBaseContext(),
                                "Usuario incorrecto. ", Toast.LENGTH_SHORT)
                                        .show();
                }
    } catch (Exception e) {
                Toast.makeText(getBaseContext(),
                                "Error al conectar con el servidor. ",
                                Toast.LENGTH_SHORT).show();
    }
    // FIN Llamada a Servidor Web PHP
                            }
                });
    }

    class Post {
                private InputStream is = null;
                private String respuesta = "";

    private void conectaPost(ArrayList parametros, String URL) {
                ArrayList nameValuePairs;
                try {

                        HttpClient httpclient = new DefaultHttpClient();

                        HttpPost httppost = new HttpPost(URL);
```

```
                                        nameValuePairs = new ArrayList();

                                        if (parametros != null) {
                                                for (int i = 0; i < parametros.size() - 1; i += 2) {
        nameValuePairs.add(new BasicNameValuePair((String)parametros.get(i),
(String)parametros.get(i + 1)));
                                                }
        httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
                                        }
                                        HttpResponse response = httpclient.execute(httppost);
                                        HttpEntity entity = response.getEntity();
                                        is = entity.getContent();
                        } catch (Exception e) {

                                        Log.e("log_tag", "Error in http connection " + e.toString());

                        } finally {

                        }
        }

        private void getRespuestaPost() {
        try {
        BufferedReader reader = new BufferedReader(
        new InputStreamReader(is, "iso-8859-1"), 8);
        StringBuilder sb = new StringBuilder();
        String line = null;
        while ((line = reader.readLine()) != null) {
                        sb.append(line + "\n");
        }
        is.close();
        respuesta = sb.toString();
        Log.e("log_tag", "Cadena JSon " + respuesta);
        } catch (Exception e) {

                        Log.e("log_tag", "Error converting result " + e.toString());

                        }
        }

        @SuppressWarnings("finally")
        private JSONArray getJsonArray() {
                        JSONArray jArray = null;
                        try {

                                        jArray = new JSONArray(respuesta);

                        } catch (Exception e) {

                        } finally {
                                        return jArray;
                        }
        }

        public JSONArray getServerData(ArrayList parametros, String URL) {
                        conectaPost(parametros, URL);
                        if (is != null) {
```

```
                    getRespuestaPost();
                }
                if (respuesta != null && respuesta.trim() != "") {
                                        return getJsonArray();
                                } else {
                                        return null;
                                }
                        }


    }


}
```

5.5.2 Archivo de configuración AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="net.login.android"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="5"
        android:targetSdkVersion="15" />

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

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
     <uses-permission android:name="android.permission.INTERNET"/>
</manifest>
```

5.5.3 Archivo de interfaz de la aplicación activity_login.xml

```
<AbsoluteLayout
android:id="@+id/widget29"
android:layout_width="fill_parent"
android:layout_height="fill_parent"
xmlns:android="http://schemas.android.com/apk/res/android"
>

<EditText
android:id="@+id/txtUsuario"
android:layout_width="150px"
android:layout_height="wrap_content"
```

```
android:textSize="18sp"
android:layout_x="80px"
android:layout_y="82px"
> </EditText>



<TextView
android:id="@+id/lblUsuario"
android:layout_width="wrap_content"
android:layout_height="21px"
android:text="User"
android:layout_x="20px"
android:layout_y="102px"
> </TextView>

<TextView
android:id="@+id/lblContrasena"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="Password"
android:layout_x="20px"
android:layout_y="152px"
> </TextView>

<EditText
android:id="@+id/txtPass"
android:layout_width="150px"
android:layout_height="wrap_content"
android:textSize="18sp"
android:layout_x="80px"
android:layout_y="142px"
> </EditText>
<Button
android:id="@+id/btnValidar"
android:layout_width="116px"
android:layout_height="44px"
android:text="Validar"
android:layout_x="80px"
android:layout_y="202px"
> </Button>

</AbsoluteLayout>
```

Código del Servicio Web “login.php”

```
<?
$user = $_POST['Usuario'];
$pass = $_POST['Contrasena'];

$link = mysql_connect('db378.1and1.es', 'dbo355128946', 'flores012');
if (!$link) {
   die('No pudo conectar: ' . mysql_error());
}

mysql_select_db("db355128946");
```

```
$q=mysql_query("SELECT * FROM usuarios WHERE USER='{$user}' AND PASSWORD='{$pass}' ");

while($e=mysql_fetch_assoc($q))

        $output[]=$e;

    print(json_encode($output));

mysql_close($link);
?>
```

5.5.4 Capturas de pantalla aplicación en marcha

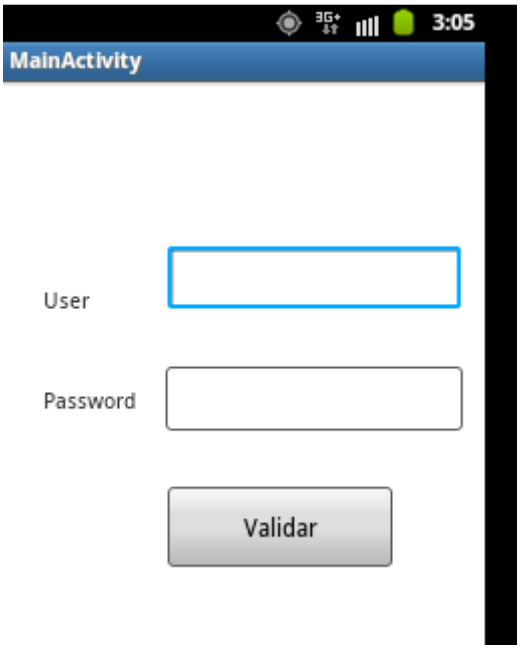


Imagen 5.7 Aplicación Login

Imagen 5.8 Login incorrecto

Imagen 5.9 Login correcto

5.6 Conexión a base de datos, lectura de datos e impresión por pantalla de los mismos.

Una vez que se puede establecer comunicación con el servidor web el siguiente y se pueden insertar datos de la aplicación en la base de datos remota, lo último que se necesita es poder leer datos de la base de datos porque esa quizá sea una de las funcionalidades más importante de todo el sistema

5.6.1 Código fuente de la aplicación "RecepciónActivity.java"

```
package com.example.mostrar_coord;


import org.apache.http.HttpResponse;

import org.apache.http.client.HttpClient;

import org.apache.http.client.methods.HttpDelete;

import org.apache.http.client.methods.HttpGet;

import org.apache.http.client.methods.HttpPost;

import org.apache.http.client.methods.HttpPut;

import org.apache.http.entity.StringEntity;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.util.EntityUtils;

import org.json.JSONArray;

import org.json.JSONObject;

import android.app.Activity;

import android.os.Bundle;

import android.util.Log;

import android.view.View;

import android.view.View.OnClickListener;

import android.widget.ArrayAdapter;

import android.widget.Button;

import android.widget.EditText;

import android.widget.ListView;
```

```
import android.widget.TextView;
public class MainActivity extends Activity {
                        private Button btnListar;
                private ListView lstClientes;

   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
      btnListar = (Button)findViewById(R.id.btnListar);
      lstClientes = (ListView)findViewById(R.id.lstClientes);
      btnListar.setOnClickListener(new OnClickListener() {
                                    @Override
                                    public void onClick(View v) {
                                                HttpClient httpClient = new DefaultHttpClient();
                              HttpGet del = new HttpGet("http://floristasiris.com/taxi/mostrar.php");
                                                del.setHeader("content-type", "application/json");
                                                try
                                    {
                                    HttpResponse resp = httpClient.execute(del);
                                    String respStr = EntityUtils.toString(resp.getEntity());
                                    JSONArray respJSON = new JSONArray(respStr);
                                    String[] clientes = new String[respJSON.length()];
                                    for(int i=0; i<respJSON.length(); i++)
                                    {
                                                JSONObject obj = respJSON.getJSONObject(i);
                                                String imei = obj.getString("imei");
                                                String latitud = obj.getString("latitud");
                                                String longitud = obj.getString("longitud");
```

```
                    clientes[i] = "" + imei + "-" + latitud + "-" + longitud;
                }
                    //Rellenamos la lista con los resultados
                ArrayAdapter<String> adaptador =
                    new ArrayAdapter<String>(MainActivity.this,
                        android.R.layout.simple_list_item_1, clientes);
                            lstClientes.setAdapter(adaptador);
            }
            catch(Exception ex)
            {
                Log.e("ServicioRest","Error!", ex);
            }
                }
        });
    }
}
```

5.6.2 Archivo de configuración AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.mostrar_coord"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />
 <uses-permission android:name="android.permission.INTERNET"/>
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
```

```
    </activity>
  </application>
```

5.6.3 Archivo de interfaz de la aplicación “activity_recibir.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:orientation="vertical" >

    <Button
      android:id="@+id/btnListar"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Listar" />

  <ListView
    android:id="@+id/lstClientes"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
  </ListView>

</LinearLayout>
```

Código del Servicio Web mostrar.php

Este servicio es el encargado de recibir los datos de la aplicación android(IMEI, Latitud y Longitud) e insertarlos en la base de datos

```
<?php
mysql_connect("db378.1and1.es", "dbo355128946","flores012");
mysql_select_db("db355128946");

$pregunta = "SELECT * " .
    "FROM taxi" ;

$sql=mysql_query($pregunta);
while($row=mysql_fetch_assoc($sql))
$output[]=$row; print(json_encode($output)); mysql_close();
?>
```

5.7 Obtención de IMEI y coordenadas y envío y almacenamiento en servidor MYSQL

Una vez que se puede establecer comunicación con el servidor web el siguiente paso es insertar datos de la aplicación en la base de datos remota, en concreto, estos datos serán las coordenadas y el identificador IMEI.

5.7.1 Código fuente de la aplicación "EnvioActivity.java"

```
package net.envio.android;

import android.app.Activity;
import android.os.Bundle;
import android.content.Context;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.telephony.TelephonyManager;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import org.apache.http.client.HttpClient;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.HttpResponse;
import org.apache.http.HttpEntity;
import org.apache.http.NameValuePair;

import java.io.InputStream;
import java.util.ArrayList;

public class MainActivity extends Activity {
private Button btnActualizar;
private Button btnDesactivar;
private TextView lblLatitud;
private TextView lblLongitud;
private TextView lblPrecision;
private TextView lblEstado;

private LocationManager locManager;
private LocationListener locListener;

private Location defaultLoc;

   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_main);
```

```
        btnActualizar = (Button)findViewById(R.id.BtnActualizar);
        btnDesactivar = (Button)findViewById(R.id.BtnDesactivar);
        lblLatitud = (TextView)findViewById(R.id.LblPosLatitud);
        lblLongitud = (TextView)findViewById(R.id.LblPosLongitud);
        lblPrecision = (TextView)findViewById(R.id.LblPosPrecision);
        lblEstado = (TextView)findViewById(R.id.LblEstado);

        btnActualizar.setOnClickListener(new OnClickListener() {
//@Override
public void onClick(View v) {
comenzarLocalizacion();
}
});

        btnDesactivar.setOnClickListener(new OnClickListener() {
//@Override
public void onClick(View v) {
locManager.removeUpdates(locListener);
}
});
    }

    public void postData(Location loc){
        String textDeviceID=null;

        //Primero obtene mos el imei del telefono
        TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
        textDeviceID=getDeviceID(telephonyManager);


     ArrayList<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(2);

     nameValuePairs.add(new BasicNameValuePair("imei",String.valueOf(textDeviceID)));
        nameValuePairs.add(new BasicNameValuePair("latitud",String.valueOf(loc.getLatitude())));
        nameValuePairs.add(new BasicNameValuePair("longitud",String.valueOf(loc.getLongitude())));

    ;
        //http post
        try{
            HttpClient httpclient = new DefaultHttpClient();
            HttpPost httppost = new HttpPost("http://www.floristasiris.com/taxi/inserta.php");
            httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
            HttpResponse response = httpclient.execute(httppost);
            HttpEntity entity = response.getEntity();
            InputStream is = entity.getContent();
            Log.i("Connection made", response.getStatusLine().toString());
        }

        catch(Exception e)
        {
            Log.e("log_tag", "Error in http connection "+e.toString());
        }
    }

    private void comenzarLocalizacion()
```

```
   {
    //Obtenemos una referencia al LocationManager
    locManager = (LocationManager)getSystemService(Context.LOCATION_SERVICE);

    //Obtenemos la última posición conocida
    Location loc = locManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

    defaultLoc = loc;

    //Mostramos la última posición conocida
    mostrarPosicion(loc);

    //Nos registramos para recibir actualizaciones de la posición
    locListener = new LocationListener() {
public void onLocationChanged(Location location) {
mostrarPosicion(location);
}
public void onProviderDisabled(String provider){
lblEstado.setText("Provider OFF");
}
public void onProviderEnabled(String provider){
lblEstado.setText("Provider ON ");
}
public void onStatusChanged(String provider, int status, Bundle extras){
Log.i("", "Provider Status: " + status);
lblEstado.setText("Provider Status: " + status);
}
    };

    locManager.requestLocationUpdates(
    LocationManager.NETWORK_PROVIDER, 1000, 0, locListener);

   }

   private void mostrarPosicion(Location loc) {
    if(loc != null)
    {
    lblLatitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
    lblLongitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));
    lblPrecision.setText("Precision: " + String.valueOf(loc.getAccuracy()));
    Log.i("", String.valueOf(loc.getLatitude() + " - " + String.valueOf(loc.getLongitude())));
    postData(loc);
    }
    else
    {
    lblLatitud.setText("Latitud: (sin_datos)");
    lblLongitud.setText("Longitud: (sin_datos)");
    lblPrecision.setText("Precision: (sin_datos)");
    }
   }
   String getDeviceID(TelephonyManager phonyManager){

            String id = phonyManager.getDeviceId();
            if (id == null){
             id = "not available";
            }
```

```
            int phoneType = phonyManager.getPhoneType();
            switch(phoneType){
            case TelephonyManager.PHONE_TYPE_NONE:
             return "NONE: " + id;

            case TelephonyManager.PHONE_TYPE_GSM:
             //return "GSM: IMEI=" + id;
                        return id;

            case TelephonyManager.PHONE_TYPE_CDMA:
             return "CDMA: MEID/ESN=" + id;

            /*
             *  for API Level 11 or above
             *  case TelephonyManager.PHONE_TYPE_SIP:
             *   return "SIP";
             */

            default:}
             return "UNKNOWN: ID=" + id;
            }
}
```

5.7.2 Archivo de configuración AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
   package="net.envio.android"
   android:versionCode="1"
   android:versionName="1.0" >

   <uses-sdk
      android:minSdkVersion="5"
      android:targetSdkVersion="15" />
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
<uses-permission android:name="android.permission.INTERNET" />

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

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

            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
      </activity>
   </application>

</manifest>
```

5.7.3 Archivo de interfaz de la aplicación “activity_envio.xml”

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent" >

   <TextView android:id="@+id/LblPosicion"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_margin="5dip"
             android:text="Posición Actual:" />

          <TextView android:id="@+id/LblPosLatitud"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_margin="5dip"
             android:background="#aaaaaa"
             android:textColor="#000000" />

          <TextView android:id="@+id/LblPosLongitud"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_margin="5dip"
             android:background="#aaaaaa"
             android:textColor="#000000" />

          <TextView android:id="@+id/LblPosPrecision"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_margin="5dip"
             android:background="#aaaaaa"
             android:textColor="#000000" />

          <Button android:id="@+id/BtnActualizar"
                  android:layout_width="fill_parent"
                  android:layout_height="wrap_content"
                  android:text="Enviar" />

          <Button android:id="@+id/BtnDesactivar"
                  android:layout_width="fill_parent"
                  android:layout_height="wrap_content"
                  android:text="Desactivar" />

          <TextView android:id="@+id/LblEst"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_margin="5dip"
             android:text="Estado proveedor:" />

          <TextView android:id="@+id/LblEstado"
             android:layout_width="fill_parent"
             android:layout_height="wrap_content"
             android:layout_margin="5dip"
             android:background="#aaaaaa"
             android:textColor="#000000" />
```

```
</LinearLayout>
```

Código del Servicio Web inserta.php

Este servicio es el encargado de recibir los datos de la aplicación android(IMEI, Latitud y Longitud) e insertarlos en la base de datos

```
<?

$imei = $_POST['imei'];
$latitud = $_POST['latitud'];
$longitud = $_POST['longitud'];

echo $imei;
echo $latitud;
echo $longitud;

$link = mysql_connect('db378.1and1.es', 'dbo355128946', 'flores012');

if (!$link) {
   die('No pudo conectar: ' . mysql_error());
}

mysql_select_db("db355128946");

$q=mysql_query("INSERT INTO taxi (imei, latitud, longitud) VALUES ($imei, $latitud, $longitud)");

mysql_close($link);
?>
```

Código del Servicio Web ver.php

Este sencillo script nos permite comprobar que efectivamente los datos se han guardado en la base de datos

```
<table border="1" width="700px" cellspacing=1 cellpadding=2 style="font-size: 8pt"><tr>
<td><font face="verdana"><b>Imei</b></font></td>
<td><font face="verdana"><b>Latitud</b></font></td>
<td><font face="verdana"><b>Longitud</b></font></td>

</tr>
<?php
  $link = @mysql_connect("db378.1and1.es", "dbo355128946","flores012")
     or die ("Error al conectar a la base de datos.");
  @mysql_select_db("db355128946", $link)
     or die ("Error al conectar a la base de datos.");

  $query = "SELECT * " .
     "FROM taxi" ;

  $result = mysql_query($query);
```

```
 while($row = mysql_fetch_array($result))
 {
 echo "<tr><td width=\"30%\"><font face=\"verdana\">" .
            $row["imei"] . "</font></td>";
  echo "<td width=\"30%\"><font face=\"verdana\">" .
            $row["latitud"] . "</font></td>";
  echo "<td width=\"30%\"><font face=\"verdana\">" .
            $row["longitud"] . "</font></td></tr>";
 }

 mysql_free_result($result);
 mysql_close($link);
?>
</table>
```

5.7.4 Capturas aplicación en marcha

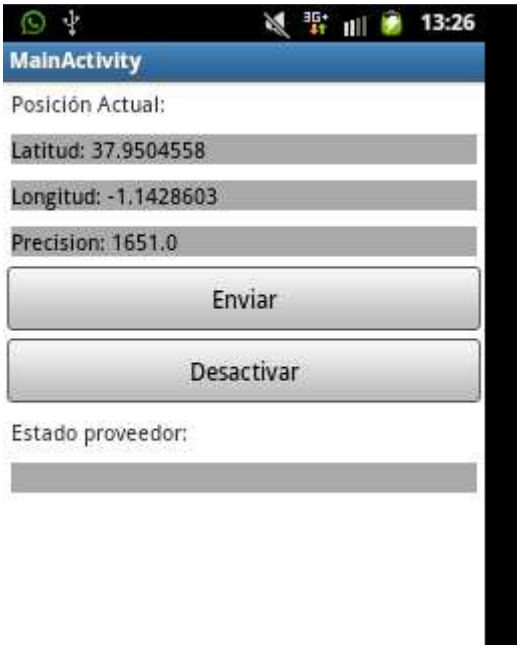
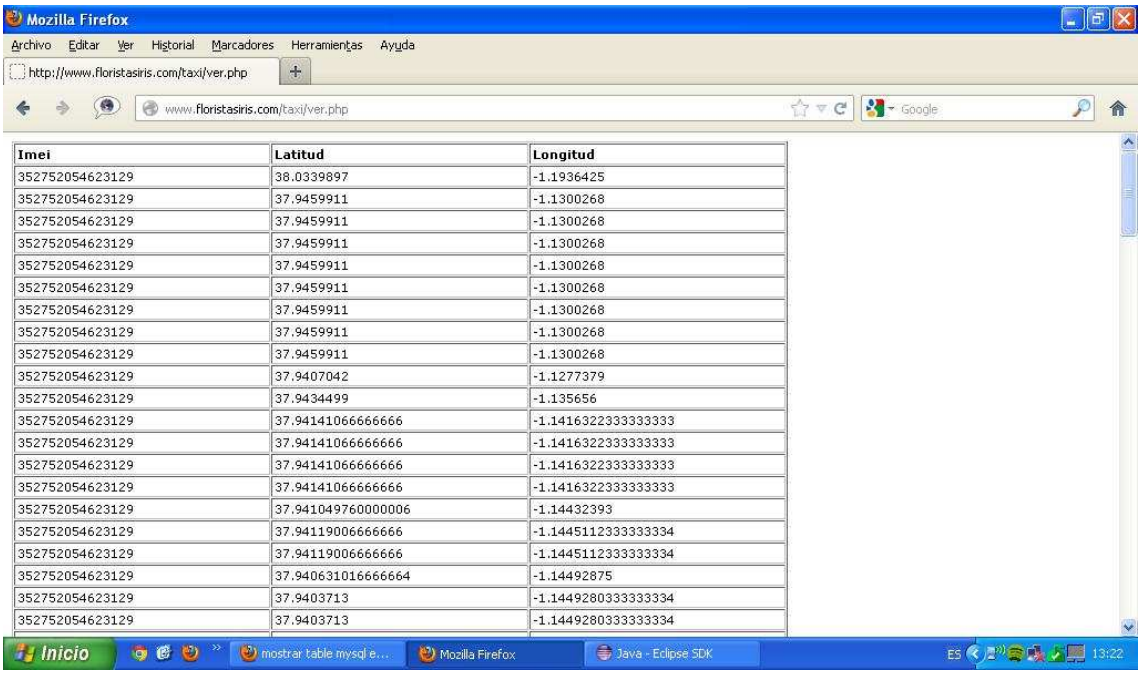


Imagen 5.10 Enviar Datos



Imei	Latitud	Longitud
352752054623129	38.0339897	-1.1936425
352752054623129	37.9459911	-1.1300268
352752054623129	37.9459911	-1.1300268
352752054623129	37.9459911	-1.1300268
352752054623129	37.9459911	-1.1300268
352752054623129	37.9459911	-1.1300268
352752054623129	37.9459911	-1.1300268
352752054623129	37.9459911	-1.1300268
352752054623129	37.9459911	-1.1300268
352752054623129	37.9407042	-1.1277379
352752054623129	37.9434499	-1.135656
352752054623129	37.94141066666666	-1.1416322333333333
352752054623129	37.94141066666666	-1.1416322333333333
352752054623129	37.94141066666666	-1.1416322333333333
352752054623129	37.94141066666666	-1.1416322333333333
352752054623129	37.941049760000006	-1.14432393
352752054623129	37.94119006666666	-1.1445112333333334
352752054623129	37.94119006666666	-1.1445112333333334
352752054623129	37.940631016666664	-1.14492875
352752054623129	37.9403713	-1.1449280333333334
352752054623129	37.9403713	-1.1449280333333334

Imagen 5.11 Verificar Datos

6. Diseño del sistema

Una vez que se han explicado los conceptos básicos de Android, las herramientas utilizadas y sobre todo, se han adquirido los conocimientos necesarios para la funcionalidad del sistema se procede ahora a explicar cómo se ha diseñando el sistema.

La figura 6.1 muestra la arquitectura general del sistema:

Figura 6.1. Arquitectura general del sistema

6.1 Aplicación taxista

La aplicación taxista tiene las siguientes funcionalidades:

- Obtener coordenadas GPS e identificador del terminal IMEI
- Mostrar en un mapa la posición actual
- Enviar datos al servicio web y quedar a la espera de peticiones
- Enviar confirmación de petición a cliente
- Detener el envío de datos a servidor hasta que el taxi vuelva a quedar libre

A continuación se muestran las distintas capturas de la aplicación en funcionamiento y se explica lo que se hace en cada pantalla así como los diferentes elementos de la interfaz.

La primera pantalla que se muestra invita al usuario a introducir sus datos. Se ha decidido hacerlo de esta manera para evitar que los taxistas ilegales puedan utilizar esta aplicación.

Imagen 6.1 Pantalla principal de TaxiMap Taxista

Para proceder a su registro en el sistema, el taxista pulsará el botón registro que le llevará a la pantalla mostrada en la Imagen 6.2. Además de los datos que se piden la aplicación recoge en este punto el identificador IMEI del usuario que junto con los anteriores son enviados al servicio web.

Cuando el servicio web recibe los datos los almacena en la base de datos y envía al usuario un mail en el que se le informa de cuál es su nombre de usuario y contraseña.

Imagen 6.2 Pantalla de registro de taxistas en el sistema

Imagen 6.3 Mensaje de confirmación de registro

Una vez que el usuario se ha registrado en el sistema puede ingresar su nombre de usuario y contraseña en la pantalla principal lo cual le llevará a la pantalla mostrada en la figura 6.4, en la cual se muestra al usuario sus coordenadas GPS, identificador IMEI y un mapa con su posición. Por último hay un botón para activar localización, que es enviar sus coordenadas al sistema y que este esté operativo para clientes cercanos.

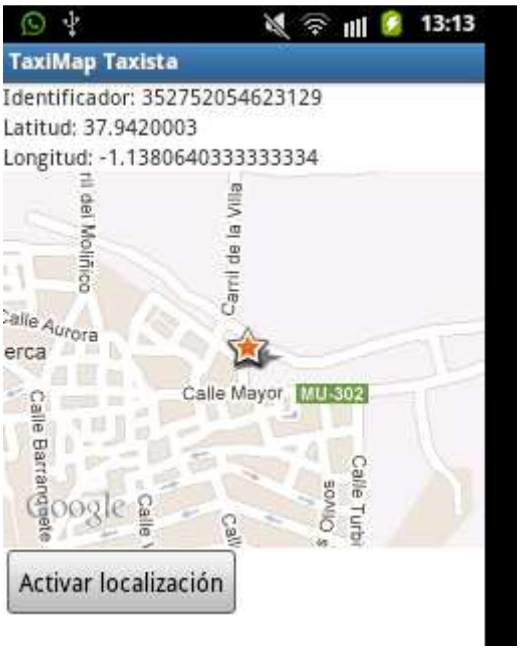


Imagen 6.4 Localización del taxi

Cuando se pulsa el botón Activar Localización la aplicación llama a un servicio que se encarga de enviar las coordenadas al servidor cada 10 segundos. La aplicación queda a la vez, a la espera de posibles clientes tal como se muestra en la imagen 6.6

Imagen 6.5 Envío de datos al servidor

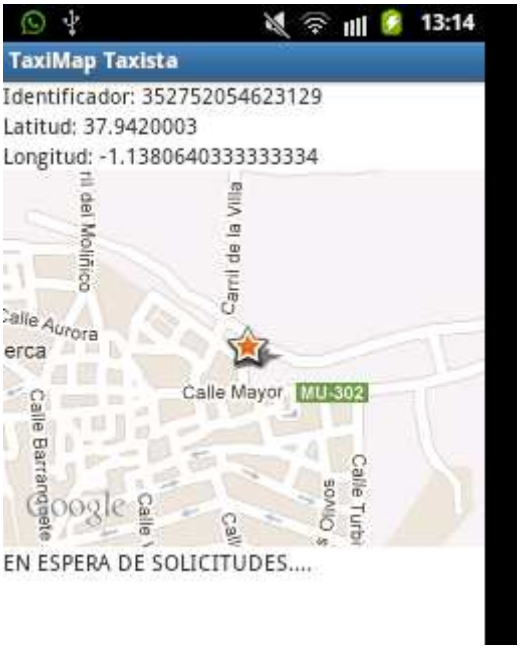


Imagen 6.6 Pantalla de espera solicitudes

Cuando un cliente solicita ese taxi se muestra un aviso al taxista tal y como puede verse en la figura 6.7

Imagen 6.7 Solicitud recibida

Cuando se pulsa el botón Ver cliente la aplicación muestra al taxista su situación y la del cliente tal y como puede observarse en la siguiente imagen:

Imagen 6.8 Pantalla de localización de cliente y taxista

Cuando el taxista pulsa sobre *Confirmar cliente* se envía un mensaje de aviso al cliente para que no se mueva del sitio y se desactiva el servicio de localización del taxista, sus coordenadas dejan de enviarse al sistema pues se supone que ya está ocupado. La aplicación vuelve a la pantalla de localización(Imagen 6.4):

Imagen 6.9 Aviso de confirmación de recogida de cliente

Cuando se pulsa el botón rechazar cliente, se notifica al cliente que ha sido rechazado invitándole a buscar de nuevo y la aplicación vuelve al servicio de localización en espera de nuevos clientes.

6.2 Aplicación cliente

La aplicación cliente tiene las siguientes funcionalidades:

- Obtener coordenadas GPS e identificador del terminal IMEI
- Mostrar en un mapa la posición actual
- Enviar datos al servicio web y quedar a la espera de datos
- Recibir datos(coordenadas e identificadores) del servicio web y mostrar en el mapa
- Realizar petición a taxi más cercano
- Quedar a la espera de respuesta.

A continuación se muestran las distintas capturas de la aplicación en funcionamiento y se explica lo que se hace en cada pantalla así como los diferentes elementos de la interfaz.

Como ya se comentó en al apartado de pruebas la localización por GPS presenta un inconveniente en interiores pues esta no puede ser realizada mediante los satélites así que para esta aplicación el primer paso es dar elegir al usuario la forma de localización óptima en función de donde se encuentre.

Imagen 6.10 Pantalla principal aplicación cliente

Una vez que el cliente selecciona la forma de localización óptima pasamos a la siguiente pantalla en la cual se muestran las coordenadas e IMEI del usuario, su posición en un mapa y por último hay un botón mediante el cual la aplicación inicia el servicio web que se encarga de buscar los taxis registrados libres más cercanos.

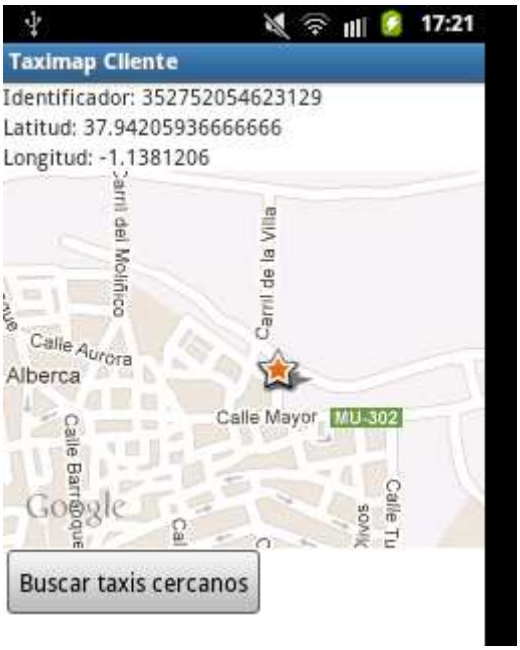


Imagen 6.11 Pantalla de localización de cliente

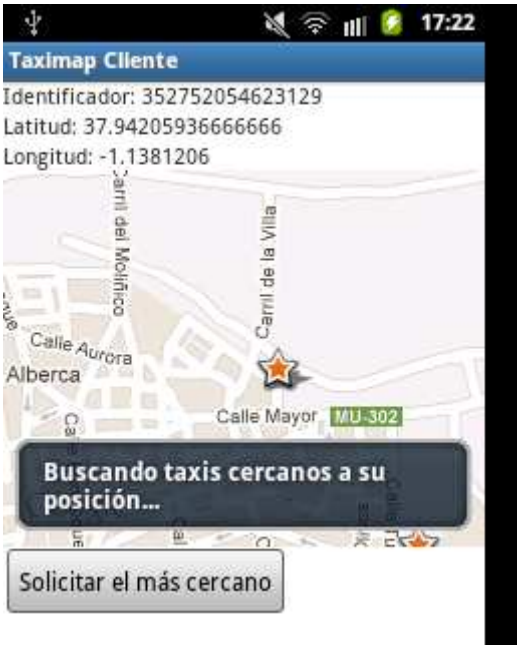


Imagen 6.12 Servicio de localización de Taxis cercanos

Imagen 6.13 Mapa de taxis cercanos

Una vez que se tienen los taxis libres encontrados más próximos, la aplicación permite contactar con el que más cerca esté a través del botón Solicitar el más cercano.

Imagen 6.14 Servicio de solicitud de taxi más cercano

Cuando se solicita el taxi más cercano la aplicación muestra el mensaje de estado de espera como se puede ver en la imagen 6.14 y pasa a la pantalla de la imagen 6.15 en la que se muestra sobre un mapa la localización del cliente y la del taxi solicitado.

Además, en la parte de abajo se muestra un mensaje de espera de confirmación.

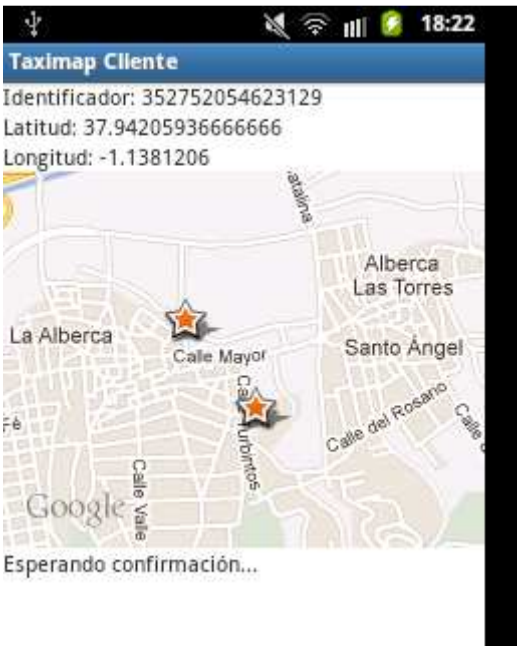


Imagen 6.15 Pantalla de espera de respuesta

Una vez que el taxista responde la petición esta respuesta es transmitida al cliente. Cuando la respuesta es afirmativa, lo cual quiere decir que el taxista acepta la petición se informa a través de una notificación y la aplicación se cierra.

Imagen 6.16 Mensaje de confirmación de petición

Imagen 6.17 Mensaje de cierre de la aplicación

Si por el contrario, el taxi rechaza la petición entonces se informa al usuario mediante un mensaje como el mostrado en la imagen 6.18 en el que se incluye un botón para realizar una nueva búsqueda

Imagen 6.18 Mensaje de petición rechazada

6.3 Centro de control

El centro de control nos es más que una página web desde la que se pueden visualizar los clientes que están usando el sistema, gestionar los taxistas registrados y obtener cierta información de las carreras realizadas filtrada por diversos campos(fecha, IMEI, nombre de taxista…)

En la siguiente imagen se muestra la pagina principal del panel de control, en ella hay 3 zonas principales; la zona de gestión de clientes, la zona de gestión de taxistas y la zona de gestión de carreras.

Imagen 6.19 Centro de control

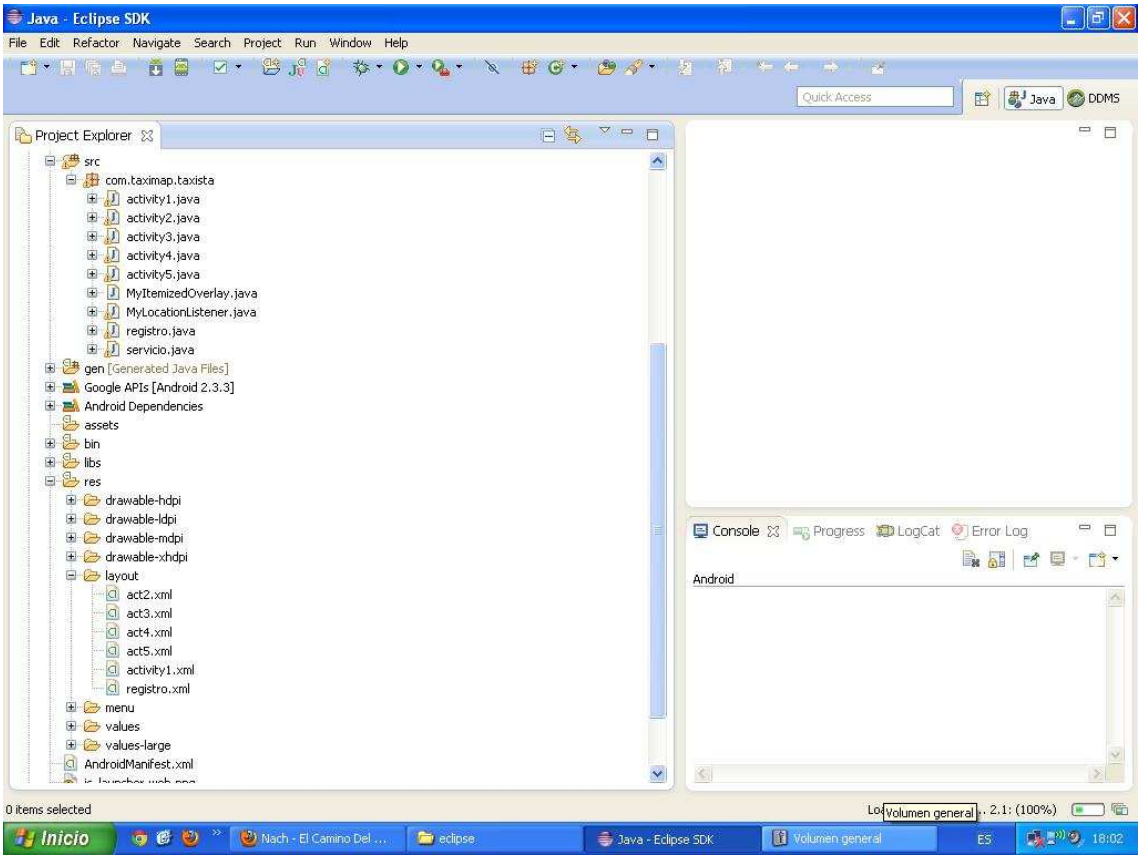
Imagen 6.20 Centro de control-Taxistas

Imagen 6.21 Centro de control-Carreras

Imagen 6.22 Centro de control-Clientes

7. Implementación del sistema

7.1 Aplicación Taxista



7.1 Explorador de proyectos de Eclipse, Aplicación taxista

Si se observa la imagen 7.1 puede verse una captura de pantalla de eclipse en la que tenemos a la izquierda la aplicación totalmente desplegada como puede verse está formada por nueve clases de las cuales seis son utilizadas como activities o pantallas de la aplicación, hay otras 2 que se usan como objetos en otras y una que es un servicio, 1 servicio y

Al igual que cualquier aplicación android tiene en archivo de interfaz XML para cada pantalla y el archivo de configuración de la aplicación “AndroidManifest.xml”. A continuación se muestra el código de cada una de estas clases y se explica lo que hace:

Activity1.Java

Esta clase es la pantalla principal, en ella se muestra el mensaje de bienvenida y en orden descendente; dos campos de texto para que el usuario se identifique una vez

que se ha registrado, un botón para enviar los datos y un último botón abajo para registrarse en caso de ser la primera vez que se usa la aplicación.

Lo que la hace la activity1 es recoger los datos del usuario y enviarlos al servicio web a través del estándar JSON. EL servicio será el encargado de cogerlos comparar con los valores de la base de datos y devolver el resultado. Además esta activity abre la activity *"registro"* cuando se pulsa sobre el botón inferior.

```
package com.taximap.taxista;

import android.os.Bundle;

import android.app.Activity;

import android.view.Menu;

import android.view.MenuItem;

import android.support.v4.app.NavUtils;

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.util.ArrayList;

import org.apache.http.HttpEntity;

import org.apache.http.HttpResponse;

import org.apache.http.NameValuePair;

import org.apache.http.client.HttpClient;

import org.apache.http.client.entity.UrlEncodedFormEntity;

import org.apache.http.client.methods.HttpPost;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.message.BasicNameValuePair;

import org.json.JSONArray;

import org.json.JSONObject;

import android.app.Activity;

import android.content.Context;

import android.content.Intent;

import android.os.Bundle;
```

```
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnKeyListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
public class activity1 extends Activity {
        EditText user;
        EditText pass;
        Button validar;
        Button registro;
   @Override
   public void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity1);
      user = (EditText) findViewById(R.id.txtUsuario);
      pass = (EditText) findViewById(R.id.txtPass);
      validar = (Button) findViewById(R.id.btnValidar);
      registro = (Button) findViewById(R.id.btnRegistro);

      registro.setOnClickListener(new OnClickListener() {
          @Override
          public void onClick(View v) {
                Intent i= new Intent(activity1.this, registro.class);
               startActivity(i);
```

```
        }
});

validar.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
        ArrayList parametros = new ArrayList();
        parametros.add("Usuario");
        parametros.add(user.getText().toString());
        parametros.add("Contrasena");
        parametros.add(pass.getText().toString());

        // Llamada a Servidor Web PHP
        try {
                        Post post = new Post();
                        JSONArray datos = post.getServerData(parametros,
                                                "http://floristasiris.com/taxi/login.php");
        // No se puede poner localhost, carga la consola de Windows
        // y escribe ipconfig/all para ver tu IP
        if (datos != null && datos.length() > 0) {
                        JSONObject json_data = datos.getJSONObject(0);
                        int numRegistrados = json_data.getInt("ID_USUARIO");
        if (numRegistrados > 0) {
          Toast.makeText(getBaseContext(),
          "Usuario correcto. ", Toast.LENGTH_SHORT).show();
          //El usuario está registrado, ahora nos vamos a la segunda activity
          Intent i= new Intent(activity1.this, activity2.class);
          startActivity(i);
```

```
                }
} else {
                Toast.makeText(getBaseContext(),
                                "Usuario incorrecto. ", Toast.LENGTH_SHORT)
                                .show();
                }
} catch (Exception e) {
                Toast.makeText(getBaseContext(),
                                "Error al conectar con el servidor. ",
                                Toast.LENGTH_SHORT).show();
}
// FIN Llamada a Servidor Web PHP
                        }
                });
}

class Post {
                private InputStream is = null;
                private String respuesta = "";

private void conectaPost(ArrayList parametros, String URL) {
                ArrayList nameValuePairs;
                try {
                        HttpClient httpclient = new DefaultHttpClient();
                        HttpPost httppost = new HttpPost(URL);
                        nameValuePairs = new ArrayList();

                        if (parametros != null) {
```

```
                                        for (int i = 0; i < parametros.size() - 1; i += 2) {
        nameValuePairs.add(new BasicNameValuePair((String)parametros.get(i),
(String)parametros.get(i + 1)));
                                        }
        httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
                                }
                                HttpResponse response = httpclient.execute(httppost);
                                HttpEntity entity = response.getEntity();
                                is = entity.getContent();
                        } catch (Exception e) {
                                Log.e("log_tag", "Error in http connection " + e.toString());
                        } finally {
                        }
        }
        private void getRespuestaPost() {
        try {
        BufferedReader reader = new BufferedReader(
        new InputStreamReader(is, "iso-8859-1"), 8);
        StringBuilder sb = new StringBuilder();
        String line = null;
        while ((line = reader.readLine()) != null) {
                        sb.append(line + "\n");
        }
        is.close();
        respuesta = sb.toString();
        Log.e("log_tag", "Cadena JSon " + respuesta);
        } catch (Exception e) {
                        Log.e("log_tag", "Error converting result " + e.toString());
                        }
        }
```

```
        @SuppressWarnings("finally")
        private JSONArray getJsonArray() {
                JSONArray jArray = null;
                try {
                        jArray = new JSONArray(respuesta);
                } catch (Exception e) {
                } finally {
                        return jArray;
                }
        }
        public JSONArray getServerData(ArrayList parametros, String URL) {
                conectaPost(parametros, URL);
                if (is != null) {
                getRespuestaPost();
                }
                if (respuesta != null && respuesta.trim() != "") {
                return getJsonArray();
                    } else {
                        return null;
                            }
                           }
   }
}
```

Registro.Java

La activity es la encargada del registro de usuarior, tienes varios campos de texto en los que se recogen los datos del usuario y además de forma totalmente invisible al ususario toma el numero IMEI del terminal que será el que lo identifique posteriormente. Posteriormente los datos son enviados al servicio web para que los

inserte en la base de datos y para que comunique al usuario a través de un correo electrónico sus datos Login y Password.

Código Fuente Registro.java

```
package com.taximap.taxista;

import android.os.Bundle;

import android.app.Activity;

import android.view.Menu;

import android.view.MenuItem;

import android.support.v4.app.NavUtils;

import android.telephony.TelephonyManager;

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.util.ArrayList;

import org.apache.http.HttpEntity;

import org.apache.http.HttpResponse;

import org.apache.http.NameValuePair;

import org.apache.http.client.HttpClient;

import org.apache.http.client.entity.UrlEncodedFormEntity;

import org.apache.http.client.methods.HttpPost;

import org.apache.http.entity.StringEntity;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.message.BasicNameValuePair;

import org.apache.http.util.EntityUtils;

import org.json.JSONArray;

import org.json.JSONObject;

import android.app.Activity;

import android.content.Context;
```

```
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnKeyListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class registro extends Activity {
        private Button registrado;
        private EditText nombre;
        private EditText apellido1;
        private EditText apellido2;
        private EditText mail;

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.registro);
    nombre = (EditText)findViewById(R.id.nombre);
    apellido1 = (EditText)findViewById(R.id.apellido1);
    apellido2 = (EditText)findViewById(R.id.apellido2);
    mail = (EditText)findViewById(R.id.mail);
    registrado = (Button) findViewById(R.id.registro);
```

```
String textDeviceID=null;
TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
textDeviceID=getDeviceID(telephonyManager);

registrado.setOnClickListener(new OnClickListener() {
    public void onClick(View v){

        /*nombre.setText(String.valueOf(nombre));
        apellido1.setText(String.valueOf(apellido1));
        apellido2.setText(String.valueOf(apellido2));
        mail.setText(String.valueOf(mail));*/

            ArrayList<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(2);

        nameValuePairs.add(new BasicNameValuePair("nombre",String.valueOf(nombre)));
        nameValuePairs.add(new BasicNameValuePair("apellido1",String.valueOf(apellido1)));
        nameValuePairs.add(new BasicNameValuePair("apellido2",String.valueOf(apellido2)));
        nameValuePairs.add(new BasicNameValuePair("mail",String.valueOf(mail)));
        //http post
        try{
            HttpClient httpclient = new DefaultHttpClient();
            HttpPost httppost = new
HttpPost("http://www.floristasiris.com/taxi/inserta_registro.php");
            httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
            HttpResponse response = httpclient.execute(httppost);
            HttpEntity entity = response.getEntity();
            InputStream is = entity.getContent();
            Log.i("Connection made", response.getStatusLine().toString());
        }
```

```
catch(Exception e)
{
   Log.e("log_tag", "Error in http connection "+e.toString());
}
        /*HttpClient httpClient = new DefaultHttpClient();
            HttpPost post = new
HttpPost("http://floristasiris.com/taxi/inserta_registro.php");
            post.setHeader("content-type", "application/json");
            try
        {
                //Construimos el objeto cliente en formato JSON
                JSONObject dato = new JSONObject();

                //dato.put("Id", Integer.parseInt(txtId.getText().toString()));

                //dato.put("telefono",
Integer.parseInt(txtTelefono.getText().toString()));
                dato.put("nombre", nombre.getText().toString());
                dato.put("apellido1", apellido1.getText().toString());
                dato.put("apellido2", apellido2.getText().toString());
                dato.put("mail", mail.getText().toString());
                //dato.put("imei", textDeviceID);

                StringEntity entity = new StringEntity(dato.toString());
                post.setEntity(entity);
        HttpResponse resp = httpClient.execute(post);
        String respStr = EntityUtils.toString(resp.getEntity());
        if(respStr.equals("true"))
            Toast.makeText(getBaseContext(),
           "Datos enviados, revise su e-mail", Toast.LENGTH_SHORT).show();
```

```
                    }
                catch(Exception ex)
                {
                    Log.e("ServicioRest","Error!", ex);
                }*/
            Toast.makeText(getBaseContext(),
            "Datos enviados, revise su e-mail", Toast.LENGTH_SHORT).show();
            Intent i= new Intent(registro.this, activity1.class);
            startActivity(i);
        }//click view
    });//onclicklistener*/
}//create
    String getDeviceID(TelephonyManager phonyManager){

     String id = phonyManager.getDeviceId();
     if (id == null){
      id = "not available";
     }
     int phoneType = phonyManager.getPhoneType();
     switch(phoneType){
     case TelephonyManager.PHONE_TYPE_NONE:
      return "NONE: " + id;
     case TelephonyManager.PHONE_TYPE_GSM:
      //return "GSM: IMEI=" + id;
            return id;
     case TelephonyManager.PHONE_TYPE_CDMA:
      return "CDMA: MEID/ESN=" + id;
     /*
      *  for API Level 11 or above
```

```
         *  case TelephonyManager.PHONE_TYPE_SIP:
         *   return "SIP";
         */
        default:}
         return "UNKNOWN: ID=" + id;
        }
}
```

Activity2.Java

La activity se encarga de recoger los datos GPS del usuario y mostrar la ubicación actual en un mapa, además se imprimen las coordenadas y el IMEI como campos de texto, finalmente cuanta con un botón que es el que llama a la activity que contiene el servicio de envío de coordenadas al servidor.

```
package com.taximap.taxista;
import android.location.Location;
import android.location.LocationManager;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.MenuItem;
import android.support.v4.app.NavUtils;
import android.telephony.TelephonyManager;
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.HttpClient;
```

```
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnKeyListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.maps.GeoPoint;
```

```
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapView;
import android.content.Context;
import android.graphics.drawable.Drawable;
import android.telephony.TelephonyManager;
import android.util.Log;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.Toast;



public class activity2 extends MapActivity implements LocationListener{
        private LocationManager locationManager;
                private Button send;
  class MyOverlay extends Overlay {
```

```
        GeoPoint point;
        public MyOverlay(GeoPoint point) {
                super();
                this.point = point;
        }

    @Override
    public boolean draw(Canvas canvas, MapView mapView, boolean shadow, long when) {
      super.draw(canvas, mapView, shadow);

      Point scrnPoint = new Point();
      mapView.getProjection().toPixels(this.point, scrnPoint);
      Bitmap marker = BitmapFactory.decodeResource(getResources(), R.drawable.icon);
      canvas.drawBitmap(marker,
                scrnPoint.x - marker.getWidth() / 2,
                scrnPoint.y - marker.getHeight() / 2, null);
      return true;
    }
  }

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.act2);
    String textDeviceID=null;
    //Primero obtene mos el imei del telefono
    TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
    textDeviceID=getDeviceID(telephonyManager);
   final TextView imei = (TextView)findViewById(R.id.imei);
```

```
    //TextView imei=null;
    imei.setText("Identificador: " + textDeviceID);
     LocationManager locationManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
    //updateLocation(locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER));
    //locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 6000, 50, this);
        Location loc =
                locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
         //int im=Integer.parseInt(textDeviceID);
       // double lat = loc.getLatitude()*1E6;
        //double lon = loc.getLongitude()*1E6;
    final TextView latitud = (TextView)findViewById(R.id.latitud);
    final TextView longitud = (TextView)findViewById(R.id.longitud);
    latitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
    longitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));
    MapView mapView = (MapView) findViewById(R.id.mapa);
   //new MyLocationListener(mapView.getController(), locationManager, getBaseContext());
    mapView.setBuiltInZoomControls(true);
    Drawable marker=getResources().getDrawable(android.R.drawable.star_big_on);
    int markerWidth = marker.getIntrinsicWidth();
    int markerHeight = marker.getIntrinsicHeight();
    marker.setBounds(0, markerHeight, markerWidth, 0);
    MyItemizedOverlay myItemizedOverlay = new MyItemizedOverlay(marker);
    mapView.getOverlays().add(myItemizedOverlay);
        Location loc1 =
                locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
    GeoPoint myPoint1 = new GeoPoint((int) (loc1.getLatitude() * 1E6), (int) (loc1.getLongitude() *
1E6));
    myItemizedOverlay.addItem(myPoint1, "taxi1", "taxi1");
```

```
    /*GeoPoint myPoint2 = new GeoPoint((int) (loc.getLatitude() * 1E6-3000), (int) (loc.getLongitude() *
1E6+3000));

    myItemizedOverlay.addItem(myPoint2, "taxi2", "taxi2");

    GeoPoint myPoint3 = new GeoPoint((int) (loc.getLatitude() * 1E6+6000), (int) (loc.getLongitude() *
1E6+6000));

    myItemizedOverlay.addItem(myPoint3, "taxi3", "taxi3");

    GeoPoint myPoint4 = new GeoPoint((int) (loc.getLatitude() * 1E6+9000), (int) (loc.getLongitude() *
1E6-9000));

    myItemizedOverlay.addItem(myPoint4, "taxi4", "taxi4");

    GeoPoint myPoint5 = new GeoPoint((int) (loc.getLatitude() * 1E6-12000), (int) (loc.getLongitude() *
1E6+12000));

    myItemizedOverlay.addItem(myPoint5, "taxi5", "taxi5");*/

 // updateLocation(loc);


    send = (Button) findViewById(R.id.activar);

    send.setOnClickListener(new OnClickListener() {

        public void onClick(View v) {

                Intent i= new Intent(activity2.this, activity4.class);

               startActivity(i);

        }

    });


  }//cierre oncreate()

        String getDeviceID(TelephonyManager phonyManager){

         String id = phonyManager.getDeviceId();

         if (id == null){

          id = "not available";

         }

         int phoneType = phonyManager.getPhoneType();

         switch(phoneType){

         case TelephonyManager.PHONE_TYPE_NONE:
```

```
return "NONE: " + id;

case TelephonyManager.PHONE_TYPE_GSM:
//return "GSM: IMEI=" + id;
        return id;

case TelephonyManager.PHONE_TYPE_CDMA:
return "CDMA: MEID/ESN=" + id;
/*
 *  for API Level 11 or above
 *  case TelephonyManager.PHONE_TYPE_SIP:
 *   return "SIP";
 */

default:}
return "UNKNOWN: ID=" + id;
}
@Override
        protected boolean isRouteDisplayed() {
                return false;
        }
        @Override
        public void onLocationChanged(Location location) {
                updateLocation(location);
        }

        @Override
        public void onProviderDisabled(String provider) {
                Intent intent = new Intent(
android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);
```

```
                startActivity(intent);
            }
            @Override
            public void onStatusChanged(String provider, int status, Bundle extras) {}
            @Override
            public void onProviderEnabled(String provider) {}

            protected void updateLocation(Location location){
                MapView mapView = (MapView) findViewById(R.id.mapa);
        MapController mapController = mapView.getController();
        GeoPoint point = new GeoPoint((int) (location.getLatitude() * 1E6), (int)
(location.getLongitude() * 1E6));
        mapController.animateTo(point);
        mapController.setZoom(30);
            }
}//cierre class
```

Activity 4.Java

Esta activity se encarga de inciar el servicio de localización enviando las coordenadas al servicio web. Luego queda a la espera de notificaciones, o lo que es lo mismo de que un cliente pida un taxi

```
package com.taximap.taxista;
import android.location.Location;
import android.location.LocationManager;
import android.os.Bundle;
import android.app.Activity;
import android.app.AlertDialog;
import android.view.Menu;
import android.view.MenuItem;
import android.support.v4.app.NavUtils;
```

```
import android.telephony.TelephonyManager;
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import org.apache.http.HttpEntity;
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.app.Activity;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
```

```
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnKeyListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapView;

import android.content.Context;
import android.graphics.drawable.Drawable;
import android.telephony.TelephonyManager;
import android.util.Log;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.location.Address;
import android.location.Geocoder;
```

```
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.Toast;



public class activity4 extends MapActivity implements LocationListener{
        private LocationManager locationManager;
   class MyOverlay extends Overlay {
        GeoPoint point;

        public MyOverlay(GeoPoint point) {
                super();
                this.point = point;
        }

    @Override
    public boolean draw(Canvas canvas, MapView mapView, boolean shadow, long when) {
      super.draw(canvas, mapView, shadow);

      Point scrnPoint = new Point();
      mapView.getProjection().toPixels(this.point, scrnPoint);

      Bitmap marker = BitmapFactory.decodeResource(getResources(), R.drawable.icon);
      canvas.drawBitmap(marker,
                scrnPoint.x - marker.getWidth() / 2,
                scrnPoint.y - marker.getHeight() / 2, null);
      return true;
```

```
    }
  }

  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.act4);
    String textDeviceID=null;
    //Toast.makeText(getBaseContext(),
      //    "Enviando datos al servidor... ", Toast.LENGTH_SHORT).show();

    //Primero obtene mos el imei del telefono
    TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
    textDeviceID=getDeviceID(telephonyManager);
    final TextView imei = (TextView)findViewById(R.id.imei);
    //TextView imei=null;
    imei.setText("Identificador: " + textDeviceID);
    LocationManager locationManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
    //updateLocation(locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER));
    //locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 6000, 50, this);
        Location loc =
                locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
        //int im=Integer.parseInt(textDeviceID);
     // double lat = loc.getLatitude()*1E6;
        //double lon = loc.getLongitude()*1E6;

    final TextView latitud = (TextView)findViewById(R.id.latitud);
    final TextView longitud = (TextView)findViewById(R.id.longitud);
```

```
latitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
longitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));

MapView mapView = (MapView) findViewById(R.id.mapa);

//new MyLocationListener(mapView.getController(), locationManager, getBaseContext());
mapView.setBuiltInZoomControls(true);
Drawable marker=getResources().getDrawable(android.R.drawable.star_big_on);
int markerWidth = marker.getIntrinsicWidth();
int markerHeight = marker.getIntrinsicHeight();
marker.setBounds(0, markerHeight, markerWidth, 0);
MyItemizedOverlay myItemizedOverlay = new MyItemizedOverlay(marker);
mapView.getOverlays().add(myItemizedOverlay);

    Location loc1 =
            locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

GeoPoint myPoint1 = new GeoPoint((int) (loc1.getLatitude() * 1E6), (int) (loc1.getLongitude() *
1E6));
myItemizedOverlay.addItem(myPoint1, "taxi1", "taxi1");

Intent svc = new Intent(activity4.this, servicio.class);
startService(svc);
    AlertDialog alertDialog = new AlertDialog.Builder(
     activity4.this).create();
// Setting Dialog Title
    alertDialog.setTitle("Solicitud recibida");
    // Setting Dialog Message
    //alertDialog.setMessage("Elija una opción");
```

```
        // Setting Icon to Dialog

        // Setting OK Button

        alertDialog.setButton("Ver cliente", new DialogInterface.OnClickListener() {

            public void onClick(DialogInterface dialog, int which) {

            // Write your code here to execute after dialog closed


                Toast.makeText(getApplicationContext(), "Cargando mapa..",
Toast.LENGTH_SHORT).show();

                    Intent i= new Intent(activity4.this, activity5.class);

                    startActivity(i);

                }

        });


        // Showing Alert Message

        alertDialog.show();

 // updateLocation(loc);

    /* ArrayList<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(2);


     nameValuePairs.add(new BasicNameValuePair("imei",String.valueOf(textDeviceID)));

       nameValuePairs.add(new BasicNameValuePair("latitud",String.valueOf(loc.getLatitude())));

       nameValuePairs.add(new BasicNameValuePair("longitud",String.valueOf(loc.getLongitude())));

            //http post

       try{

          HttpClient httpclient = new DefaultHttpClient();

          HttpPost httppost = new HttpPost("http://www.floristasiris.com/taxi/inserta_coord_taxi.php");

          httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));

          HttpResponse response = httpclient.execute(httppost);

          HttpEntity entity = response.getEntity();

          InputStream is = entity.getContent();

          Log.i("Connection made", response.getStatusLine().toString());
```

```
    }
    catch(Exception e)
    {
       Log.e("log_tag", "Error in http connection "+e.toString());
    }
  */
}//cierre oncreate()
        String getDeviceID(TelephonyManager phonyManager){

         String id = phonyManager.getDeviceId();
         if (id == null){
          id = "not available";
         }
         int phoneType = phonyManager.getPhoneType();
         switch(phoneType){
         case TelephonyManager.PHONE_TYPE_NONE:
          return "NONE: " + id;
         case TelephonyManager.PHONE_TYPE_GSM:
          //return "GSM: IMEI=" + id;
                    return id;
         case TelephonyManager.PHONE_TYPE_CDMA:
          return "CDMA: MEID/ESN=" + id;

         /*
          *  for API Level 11 or above
          *  case TelephonyManager.PHONE_TYPE_SIP:
          *   return "SIP";
          */
```

```
        default:}
         return "UNKNOWN: ID=" + id;
        }
        @Override
                protected boolean isRouteDisplayed() {
                        return false;
                }
                @Override
                public void onLocationChanged(Location location) {
                        updateLocation(location);
                }
                @Override
                public void onProviderDisabled(String provider) {
                        Intent intent = new Intent(
android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                        startActivity(intent);
                }
                @Override
                public void onStatusChanged(String provider, int status, Bundle extras) {}

                @Override
                public void onProviderEnabled(String provider) {}
                protected void updateLocation(Location location){
                        MapView mapView = (MapView) findViewById(R.id.mapa);
            MapController mapController = mapView.getController();
            GeoPoint point = new GeoPoint((int) (location.getLatitude() * 1E6), (int)
(location.getLongitude() * 1E6));
            mapController.animateTo(point);
            mapController.setZoom(30);
                }
```

}//cierre class

Activity 5.Java

Es la última pantalla que muestra la aplicación, en ella se muestra en un mapa la localización del taxista yt del cliente que lo solicita y muestra un mensaje de confirmación para el taxista para que cofirme si va o no a por el cliente

```
package com.taximap.taxista;

import android.location.Location;

import android.location.LocationManager;

import android.os.Bundle;

import android.app.Activity;

import android.app.AlertDialog;

import android.view.Menu;

import android.view.MenuItem;

import android.support.v4.app.NavUtils;

import android.telephony.TelephonyManager;

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.util.ArrayList;

import org.apache.http.HttpEntity;

import org.apache.http.HttpResponse;

import org.apache.http.NameValuePair;

import org.apache.http.client.HttpClient;

import org.apache.http.client.entity.UrlEncodedFormEntity;

import org.apache.http.client.methods.HttpPost;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.message.BasicNameValuePair;

import org.json.JSONArray;
```

```
import org.json.JSONObject;

import com.google.android.maps.GeoPoint;

import com.google.android.maps.MapController;

import com.google.android.maps.MapView;

import com.google.android.maps.Overlay;

import android.app.Activity;

import android.content.Context;

import android.content.DialogInterface;

import android.content.Intent;

import android.graphics.Bitmap;

import android.graphics.BitmapFactory;

import android.graphics.Canvas;

import android.graphics.Point;

import android.graphics.drawable.Drawable;

import android.os.Bundle;

import android.util.Log;

import android.view.KeyEvent;

import android.view.View;

import android.view.View.OnClickListener;

import android.view.View.OnKeyListener;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

import com.google.android.maps.GeoPoint;

import com.google.android.maps.MapActivity;

import com.google.android.maps.MapView;

import android.content.Context;

import android.graphics.drawable.Drawable;
```

```
import android.telephony.TelephonyManager;
import android.util.Log;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.Toast;

public class activity5 extends MapActivity implements LocationListener{
        private Button confirmar;
        private Button rechazar;
        private LocationManager locationManager;
   class MyOverlay extends Overlay {
        GeoPoint point;

        public MyOverlay(GeoPoint point) {
                super();
```

```
                this.point = point;
            }
        @Override
        public boolean draw(Canvas canvas, MapView mapView, boolean shadow, long when) {
            super.draw(canvas, mapView, shadow);
            Point scrnPoint = new Point();
            mapView.getProjection().toPixels(this.point, scrnPoint);
            Bitmap marker = BitmapFactory.decodeResource(getResources(), R.drawable.icon);
            canvas.drawBitmap(marker,
                    scrnPoint.x - marker.getWidth() / 2,
                    scrnPoint.y - marker.getHeight() / 2, null);
            return true;
        }
    }

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.act5);
        String textDeviceID=null;
        //Primero obtene mos el imei del telefono
        TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
        textDeviceID=getDeviceID(telephonyManager);
       final TextView imei = (TextView)findViewById(R.id.imei);
       //TextView imei=null;
       imei.setText("Identificador: " + textDeviceID);
        LocationManager locationManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
        //updateLocation(locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER));
```

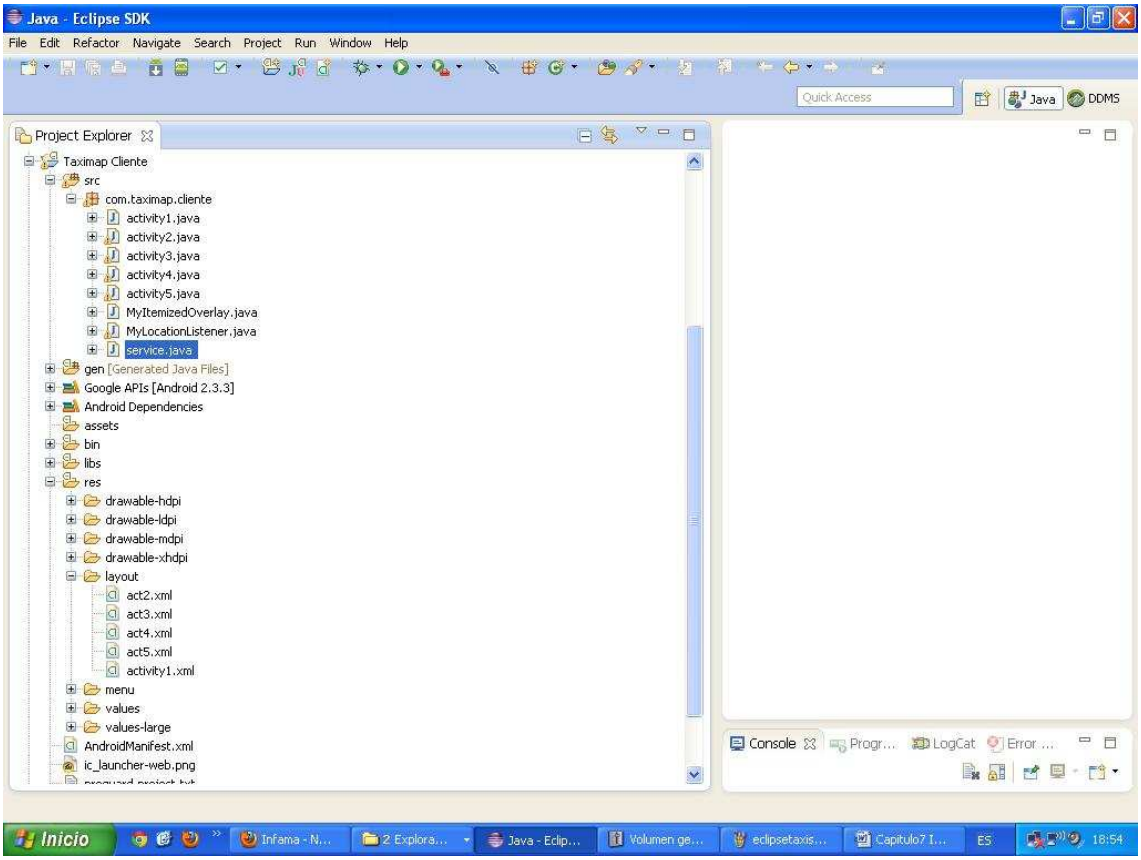
```
//locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 6000, 50, this);
        Location loc =
                locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
final TextView latitud = (TextView)findViewById(R.id.latitud);
final TextView longitud = (TextView)findViewById(R.id.longitud);
latitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
longitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));
MapView mapView = (MapView) findViewById(R.id.mapa);
//new MyLocationListener(mapView.getController(), locationManager, getBaseContext());
mapView.setBuiltInZoomControls(true);
Drawable marker=getResources().getDrawable(android.R.drawable.star_big_on);
int markerWidth = marker.getIntrinsicWidth();
int markerHeight = marker.getIntrinsicHeight();
marker.setBounds(0, markerHeight, markerWidth, 0);
MyItemizedOverlay myItemizedOverlay = new MyItemizedOverlay(marker);
mapView.getOverlays().add(myItemizedOverlay);
        Location loc1 =
                locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

GeoPoint myPoint1 = new GeoPoint((int) (loc1.getLatitude() * 1E6), (int) (loc1.getLongitude() *
1E6));
myItemizedOverlay.addItem(myPoint1, "taxi1", "taxi1");

GeoPoint myPoint2 = new GeoPoint((int) (loc.getLatitude() * 1E6), (int) (loc.getLongitude() *
1E6+1000));
myItemizedOverlay.addItem(myPoint2, "taxi2", "taxi2");
confirmar = (Button) findViewById(R.id.confirmar);
confirmar.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
                Toast.makeText(getBaseContext(),
```

```
                "Cliente a la espera de ser recogido", Toast.LENGTH_SHORT).show();
                Intent i= new Intent(activity5.this, activity2.class);
                startActivity(i);
        }
    });

    rechazar = (Button) findViewById(R.id.rechazar);
    rechazar.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
                Toast.makeText(getBaseContext(),
                "Cliente rechazado", Toast.LENGTH_SHORT).show();
                Intent i= new Intent(activity5.this, activity4.class);
                startActivity(i);
        }
    });

}//cierre oncreate()
        String getDeviceID(TelephonyManager phonyManager){
         String id = phonyManager.getDeviceId();
         if (id == null){
          id = "not available";
         }
         int phoneType = phonyManager.getPhoneType();
         switch(phoneType){
         case TelephonyManager.PHONE_TYPE_NONE:
          return "NONE: " + id;
         case TelephonyManager.PHONE_TYPE_GSM:
          //return "GSM: IMEI=" + id;
                  return id;
```

```
case TelephonyManager.PHONE_TYPE_CDMA:
 return "CDMA: MEID/ESN=" + id;
/*
 *  for API Level 11 or above
 *  case TelephonyManager.PHONE_TYPE_SIP:
 *   return "SIP";
 */
default:}
 return "UNKNOWN: ID=" + id;
}
@Override
        protected boolean isRouteDisplayed() {
                return false;
        }

        @Override
        public void onLocationChanged(Location location) {
                updateLocation(location);
        }

        @Override
        public void onProviderDisabled(String provider) {
                Intent intent = new Intent(
android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                startActivity(intent);
        }
        @Override
        public void onStatusChanged(String provider, int status, Bundle extras) {}
        @Override
        public void onProviderEnabled(String provider) {}
```

```
        protected void updateLocation(Location location){

                MapView mapView = (MapView) findViewById(R.id.mapa);

        MapController mapController = mapView.getController();

        GeoPoint point = new GeoPoint((int) (location.getLatitude() * 1E6), (int)
(location.getLongitude() * 1E6));

        mapController.animateTo(point);

        mapController.setZoom(30);

            }


}//cierre class
```

7.2 Aplicación Cliente



7.1 Explorador de proyectos de Eclipse, Aplicación cliente

Si se observa la imagen 7.2 puede verse una captura de pantalla de eclipse en la que tenemos a la izquierda la aplicación totalmente desplegada como puede verse está formada por nueve clases de las cuales seis son utilizadas como Activities o pantallas de la aplicación, hay otras 2 que se usan como objetos en otras y una que es un servicio, 1 servicio. Al igual que cualquier aplicación Android tiene en archivo de interfaz XML para cada pantalla y el archivo de configuración de la aplicación "AndroidManifest.xml". A continuación se muestra el código de cada una de estas clases y se explica lo que hace:

Activity1.java

Esta es la pantalla de bienvenida al usuario, en este caso el cliente, en ella aparecen dos botones para que el usuario seleccione el modo de obtención de su posición.

```
package com.taximap.cliente;


import android.content.Intent;
import android.os.Bundle;
import android.app.Activity;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
public class activity1 extends Activity {
        private Button datos;
        private Button satelite;
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity1);
```

```
        datos = (Button) findViewById(R.id.button1);
        datos.setOnClickListener(new OnClickListener() {
            public void onClick(View v) {
                    Intent i= new Intent(activity1.this, activity2.class);
                    startActivity(i);
            }
        });
        satelite = (Button) findViewById(R.id.button2);
        satelite.setOnClickListener(new OnClickListener() {
            public void onClick(View v) {
                    Intent i= new Intent(activity1.this, activity3.class);
                    startActivity(i);
            }
        });
    }
}
```

Activity 2 y 3

Estas hacen básicamente lo mismo solo que una obtiene la localización mediante el GPS y la otra mediante por conexión de datos. Obtienen la localización y ubican al usuario en el mapa. Por último un botón invita al usuario a buscar taxis cercanos

package com.taximap.cliente;

```
import android.os.Bundle;
import android.telephony.TelephonyManager;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
```

```
//import com.android.mapa1.MyItemizedOverlay;
//import com.android.mapa1.R;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapView;
import android.content.Context;
import android.graphics.drawable.Drawable;
import android.telephony.TelephonyManager;
import android.util.Log;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.Toast;
public class activity2 extends MapActivity implements LocationListener{
        private Button enviar;
```

```
        private LocationManager locationManager;
  class MyOverlay extends Overlay {
        GeoPoint point;
        public MyOverlay(GeoPoint point) {
              super();
              this.point = point;
        }

    @Override
    public boolean draw(Canvas canvas, MapView mapView, boolean shadow, long when) {
      super.draw(canvas, mapView, shadow);
      Point scrnPoint = new Point();
      mapView.getProjection().toPixels(this.point, scrnPoint);
      Bitmap marker = BitmapFactory.decodeResource(getResources(), R.drawable.icon);
      canvas.drawBitmap(marker,
              scrnPoint.x - marker.getWidth() / 2,
              scrnPoint.y - marker.getHeight() / 2, null);
      return true;
    }
  }
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.act2);
    String textDeviceID=null;
    //Primero obtene mos el imei del telefono
    TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
```

```
        textDeviceID=getDeviceID(telephonyManager);
       final TextView imei = (TextView)findViewById(R.id.imei);
       //TextView imei=null;
       imei.setText("Identificador: " + textDeviceID);


        LocationManager locationManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
       //updateLocation(locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER));
       //locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 6000, 50, this);
            Location loc =
                    locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
            //int im=Integer.parseInt(textDeviceID);
        // double lat = loc.getLatitude()*1E6;
            //double lon = loc.getLongitude()*1E6;
       final TextView latitud = (TextView)findViewById(R.id.latitud);
       final TextView longitud = (TextView)findViewById(R.id.longitud);
       latitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
       longitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));
       MapView mapView = (MapView) findViewById(R.id.mapa);
           //new MyLocationListener(mapView.getController(), locationManager, getBaseContext());
       mapView.setBuiltInZoomControls(true);
       Drawable marker=getResources().getDrawable(android.R.drawable.star_big_on);
       int markerWidth = marker.getIntrinsicWidth();
       int markerHeight = marker.getIntrinsicHeight();
       marker.setBounds(0, markerHeight, markerWidth, 0);
       MyItemizedOverlay myItemizedOverlay = new MyItemizedOverlay(marker);
       mapView.getOverlays().add(myItemizedOverlay);
            Location loc1 =
                    locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
```

```
    GeoPoint myPoint1 = new GeoPoint((int) (loc1.getLatitude() * 1E6), (int) (loc1.getLongitude() *
1E6));
    myItemizedOverlay.addItem(myPoint1, "taxi1", "taxi1");
    myItemizedOverlay.addItem(myPoint4, "taxi4", "taxi4");
    enviar = (Button) findViewById(R.id.enviar);
    enviar.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {

                Intent i= new Intent(activity2.this, activity4.class);
                startActivity(i);
        }
    });

  }//cierre oncreate()

  @Override
        protected boolean isRouteDisplayed() {
                return false;
        }

        @Override
        public void onLocationChanged(Location location) {
                updateLocation(location);
        }

        @Override
        public void onProviderDisabled(String provider) {
                Intent intent = new Intent(
android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                startActivity(intent);
```

```
        }
        @Override
        public void onStatusChanged(String provider, int status, Bundle extras) {}
        @Override
        public void onProviderEnabled(String provider) {}
        protected void updateLocation(Location location){
                MapView mapView = (MapView) findViewById(R.id.mapa);
    MapController mapController = mapView.getController();
    GeoPoint point = new GeoPoint((int) (location.getLatitude() * 1E6), (int) (location.getLongitude() *
1E6));
    mapController.animateTo(point);
    mapController.setZoom(12);
        }
        String getDeviceID(TelephonyManager phonyManager){

         String id = phonyManager.getDeviceId();
         if (id == null){
          id = "not available";
         }
         int phoneType = phonyManager.getPhoneType();
         switch(phoneType){
         case TelephonyManager.PHONE_TYPE_NONE:
          return "NONE: " + id;
         case TelephonyManager.PHONE_TYPE_GSM:
          //return "GSM: IMEI=" + id;
                  return id;

         case TelephonyManager.PHONE_TYPE_CDMA:
          return "CDMA: MEID/ESN=" + id;
```

```
            /*
             *  for API Level 11 or above
             *  case TelephonyManager.PHONE_TYPE_SIP:
             *   return "SIP";
             */
            default:}
             return "UNKNOWN: ID=" + id;
            }
}//cierre class
```

Activity 4.Java

Esta activity se encarga de inciar el servicio de localización enviando las coordenadas al servicio web. Luego queda a la de los datos de taxis cercanos , una vez que los recibe los representa en el mapa, de manera que se muestran la posición del cliente y la de los taxistas cercano. Por último hay un botón donde lo que se hace es solicitar los servicios del que se sitúa a una distancia menor.

```
package com.taximap.cliente;
import android.location.Location;
import android.location.LocationManager;
import android.os.Bundle;
import android.app.Activity;
import android.view.Menu;
import android.view.MenuItem;
import android.support.v4.app.NavUtils;
import android.telephony.TelephonyManager;
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.util.ArrayList;
import org.apache.http.HttpEntity;
```

```
import org.apache.http.HttpResponse;
import org.apache.http.NameValuePair;
import org.apache.http.client.HttpClient;
import org.apache.http.client.entity.UrlEncodedFormEntity;
import org.apache.http.client.methods.HttpPost;
import org.apache.http.impl.client.DefaultHttpClient;
import org.apache.http.message.BasicNameValuePair;
import org.json.JSONArray;
import org.json.JSONObject;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.app.Activity;
import android.content.Context;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnKeyListener;
import android.widget.Button;
import android.widget.EditText;
```

```
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapView;
import android.content.Context;
import android.graphics.drawable.Drawable;
import android.telephony.TelephonyManager;
import android.util.Log;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.Toast;
public class activity4 extends MapActivity implements LocationListener{
        private Button contactar;
        private LocationManager locationManager;
```

```
class MyOverlay extends Overlay {
        GeoPoint point;

        public MyOverlay(GeoPoint point) {
                super();
                this.point = point;
        }

   @Override
   public boolean draw(Canvas canvas, MapView mapView, boolean shadow, long when) {
      super.draw(canvas, mapView, shadow);

      Point scrnPoint = new Point();
      mapView.getProjection().toPixels(this.point, scrnPoint);

      Bitmap marker = BitmapFactory.decodeResource(getResources(), R.drawable.icon);
      canvas.drawBitmap(marker,
                scrnPoint.x - marker.getWidth() / 2,
                scrnPoint.y - marker.getHeight() / 2, null);
      return true;
   }
}
@Override
public void onCreate(Bundle savedInstanceState) {
   super.onCreate(savedInstanceState);
   setContentView(R.layout.act4);
   String textDeviceID=null;
```

```
        Toast.makeText(getBaseContext(),
                "Buscando taxis cercanos a su posición... ", Toast.LENGTH_SHORT).show();
        //Primero obtene mos el imei del telefono
        TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
        textDeviceID=getDeviceID(telephonyManager);
       final TextView imei = (TextView)findViewById(R.id.imei);
       //TextView imei=null;
       imei.setText("Identificador: " + textDeviceID);
        LocationManager locationManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
        //updateLocation(locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER));
        //locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 6000, 50, this);
             Location loc =
                        locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
              //int im=Integer.parseInt(textDeviceID);
         // double lat = loc.getLatitude()*1E6;
             //double lon = loc.getLongitude()*1E6;
        final TextView latitud = (TextView)findViewById(R.id.latitud);
        final TextView longitud = (TextView)findViewById(R.id.longitud);
        latitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
        longitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));
        MapView mapView = (MapView) findViewById(R.id.mapa);
       //new MyLocationListener(mapView.getController(), locationManager, getBaseContext());
        mapView.setBuiltInZoomControls(true);
        Drawable marker=getResources().getDrawable(android.R.drawable.star_big_on);
        int markerWidth = marker.getIntrinsicWidth();
        int markerHeight = marker.getIntrinsicHeight();
        marker.setBounds(0, markerHeight, markerWidth, 0);
```

```
MyItemizedOverlay myItemizedOverlay = new MyItemizedOverlay(marker);

mapView.getOverlays().add(myItemizedOverlay);

    Location loc1 =

            locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);

GeoPoint myPoint1 = new GeoPoint((int) (loc1.getLatitude() * 1E6), (int) (loc1.getLongitude() *
1E6));

myItemizedOverlay.addItem(myPoint1, "taxi1", "taxi1");

GeoPoint myPoint2 = new GeoPoint((int) (loc.getLatitude() * 1E6-3000), (int) (loc.getLongitude() *
1E6+3000));

myItemizedOverlay.addItem(myPoint2, "taxi2", "taxi2");


GeoPoint myPoint3 = new GeoPoint((int) (loc.getLatitude() * 1E6+3000), (int) (loc.getLongitude() *
1E6-3000));

myItemizedOverlay.addItem(myPoint3, "taxi3", "taxi3");


GeoPoint myPoint4 = new GeoPoint((int) (loc.getLatitude() * 1E6+3000), (int) (loc.getLongitude() *
1E6+3000));

myItemizedOverlay.addItem(myPoint4, "taxi4", "taxi4");

GeoPoint myPoint5 = new GeoPoint((int) (loc.getLatitude() * 1E6-12000), (int) (loc.getLongitude() *
1E6-12000));

myItemizedOverlay.addItem(myPoint5, "taxi5", "taxi5");

 // updateLocation(loc);

ArrayList<NameValuePair> nameValuePairs = new ArrayList<NameValuePair>(2);

nameValuePairs.add(new BasicNameValuePair("imei",String.valueOf(textDeviceID)));

  nameValuePairs.add(new BasicNameValuePair("latitud",String.valueOf(loc.getLatitude())));

  nameValuePairs.add(new BasicNameValuePair("longitud",String.valueOf(loc.getLongitude())));

  //http post

  try{

    HttpClient httpclient = new DefaultHttpClient();

    HttpPost httppost = new
HttpPost("http://www.floristasiris.com/taxi/inserta_coord_cliente.php");
```

```
    httppost.setEntity(new UrlEncodedFormEntity(nameValuePairs));
    HttpResponse response = httpclient.execute(httppost);
    HttpEntity entity = response.getEntity();
    InputStream is = entity.getContent();
    Log.i("Connection made", response.getStatusLine().toString());
            }

    catch(Exception e)
    {
    Log.e("log_tag", "Error in http connection "+e.toString());
    }

    contactar = (Button) findViewById(R.id.button1);
    contactar.setOnClickListener(new OnClickListener() {
        public void onClick(View v) {
                                Toast.makeText(getBaseContext(),
                "Enviando petición, espere respuesta... ", Toast.LENGTH_SHORT).show();
                //Intent i= new Intent(activity3.this, activity2.class);
                //startActivity(i);
        }
    });

    Intent i= new Intent(activity4.this, activity5.class);
                startActivity(i);

}//cierre oncreate()
```

```
String getDeviceID(TelephonyManager phonyManager){

String id = phonyManager.getDeviceId();
if (id == null){
 id = "not available";
}

int phoneType = phonyManager.getPhoneType();
switch(phoneType){
case TelephonyManager.PHONE_TYPE_NONE:
 return "NONE: " + id;

case TelephonyManager.PHONE_TYPE_GSM:
 //return "GSM: IMEI=" + id;
            return id;
case TelephonyManager.PHONE_TYPE_CDMA:
 return "CDMA: MEID/ESN=" + id;

/*
 *  for API Level 11 or above
 *  case TelephonyManager.PHONE_TYPE_SIP:
 *   return "SIP";
 */
default:}
 return "UNKNOWN: ID=" + id;
}
@Override
          protected boolean isRouteDisplayed() {
                    return false;
```

```
                }

                @Override

                public void onLocationChanged(Location location) {

                        updateLocation(location);

                }

                @Override

                public void onProviderDisabled(String provider) {

                        Intent intent = new Intent(
android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);

                        startActivity(intent);

                }

                @Override

                public void onStatusChanged(String provider, int status, Bundle extras) {}

                @Override

                public void onProviderEnabled(String provider) {}


                protected void updateLocation(Location location){

                        MapView mapView = (MapView) findViewById(R.id.mapa);

            MapController mapController = mapView.getController();

            GeoPoint point = new GeoPoint((int) (location.getLatitude() * 1E6), (int)
(location.getLongitude() * 1E6));

            mapController.animateTo(point);

            mapController.setZoom(30);

                }


}//cierre class
```

Activity 5.Java

Es la última pantalla que muestra la aplicación, en ella se muestra en un mapa la localización del cliente y del taxista más cercano y muestra un mensaje con la confirmación del taxista o bien con le informa de que la petición fué rechazada por el taxista

```
package com.taximap.cliente;


import android.location.Location;

import android.location.LocationManager;

import android.os.Bundle;

import android.app.Activity;

import android.app.AlertDialog;

import android.view.Menu;

import android.view.MenuItem;

import android.support.v4.app.NavUtils;

import android.telephony.TelephonyManager;

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.util.ArrayList;

import org.apache.http.HttpEntity;

import org.apache.http.HttpResponse;

import org.apache.http.NameValuePair;

import org.apache.http.client.HttpClient;

import org.apache.http.client.entity.UrlEncodedFormEntity;

import org.apache.http.client.methods.HttpPost;

import org.apache.http.impl.client.DefaultHttpClient;

import org.apache.http.message.BasicNameValuePair;

import org.json.JSONArray;

import org.json.JSONObject;
```

```
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.app.Activity;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.graphics.drawable.Drawable;
import android.os.Bundle;
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.view.View.OnClickListener;
import android.view.View.OnKeyListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapView;
import android.content.Context;
import android.graphics.drawable.Drawable;
```

```
import android.telephony.TelephonyManager;
import android.util.Log;
import com.google.android.maps.GeoPoint;
import com.google.android.maps.MapActivity;
import com.google.android.maps.MapController;
import com.google.android.maps.MapView;
import com.google.android.maps.Overlay;
import android.content.Intent;
import android.graphics.Bitmap;
import android.graphics.BitmapFactory;
import android.graphics.Canvas;
import android.graphics.Point;
import android.location.Address;
import android.location.Geocoder;
import android.location.Location;
import android.location.LocationListener;
import android.location.LocationManager;
import android.os.Bundle;
import android.widget.Toast;
public class activity5 extends MapActivity implements LocationListener{
        private Button contactar;
        private LocationManager locationManager;
  class MyOverlay extends Overlay {
        GeoPoint point;
        public MyOverlay(GeoPoint point) {
                super();
                this.point = point;
        }
```

```
        @Override
        public boolean draw(Canvas canvas, MapView mapView, boolean shadow, long when) {
            super.draw(canvas, mapView, shadow);
            Point scrnPoint = new Point();
            mapView.getProjection().toPixels(this.point, scrnPoint);
            Bitmap marker = BitmapFactory.decodeResource(getResources(), R.drawable.icon);
            canvas.drawBitmap(marker,
                        scrnPoint.x - marker.getWidth() / 2,
                        scrnPoint.y - marker.getHeight() / 2, null);
            return true;
        }
    }
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.act5);
        String textDeviceID=null;
        //Primero obtene mos el imei del telefono
        TelephonyManager telephonyManager =
(TelephonyManager)getSystemService(Context.TELEPHONY_SERVICE);
        textDeviceID=getDeviceID(telephonyManager);
       final TextView imei = (TextView)findViewById(R.id.imei);
       //TextView imei=null;
       imei.setText("Identificador: " + textDeviceID);
        LocationManager locationManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);
        //updateLocation(locationManager.getLastKnownLocation(LocationManager.GPS_PROVIDER));
        //locationManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 6000, 50, this);
            Location loc =
                        locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
```

```
        //int im=Integer.parseInt(textDeviceID);
     // double lat = loc.getLatitude()*1E6;
        //double lon = loc.getLongitude()*1E6;
    final TextView latitud = (TextView)findViewById(R.id.latitud);
    final TextView longitud = (TextView)findViewById(R.id.longitud);
    latitud.setText("Latitud: " + String.valueOf(loc.getLatitude()));
    longitud.setText("Longitud: " + String.valueOf(loc.getLongitude()));
    MapView mapView = (MapView) findViewById(R.id.mapa);
   //new MyLocationListener(mapView.getController(), locationManager, getBaseContext());
    mapView.setBuiltInZoomControls(true);
    Drawable marker=getResources().getDrawable(android.R.drawable.star_big_on);
    int markerWidth = marker.getIntrinsicWidth();
    int markerHeight = marker.getIntrinsicHeight();
    marker.setBounds(0, markerHeight, markerWidth, 0);
    MyItemizedOverlay myItemizedOverlay = new MyItemizedOverlay(marker);
    mapView.getOverlays().add(myItemizedOverlay);
        Location loc1 =
                locationManager.getLastKnownLocation(LocationManager.NETWORK_PROVIDER);
    GeoPoint myPoint1 = new GeoPoint((int) (loc1.getLatitude() * 1E6), (int) (loc1.getLongitude() *
1E6));
    myItemizedOverlay.addItem(myPoint1, "taxi1", "taxi1");
    GeoPoint myPoint2 = new GeoPoint((int) (loc.getLatitude() * 1E6-3000), (int) (loc.getLongitude() *
1E6+3000));
    myItemizedOverlay.addItem(myPoint2, "taxi2", "taxi2");
        AlertDialog alertDialog = new AlertDialog.Builder(
        activity5.this).create();
  // Setting Dialog Title
        alertDialog.setTitle("Petición rechazada");
        // Setting Dialog Message
```

```
    //    alertDialog.setMessage("Elija una opción");              // Setting OK Button

          alertDialog.setButton("Buscar otro taxi", new DialogInterface.OnClickListener() {

                public void onClick(DialogInterface dialog, int which) {

                // Write your code here to execute after dialog closed

               // Toast.makeText(getApplicationContext(), "Cargando mapa..",
Toast.LENGTH_SHORT).show();

                     Intent i= new Intent(activity5.this, activity3.class);

                     startActivity(i);

                }

          });

          // Showing Alert Message

          alertDialog.show();

  }//cierre oncreate()

          String getDeviceID(TelephonyManager phonyManager){

           String id = phonyManager.getDeviceId();

           if (id == null){

            id = "not available";

           }

           int phoneType = phonyManager.getPhoneType();

           switch(phoneType){

           case TelephonyManager.PHONE_TYPE_NONE:

            return "NONE: " + id;

           case TelephonyManager.PHONE_TYPE_GSM:

            //return "GSM: IMEI=" + id;

                     return id;



           case TelephonyManager.PHONE_TYPE_CDMA:

            return "CDMA: MEID/ESN=" + id;

           default:}
```

```
        return "UNKNOWN: ID=" + id;
      }
            @Override
            protected boolean isRouteDisplayed() {
                  return false;
            }
            @Override
            public void onLocationChanged(Location location) {
                  updateLocation(location);
            }
            @Override
            public void onProviderDisabled(String provider) {
                  Intent intent = new Intent(
android.provider.Settings.ACTION_LOCATION_SOURCE_SETTINGS);
                  startActivity(intent);
            }
            @Override
            public void onStatusChanged(String provider, int status, Bundle extras) {}
            @Override
            public void onProviderEnabled(String provider) {}
            protected void updateLocation(Location location){
                  MapView mapView = (MapView) findViewById(R.id.mapa);
        MapController mapController = mapView.getController();
        GeoPoint point = new GeoPoint((int) (location.getLatitude() * 1E6), (int)
(location.getLongitude() * 1E6));
        mapController.animateTo(point);
        mapController.setZoom(30);
            }
}//cierre class
```

7.3 Base de Datos

Por último, se explica como se ha implementado la base de datos. Para realizar esta, se han hecho 5 tablas en las cuales se va almacenando la información para que el sistema funcione correctamente. A continuación se muestran cada una de las tablas y se explica para que se usan.

Tabla de clientes

Se usa para almacenar los clientes que acceden al sistema, y para saber los clientes que hay en activo o lo que es lo mismo, que están a la espera de un taxi. Una vez que el cliente recibe la confirmación de carrera este es eliminado de la tabla.

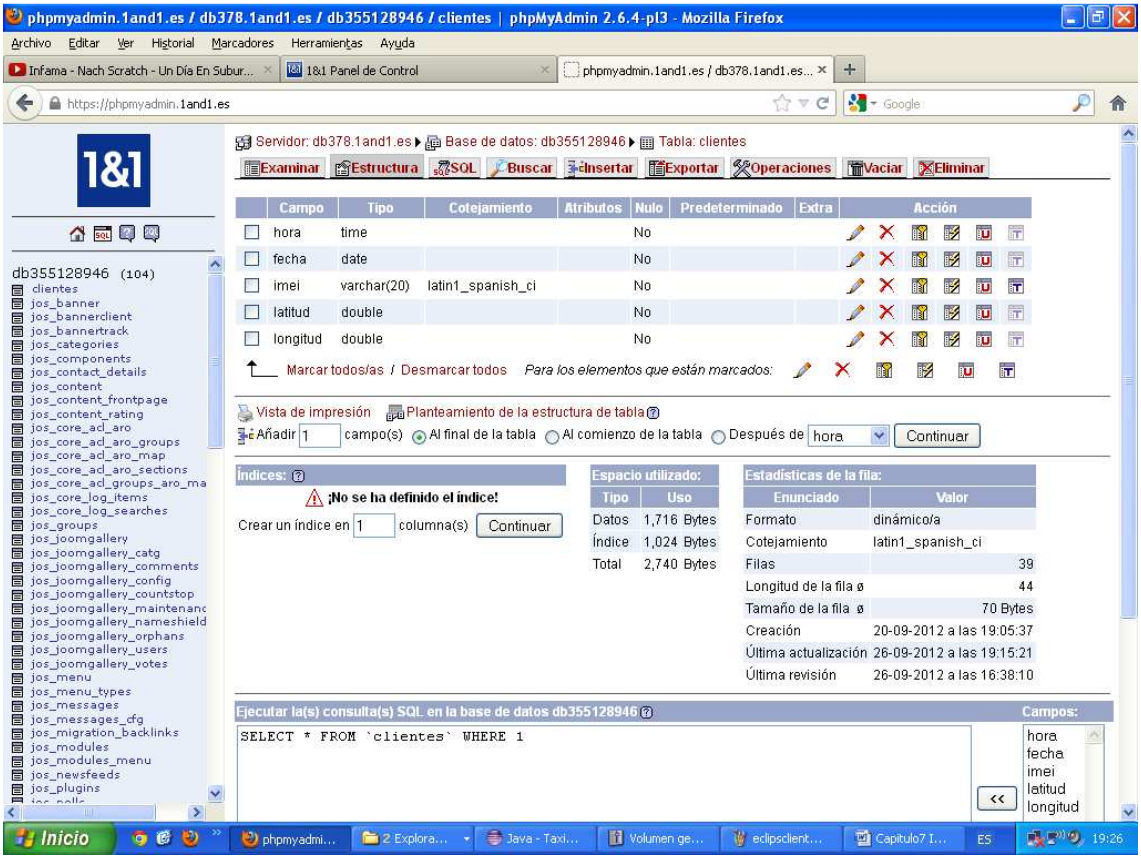


Imagen 7.3 Tabla de clientes

Tabla de taxistas

Se usa para almacenar los taxistas que acceden al sistema, y para saber los taxistas que hay en activo o lo que es lo mismo, que están a la espera de un cliente. Una vez que el taxista confirma la carrera este es eliminado de la tabla.

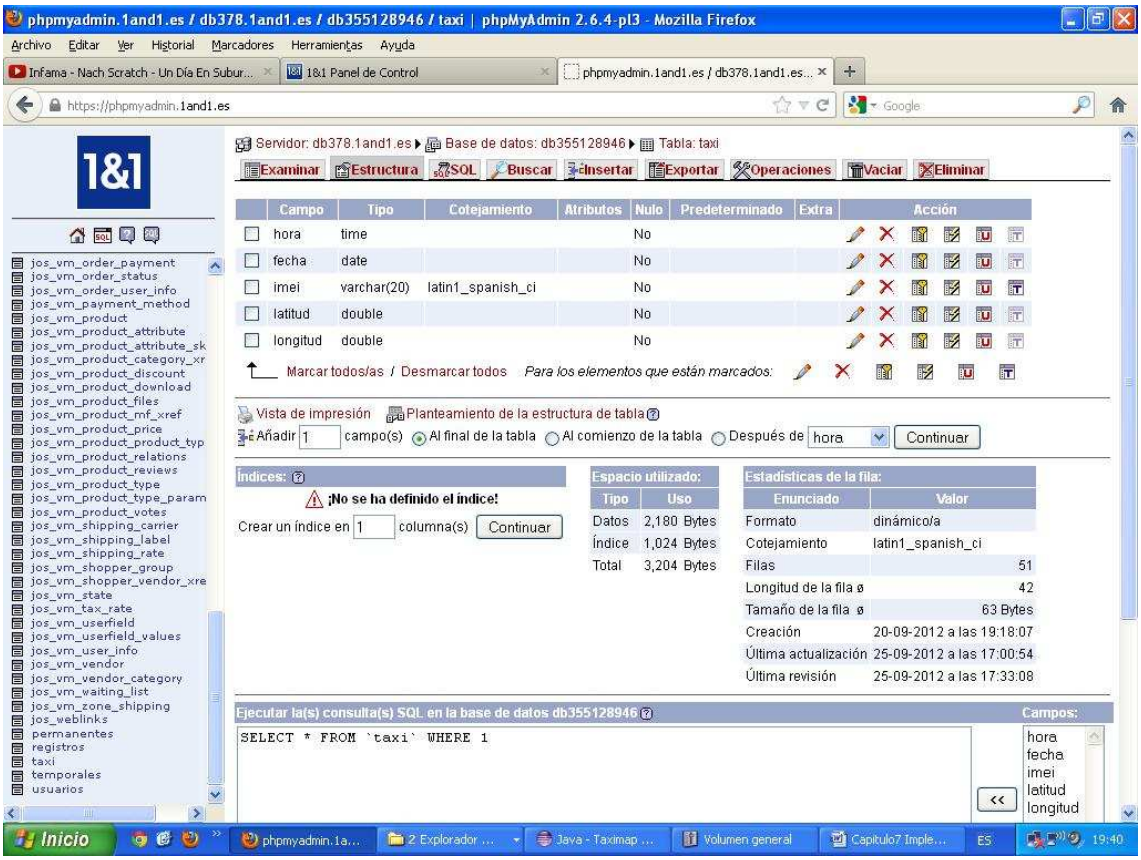


Imagen 7.4 Tabla de taxis

Tabla de usuarios

Se usa para guardar taxistas legales en el sistema. La aplicación taxista comprueba en esta tabla el login de usuario.

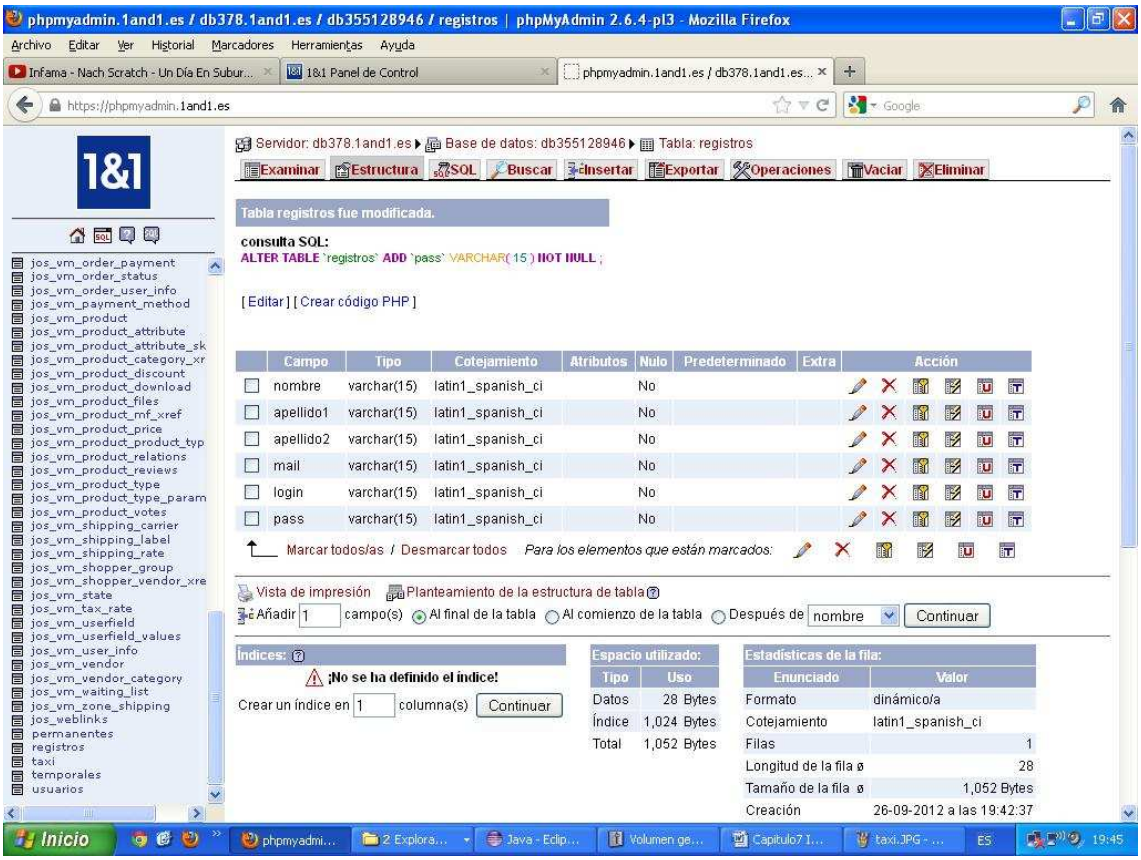


Imagen 7.5 Tabla de usuarios

Tabla de pedidos_temporales

Se usa para gestionar los pedidos de cliente a taxista y la respuesta de taxista a cliente. Cada vez que se confirma una carrera esta es eliminada de la tabla.

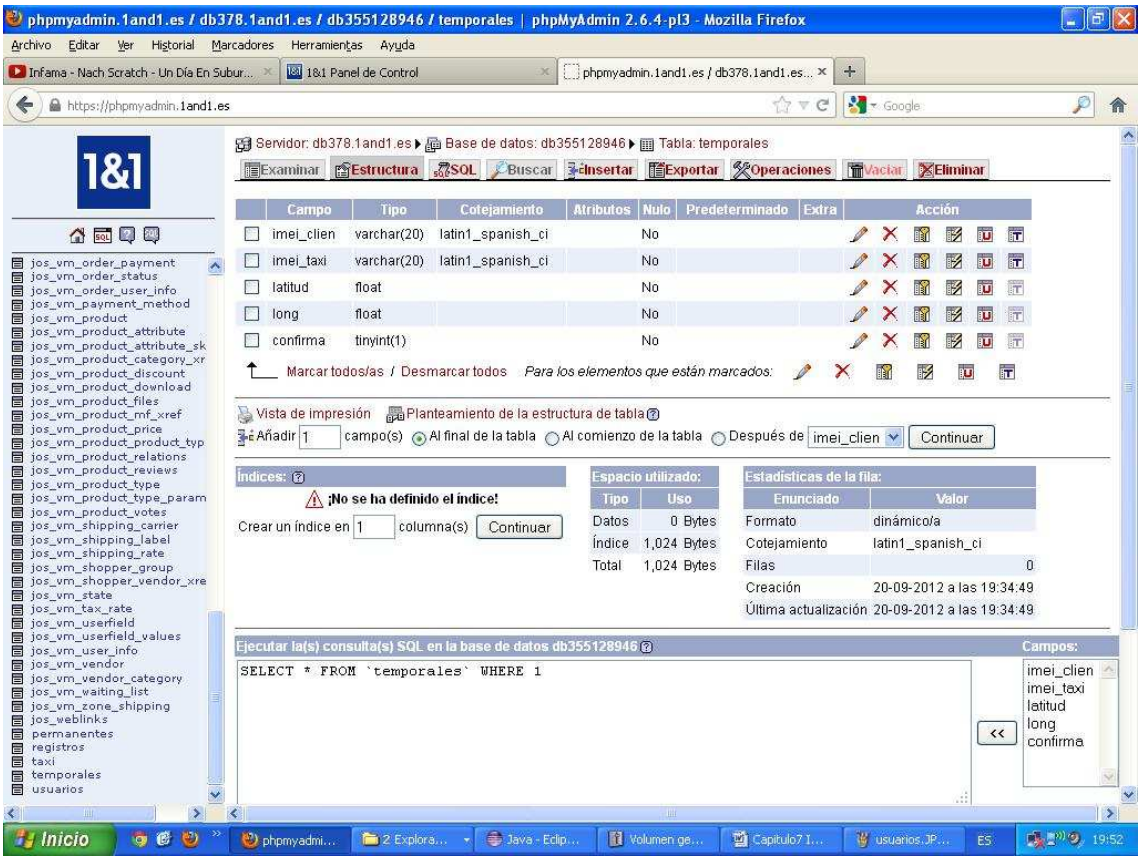


Imagen 7.6 Tabla de carreras temporales

Tabla de pedidos permanentes

Se usa para almacenar las carreras que han sido confirmadas positivamente de esta manera luego podemos obtener estadísticas del número de carreras por día, taxi..

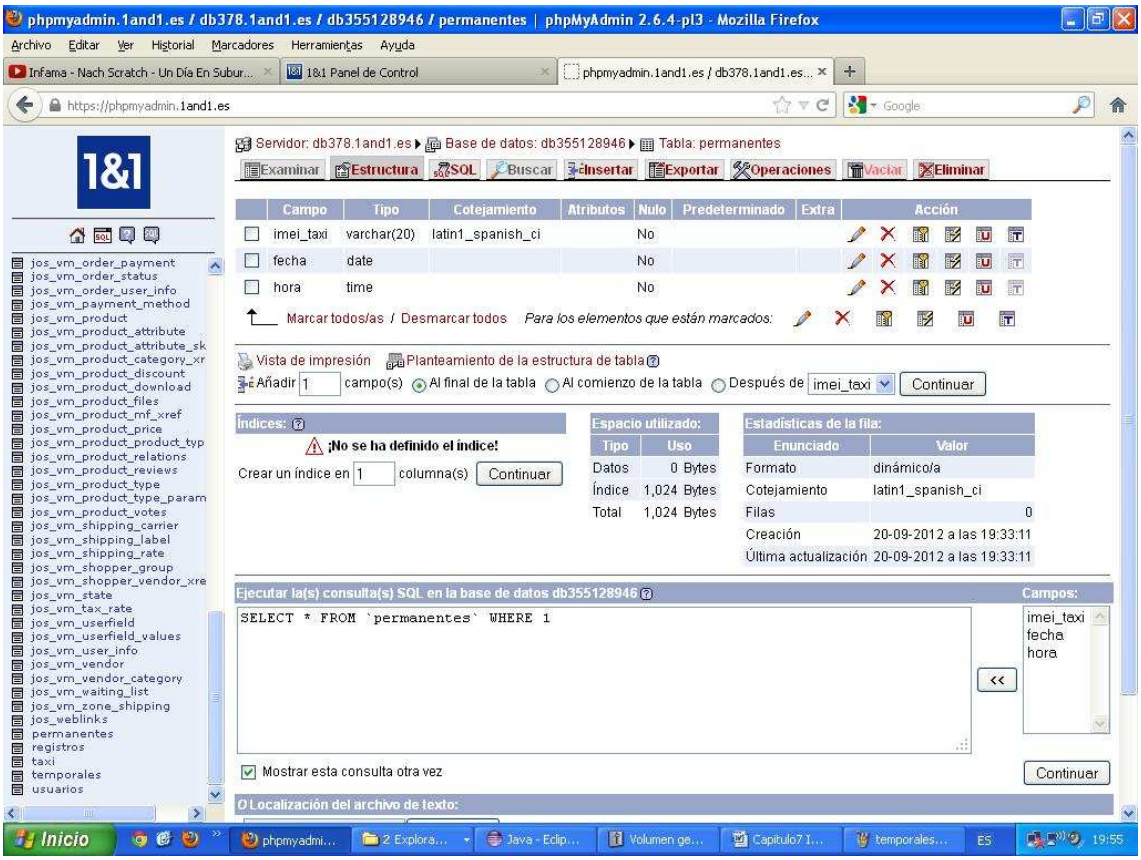


Imagen 7.7 Tabla de carreras permanentes

8. Conclusiones

Con la realización de este proyecto nos hemos dado cuenta del potencial que tiene el sistema operativo Android y que junto con las tecnologías o plataformas web puede ser de gran utilidad para la realización de multitud de sistemas que pueden ayudar a las personas, hacer que las empresas ahorren costes….

9. Lineas futuras

Se va a continuar con la realización de este proyecto para mejorar aspectos como:

- Interfaz de las aplicaciones
- Interfaz del centro de control
- Seguridad de las aplicaciones
- Seguridad del centro de control

En cuanto a la aplicación cliente se añadirá una opción que permita buscar taxis mediante mapa o bien hacerlo mediante la tecnología de realidad aumentada con la cual el usuario solo tiene que apuntar con la cámara de su móvil en un dirección para saber que taxis hay disponibles.

En cuanto a la aplicación taxista se añadirá un Arduino que irá conectado al taxímetro para que el taxista no tenga que estar activando la aplicación cada vez que deja a un cliente.

10. Bibliografía

http://es.wikipedia.org/wiki/Realidad_aumentada

http://www.maestrosdelweb.com/editorial/que-es-realidad- aumentada/

http://es.wikipedia.org/wiki/Servicio_general_de_paquetes_v%C3%ADa_radio

http://www.google.es/imgres?imgurl=http://www.e- global.es/b2b-blog/wp-images/graficos/HTC-Tattoo-1.jpg&imgrefurl

http://www.xatakamovil.com/aplicaciones/htc-tattoo- android-21-mas-cerca

http://www.testfreaks.es/telefonos-moviles/htc-tattoo/images/?page=2

http://appleweblog.com/2009/04/para-que-una-brujula- digital-en-el-iphone

http://3gmemories.com/2009/06/20/%C2%BFque-es-la- brujula-digital/

http://materiageek.com/2009/06/review-htc-magic-android/

http://es.wikipedia.org/wiki/Magnet%C3%B3metro

http://android.scenebeta.com/noticia/brujula

http://www.pce-iberica.es/instrumentos-de-medida/metros/acelerometros.htm

http://www.worldlingo.com/ma/enwiki/es/Accelerometer

"Android: Guía para desarrolladores" de Frank Ableson

http://es.onsoftware.com/p/programas-acelerometros-nokia
http://www.unandroideenvodafone.com/tag/brujula-digital/

http://es.wikipedia.org/wiki/JSON

http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=prototypejsAjaxJSON

http://nelopauselli.blogspot.com/2010/08/android- integrando-google-maps-con-el.html

"Introducción a Android y sus aplicaciones" de Jaume Pomés Olesti.

http://android.javielinux.com/locationgps.php

http://javiercancela.com/2009/08/04/realidad-aumentada- en-android-ar-compass-i/

http://awesomebytes.com/2010/07/31/usando-el-acelerometro-en-android/

http://vogella.googlecode.com/svn/trunk/de.vogella.android.locationapi/src/de/vogella/android/locationapi/

http://www.sgoliver.net/blog/?p=1267

Anexo I. Obtener una API Key para Google Maps

Para poder mostrar aplicaciones de geolocalización es necesario el uso de la clase

MapView, tal y como se explica en el apartado 3.3, para poder integrar *Google Maps* en la aplicación y mostrar mapas ya que proporciona un *wrapper* con su *API*.

Para poder acceder a dicha *API* es necesario registrarse en el servicio de *Google Maps* y aceptar los Términos de Servicio antes de que la clase *MapView* pueda mostrar los mapas. El registro es simple, gratuito y consta de dos partes:

1. Registrar la firma MD5 en el certificado que se usará para firmar la aplicación. Entonces, el servicio de registro de Mapas proporcionará una clave para la *API* que estará asociada al certificado de la aplicación.
2. Añadir la referencia a la clave en cada *MapView* (tanto en los archivos *XML* como directamente en el código).

En el caso de este proyecto, al estar en fase de *debug*, la aplicación se firmará en modo *debug* (el SDK utiliza automáticamente un certificado de *debug*). Para generar una firma MD5 con el certificado de *debug*, primero hay que localizar el *debug keystore*. En *Windows XP* se encuentra en C:\Documents and Settings\<user>\LocalSettings\ApplicationData\Android\debug .keystore y en *Linux* en ~/.android/debug.keystore

Una vez localizada la *keystore* se usa la herramienta *keytool* para obtener la firma MD5:

```
$ keytool -list -alias androiddebugkey
-keystore <path_to_debug_keystore>.keystore
-storepass android –keypass android
```

Figura 1. Obtención de la firma MD5

Una vez obtenida, se accede a la página *http://code.google.com/android/maps-API-signup.html* (ver Fig.2) y se siguen los siguientes pasos:

1. Se introduce la cuenta de *Google*
2. Se aceptan los Términos de Servicio de la *API* de *Android Maps*.
3. Se pega la firma MD5 en la casilla correspondiente.
4. Se clica en –Generar *API Key*

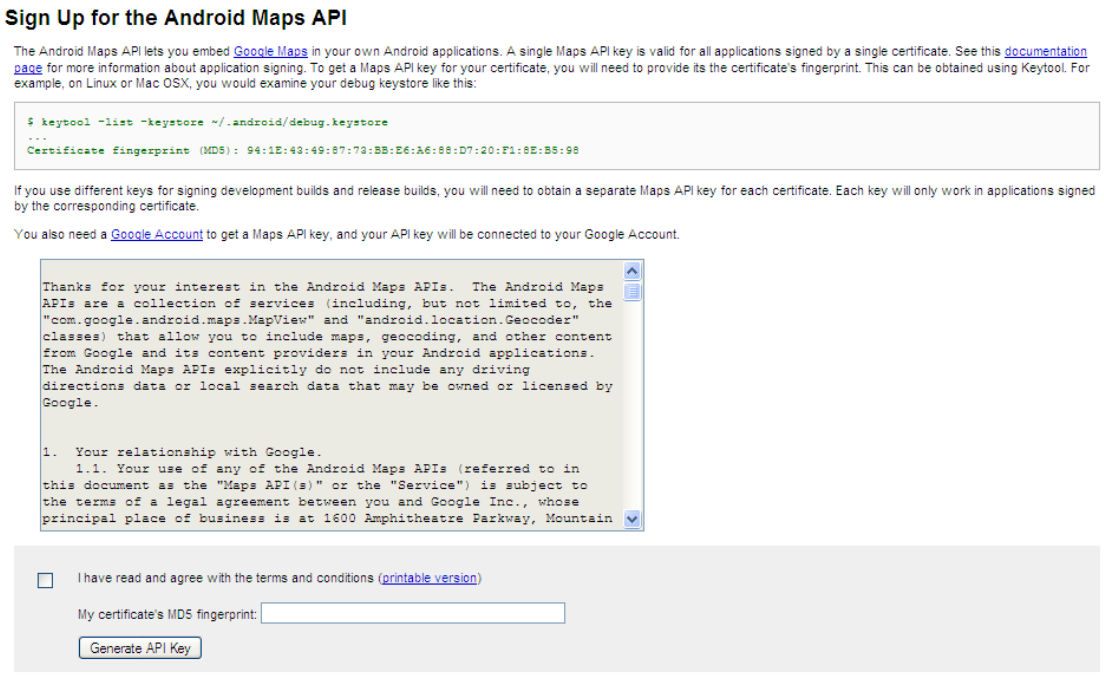


Figura 2. Web para obtener la Key para la API de Google Maps

Una vez obtenida la clave, hay que añadirla a la aplicación. En el caso de los ficheros *XML* se añade como atributo en los elementos <MapView> tal y como muestra la Fig.3:

```
<com.google.android.maps.MapView
  android:layout_width="fill_parent"
  android:layout_height="fill_parent"
  android:enabled="true" android:clickable="true"
  android:APIKey="APIKey"
  />
```

Figura 3. Inclusión de la API Key en un fichero XML

Para los objetos *MapView* declarados en el código directamente, se añade la clave como parámetro en el constructor (ver Fig. 4)

```
mMapView = new MapView(this, " APIKey ");
```

Figura 4. Inclusión de la API Key directamente en el código

Por último, hay que añadir la librería externa *com.google.android.maps* en el fichero *AndroidManifest* tal y como muestra la Fig. 5

```
    <manifest
XMLns:android="http://schemas.android.com/apk/res/androi
d" package="com.example.package.name">
...
<application android:name="MiAplicacion"  >
       <uses-library android:name="com.google.android.maps"  />
       ...
</application>
```

Figura 5. Inclusión de la librería com.google.android.maps en la aplicación