UNIVERSIDAD TÉCNICA PARTICULAR DE LOJA

La Universidad Católica de Loja

TITULACIÓN DE INGENIERO EN SISTEMAS INFORMÁTICOS Y COMPUTACIÓN

Implementación de una Aplicación Android basada en Realidad Aumentada aplicada a Puntos de Interés de la UTPL

Trabajo de fin de titulación.

Autor:

Rodrigo Alexander Saraguro Bravo

Director:

Ing. Nelson Piedra

LOJA – ECUADOR

2012

CERTIFICACIÓN

Ingeniero.

Nelson Oswaldo Piedra Pullaguari.

DIRECTOR DEL TRABAJO DE FIN DE TITULACIÓN

C E R T I F I C A:

Que el presente trabajo, denominado: “Implementación de una Aplicación Android basada en Realidad Aumentada aplicada a Puntos de Interés de la UTPL" realizado por el profesional en formación: Saraguro Bravo Rodrigo Alexander; cumple con los requisitos establecidos en las normas generales para la Graduación en la Universidad Técnica Particular de Loja, tanto en el aspecto de forma como de contenido, por lo cual me permito autorizar su presentación para los fines pertinentes.

Loja, 15 de diciembre de 2012

f) . . . . . .

CI: . . . . . .

AUTORÍA

El presente proyecto de tesis con cada una de las observaciones, análisis, evaluaciones, conclusiones y recomendaciones emitidas, son de absoluta responsabilidad del autor.

De igual manera, es necesario indicar que la información de otros autores incluida en el presente trabajo se encuentra debidamente especificada en las fuentes de referencia y apartados bibliográficos.

F.----------

Rodrigo Alexander Saraguro Bravo

CESIÓN DE DERECHOS

Yo, Rodrigo Alexander Saraguro Bravo, declaro ser autor del presente trabajo y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales.

Adicionalmente declaro conocer y aceptar la disposición del Art. 67 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: "Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado que se realicen a través, o con el apoyo financiero, académico o institucional (operativo) de la Universidad".

F. . . . . .

Rodrigo Alexander Saraguro Bravo

AGRADECIMIENTO

Mi más sincero agradecimiento a mis padres, cuya confianza y fortaleza ha sido depositado en mí hoy y siempre; de igual forma a mis hermanos y familiares cercanos por contribuir de alguna manera en cumplir esta meta.

A mi director de Tesis Ing. Nelson Piedra, por la oportunidad que me ha brindado, su excelente dirección, su tiempo de dedicación y motivación hacia mí, puntos clave para la culminación del trabajo.

A mis profesores de la carrera por sembrar en mí los conocimientos necesarios que me han servido de mucha ayuda en todo este proceso.

Y a compañeros y amigos cercanos que supieron brindarme su apoyo desinteresado en el transcurso de este objetivo.

Rodrigo Alexander Saraguro Bravo

DEDICATORIA

En primer lugar quiero dedicar el presente trabajo a mis padres, por enseñarme a luchar por mis metas, darme su apoyo permanente y completo siempre, al igual que toda mi familia.

A mis compañeros que día a día supieron brindarme su amistad, confianza y respeto.

A esa persona especial que ha confiado en mí y es mi mayor motivación.

Rodrigo Alexander Saraguro Bravo

ÍNDICE GENERAL DE CONTENIDOS

A. PÁGINAS PRELIMINARES

B. TEXTO CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

GLOSARIO

Android: Es un sistema operativo basado en el núcleo Linux diseñado originalmente para dispositivos móviles, tales como teléfonos inteligentes.

SDK: Es un conjunto de herramientas y programas de desarrollo que permite al programador crear aplicaciones para un determinado paquete de software, estructura de software, plataforma de hardware

AVD: (Android Virtual Device) una configuración del emulador que permite modelar un dispositivo real mediante la definición de opciones de hardware y software

API: (Application Program Interface) Interfaz de Programación de Aplicaciones, es el conjunto de funciones y procedimientos o métodos, en la programación orientada a objetos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

POI: (Point of interest) Punto de interés es un lugar punto específico que alguien pueda encontrar útil o interesante.

AR/RA: (Augmented Reality) Realidad aumentada, combinación de objetos reales y virtuales.

RESUMEN EJECUTIVO

Construcción de un cliente móvil Android para el servicio de puntos de interés de la UTPL, y un administrador web para la operación de datos del sistema. Se ha implementado una arquitectura distribuida a través de webservices que hacen el consumo de información de manera dual entre DBpedia y una base de datos relacional MySQL. El objetivo principal de esta aplicación móvil es difundir información de la UTPL a través de tecnologías como la realidad aumentada y geolocalización, sobre puntos de interés como: sitios del campus, centros universitarios, paradas de bus del transporte estudiantil y demás sitios importantes de la ciudad; lo que permitirá un mejor acceso a esta información tanto para estudiantes, personal y visitantes interesados en esta información.

La presente tesis inicia con el estudio del estado del arte, continua con el desarrollo de la problemática y objetivos, luego se procede a la implementación del proyecto y se finaliza con las respectivas conclusiones y recomendaciones.

CAPITULO I

1. ESTADO DEL ARTE

1.1. INTRODUCCIÓN

En la última década las nuevas tecnologías, en especial los dispositivos móviles han avanzado a un gran nivel en el que se dispone de excelentes aplicaciones capaces de gestionar grandes cantidades de información y de realizar costosas tareas y operaciones; por tal razón tenemos la oportunidad de usar aplicaciones en cualquier momento o situación en el día a día. Y si a ello se le agrega que hoy en día es posible conocer nuestra propia ubicación en tiempo real, se da la posibilidad de desarrollar una herramienta capaz de conocer, analizar y almacenar información sobre lo que nos rodea.

Una de las tecnologías existentes que aplica estos servicios para dispositivos móviles es la Realidad Aumentada, quien combina elementos reales y virtuales para facilitar nuestra visión del mundo y cambia la forma de acceder a la información. Con potentes dispositivos móviles es sencillo navegar por nuestro entorno visual y obtener mágicamente más información sobre el mismo, por ejemplo: la altura de un edificio, los departamentos que puede tener, personas importantes que trabajan en él, la distancia al café más cercano, recomendaciones para el postre que vamos a pedir en la cafetería, tweets que se están realizando en el campus, lista de eventos, etcétera.

Un agregado más a esta tecnología es que se trabaja con una plataforma libre y con gran demanda como Android, existen IDEs gratuitos como Eclipse y su SDK, que facilitan el desarrollo de sus aplicaciones basadas en lenguaje Java y ejecutadas en su propio emulador; y contar con la mayor comunidad de desarrolladores, se esperan excelentes resultados en el desarrollo de este proyecto.

1.2. REALIDAD AUMENTADA

1.2.1. Definición

La realidad aumentada (Azuma, 1995) es el término usado para definir un tipo de tecnología donde la visión de la realidad se amplía con elementos virtuales que añaden

información digital. Así definimos esta tecnología como un punto intermedio entre la realidad tal y como a conocemos y la realidad virtual. Se basa en tecnologías derivadas de la visualización o reconocimiento de la posición para crear un sistema que reconozca la información real que tenemos alrededor y cree una nueva capa de información. Esta información, se mezcla con el mundo real de forma que para el usuario coexistan objetos virtuales y reales en el mismo espacio. En función de la aplicación, la información agregada virtualmente podría ser textual, con íconos, sonidos o multimedia.

El objeto principal de la realidad aumentada es mejorar la percepción que tiene el usuario sobre su entorno y permitir nuevas formas de interacción mediante la visualización de información que el usuario no puede percibir con sus sentidos (Azuma, 1995). Actualmente existen muchas aplicaciones con sistemas de AR para visualización médica, la asistencia para la fabricación y reparación de maquinaria pesada, etc.

La figura 1 representa un esquema para sistemas basados en realidad aumentada, el mismo que tiene varios componentes: los entornos real y virtual, la aplicación o sistema, la plataforma en la que funciona el sistema, así como los servicios web que trabajan para el funcionamiento del entorno virtual.

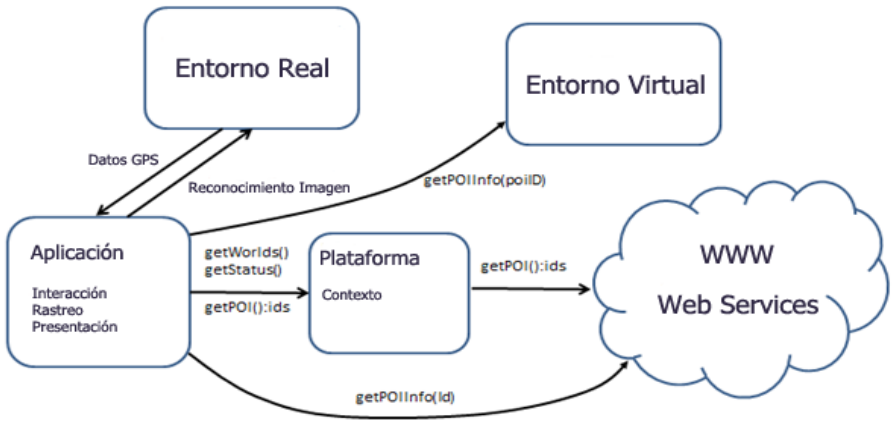

Fig. 1. Esquema de un sistema basado en realidad aumentada [1]

[1] Arquitectura Realidad Aumentada de LAYAR: http://layar.pbworks.com/w/page/7783214/Layar-Platform-Architecture-Overview

1.2.2. Características

Un sistema de realidad aumentada cuenta con las siguientes propiedades (Azuma R, 1995):

- Combina objetos reales y virtuales.
- Funciona en tiempo real.
- Se registra en tres dimensiones. Ya que la información virtual añadida normalmente se registra en un lugar del espacio.

1.2.3. Tipos de reconocimiento

Dentro del campo de la RA, existen algunos tipos de reconocimiento con los que se trabaja (Bover, 2010), siendo esta tarea la parte más costosa de la realidad virtual, y dependiendo de la técnica se requerirá de un tipo de hardware u otro. Estos son:

1.2.3.1. Reconocimiento basado en marcadores

Este tipo de registro utiliza imágenes del entorno como referencias a las que llamaremos marcadores, las cuales son reconocibles por el dispositivo. El dispositivo reconoce estos marcadores en la imagen recibida por su cámara, y con esto consigue la información de la posición del dispositivo con la cual coloca correctamente las imágenes de realidad aumentada. Este cálculo de posición se hace a través del análisis de la distancia del marcador, según su tamaño, y el ángulo en que se encuentra respecto a la cámara. Difícilmente funciona en ángulos muy cerrados.

1.2.3.2. Reconocimiento basado en objetos

Su implementación es complicada y tiene mayor coste nivel de computacional. Funciona a través del sensor de cámara del dispositivo y trata de reconocer un objeto en particular, comparándolo con una base de datos de objetos según sea su forma para descubrir de qué objeto se trata. Claramente, este sistema no requiere disponer más que una cámara en él dispositivo, y no necesita modificar el entorno para que funcione, lo que la hace totalmente portable de un entorno a otro con toda facilidad.

1.2.3.3. Reconocimiento basado en geolocalización

Este sistema requiere de un sistema de localización (GPS), y de sistemas que reconozcan la orientación del dispositivo (brújulas digitales, acelerómetros, etc). Trabaja en función de puntos de referencia como las coordenadas, entonces el dispositivo aproxima el objeto de acuerdo a su ángulo de visión, y su distancia.

1.2.4. Requerimientos

Para la implementación de aplicaciones basadas en la realidad aumentada, se requieren las siguientes especificaciones como mínimo:

- Un dispositivo que sea capaz de reconocer el entorno.
- Una cámara que capte la imagen que estamos enfocando para que luego se le añada la información digital.
- Una unidad de proceso y software especializado que sea capaz de gestionar los diferentes recursos necesarios para hacer funcionar la realidad aumentada
- Una base de conocimiento que proporcione los datos para generar esta capa virtual de información.

Por el momento no existen dispositivos exclusivos para la Realidad Aumentada (CONSUMER, 2011), por tanto se trabajan con las tecnologías disponibles:

Un computador: Debido a su potente hardware cumplen con facilidad los requisitos técnicos para brindar este servicio.

Un teléfono inteligente: Debido a su portabilidad los convierte en el elemento idóneo para soportar realidad aumentada, siempre y cuando tengan acceso inalámbrico a internet.

Tabletas: Si el dispositivo cuenta con una cámara puede cumplir con los requerimientos.

1.2.5. Diferencia Realidad Virtual y Realidad Aumentada

La realidad virtual ha sido un tema de gran interés; menos atención se ha prestado al campo relacionado de la realidad aumentada, a pesar de su potencial similar. La

diferencia entre la realidad virtual y realidad aumentada es en su trato con el mundo real. La realidad virtual sumerge a un usuario dentro de un mundo virtual que reemplaza completamente el mundo real, un ejemplo conocido es SecondLife[2]. Por el contrario, la realidad aumentada permite al usuario ver el mundo real que lo rodea y aumentar la vista del usuario del mundo real mediante la superposición o composición de objetos tridimensionales virtuales con sus contrapartes del mundo real; lo mejor de la realidad aumentada es que los objetos reales y virtuales coexisten.

1.2.6. Frameworks y Aplicaciones

Existen en la web algunos componentes de Apis, y librerías gratuitas que ayudan con el reconocimiento de imagen y el posicionamiento espacial de zonas reconocidas utilizadas en realidad aumentada. Por ejemplo, podemos citar entre otros muchos:

ArToolkit: Es una librería de software para construir aplicaciones con RA.

BuildAr: Es una librería que facilita la creación de marcas y escenas de AR. Ofrecen una versión gratuita y una versión Premium.

Atomic: Es un software libre que permite la creación de escenas de AR, también con animaciones.

A la par existe gran variedad de aplicaciones que trabajan con realidad aumentada, donde únicamente se requiere instalar la aplicación en el dispositivo para su funcionamiento y contar con una buena conexión de Internet. A continuación se mencionan las aplicaciones más descargadas sobre realidad aumentada (ComPixels, 2010):

Junaio: es una aplicación disponible para iphone, y está actualmente en desarrollo para Android. Funciona a través de la posición y orientación actual del dispositivo. Además que tiene una parte abierta para desarrolladores y puedan crear sus canales de POI (puntos de interés), en su propio servidor.

[2] Secondlife: http://secondlife.com/

Layar: es el navegador de esta tecnología. Esta herramienta permite rápidamente agregar layers (capas) que funcionan de una manera similar a los complementos de un navegador web normal. Cada capa agrega información y complejidad a su "realidad aumentada". Actualmente permite usar 312 capas diferentes, por ejemplo: se puede ver los Tweets cercanos, información de casas a la venta, restaurantes cercanos, consultar la wikipedia, etc.

Wikitude: Es una aplicación tipo Android que funciona como una enciclopedia futurista, así se rompió la línea del tiempo, ya que tenemos una enciclopedia del Siglo XXII. La información que se muestra es más ordenada que en Layar, lo cual en muchas ocasiones es mejor. Actualmente hay una versión denominada Wikitude AR Travel Guide.

Satellite AR, es una aplicación que permite mostrar información sobre los satélites que hay en la órbita de la tierra, por ejemplo: su trayectoria y posición. Simplemente se debe apuntar hacia el cielo el dispositivo móvil en cuestión, de tal forma que la cámara quede dirigida hacia él y con la ayuda del GPS sabremos donde se localizan.

1.2.7. Proyectos similares

1.2.7.1. Proyectos educativos

La realidad aumentada puede proporcionar grandes beneficios en cuanto al aprendizaje y la formación en campos tan diversos como el comercio, el ejército y la medicina. Actualmente existen muchos proyectos enfocados al conocimiento y desarrollo de habilidades para niños, entre ellos se mencionan los siguientes proyectos:

Magic Book (Doreen, 2003) del grupo activo HIT de Nueva Zelanda. Este proyecto permite al alumno leer un libro real a través de un visualizador de mano y ve sobre las páginas reales contenidos virtuales. De esta manera cuando el alumno ve una escena de Realidad Aumentada que le guste puede introducirse dentro de la escena y experimentarla en un entorno virtual inmersivo.

Water on Tap (KAUFMANN & MEYER, 2010), es una colección de mundos virtuales inmersivos Newtonworld, MaxwellWorld y PaulingWorld. *NewtonWorld* proporciona un entorno para la investigación de la cinemática y la dinámica de un movimiento unidimensional. *MaxwellWorld* apoya la exploración de la electrostática, hasta el concepto de la ley de Gauss. *PaulingWorld* permite el estudio de estructuras moleculares a través de una variedad de representaciones. Estudios de evaluación formativa de estos mundos virtuales se han realizado con respecto a la usabilidad y la capacidad de aprender.

Al mismo tiempo, existen muchos proyectos desarrollados por Institutos y Universidades de Estados Unidos y Alemania; con objeto de uso a nivel secundario, universitario y científico:

HANDHELD AUGMENTED REALITY PROJECT (HARP, 2012), con el financiamiento del Departamento de Educación de EE.UU. Star Schools Program, los investigadores de la Escuela de Educación de Harvard, junto a la Universidad de Wisconsin de Madison, y el programa de formación al docente en el MIT han desarrollado un juego basado en "realidad aumentada", diseñado para enseñar matemáticas y ciencias para la alfabetización de los estudiantes de secundaria. El juego se juega en un equipo Dell Axim de mano y usa el Sistema de Posicionamiento Global (GPS) para relacionar la ubicación de los estudiantes del mundo real a su ubicación virtual en el mundo digital del juego. Como los estudiantes se mueven en torno a una ubicación física, tales como su patio de la escuela o campos deportivos, un mapa en sus pantallas de mano los objetos digitales y personas virtuales que existen en un mundo de realidad aumentada superpuesta en el espacio real. Esta capacidad es paralela a la nueva forma de recopilación de información, comunicación y expresión, es posible gracias a nuevos medios interactivos (como la Web, GPS, teléfonos celulares con mensajes de texto, vídeo y funciones de cámara).

Augmented Reality Delivery Simulator for Medical Training (Sielhors, Obst, Burgkart, & Riener, 2004), este simulador fue construido en el Instituto de

Informática junto al Hospital de la Universidad de Balgrist, en Munchen-Alemanía y consiste de un modelo hardware y software que simula el parto de un bebé. El modelo de hardware es un cuerpo fantasma femenino que tiene dentro una cabeza de un bebé. El modelo de software en cambio se divide en dos partes. La parte fisiológica que ofrece en tiempo real (4 ms) los valores de presión arterial, ritmo cardiaco, el dolor y el suministro de oxígeno, que se da al usuario. Los valores calculados generados mediante la contracción de oxitocina, la fatiga de la madre y el feto, el suministro de oxígeno al niño, y las condiciones individuales de contorno. Estas condiciones de contorno pueden ser por ejemplo, en la madre: la sensibilidad de la producción de oxitocina, la tensión y el dolor, el volumen del corazón, así como muchos parámetros fetales. Además que uno de los componentes ofrece la visualización en 3D de la posición de la cabeza dado por la posición de la cabeza del bebé.

ARMed (Lazar, Nikolić, Nikolić, & Predrag) representa un sistema informático capaz de ayudar a los cirujanos en el examen del paciente, estableciendo el diagnóstico y la preparación de la cirugía. Su asistencia daría un gran enriquecimiento de los puntos de vista sobre el objeto de la intervención (órganos y vasos sanguíneos). Y así el sistema proporcione información adicional de interés para la cirugía. De esta manera un cirujano sería capaz de tener conocimiento del problema que debe resolverse, y la vista en 3D del cuerpo que es objeto en la operación, incluso antes de empezar la cirugía. El sistema está diseñado para permitir el uso de diferentes paquetes para la simulación biomédica. Los modelos 3D pueden obtener con la reconstrucción estándar de imágenes médicas, rayos X, resonancia magnética, tomografía computarizada, etc.

1.2.7.2. Otros proyectos relacionados

En la web podemos encontrar muchos proyectos relacionados a la RA para dispositivos móviles, entre los que se destacan:

HANDHELD AUGMENTED REALITY (Handheld Augmented Reality, Christian Doppler Laboratory, 2011) es un laboratorio de investigación de realidad

aumentada en dispositivos móviles. Dentro de sus proyectos actuales se encuentran:

Social AR[3] es un proyecto que apunta a nuevos conceptos de realidad aumentada móvil que permite a personas inexpertas en el tema crear contenido para aplicaciones móviles de AR. Dentro de este proyecto se han creado técnicas y aplicaciones que permiten la publicación de contenidos en entornos no preparados, así como el intercambio de contenidos entre los usuarios.

Real-Time Self Localization[4] es un proyecto destinado a la localización en tiempo real de un teléfono móvil con la visión y sensores. El marco de localización desarrollado se basa en escasas nubes de puntos 3D y los últimos logros en función de seguimiento natural. Las nubes de puntos 3D escasa se crean a través de una tubería de reconstrucción en 3D con imágenes captadas desde una cámara réflex digital de alta calidad como entrada.

Sistema de posicionamiento 3D sin marcadores basado en visión por computador (LABEIN, 2011), El objetivo principal del proyecto dentro de esta línea es desarrollar un método para estimar la posición, orientación y el movimiento tridimensional de una cámara a partir de las imágenes capturadas, utilizando para ello una única cámara calibrada y sin necesidad de añadir ningún tipo de marcador en la escena. Para la validación del método, el demostrador estará orientado al e-learning y formación sobre sistemas mecánicos.

1.2.8. Últimos hechos sobre RA

Según estadísticas realizadas por el equipo de HiddenCreative, referentes al uso de aplicaciones de RA en el ámbito comercial (HiddenCreative, 2011), se ha determinado que los tres principales usos son para el desarrollo de folletos virtuales, lanzamientos y

[3] Adaptado de Handheld Augmented Reality , Social Augmented Reality: http://handheldar.icg.tugraz.at/socialar.php

[4] Adaptado de Handheld Augmented Reality, Full 6-Dof Localization Framework: http://handheldar.icg.tugraz.at/localization.php

concursos, así como eventos y conferencias (fig. 2), lo que deduce que tiene gran demanda en el desarrollo de aplicaciones que trabajen con datos recientes.

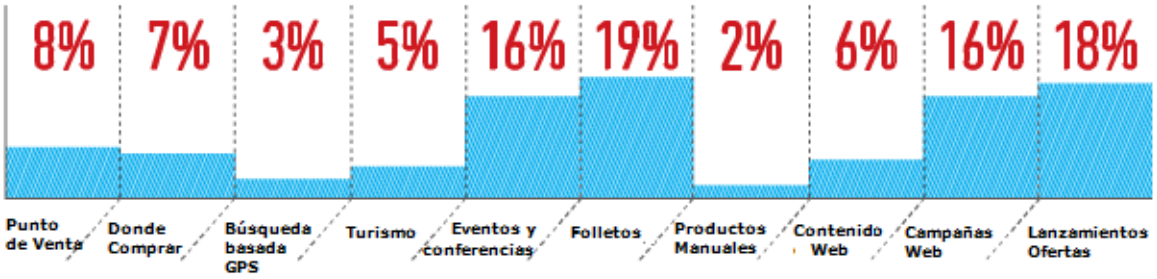

Fig. 2. Principales usos de la Realidad Aumentada

1.3. MAPAS Y GEOLOCALIZACIÓN

1.3.1. Mapas

Un Mapa es un despliegue gráfico sobre información geográfica que permite entender las distancias, referencias y puntos importantes en relación a un lugar y permite mostrar a un usuario sus puntos de interés.

1.3.1.2. Mapas digitales

Existen gran variedad de mapas SIG de carreteras, algunos abiertos como código Open Street Maps, creado por una comunidad de internet sin ánimo de lucro, y otros propios y privados de algunas empresas, como Nokia. Categorizaremos pero, en accesibles por internet o accesibles en memoria, ya que esta clasificación se adapta más a las necesidades de nuestro proyecto. Vemos algunos ejemplos de mapas de carreteras.

- ***Por Internet:***

Google Maps, Open Street Maps, CloudMade, Navteq, Bing Maps.

- ***Descargables:***

TomTom, Garmin, Nokia Ovi Maps

- ***Apis y Librerías***

A continuación se mencionan algunos de los servicios y librerías que pueden ser usadas para el desarrollo de aplicaciones con mapas y geolocalización:

a. Google Maps

Google Maps ofrece la capacidad de hacer acercamientos o alejamientos de un mapa virtual. Permite desplegar información de manera fácil y económica.

Los resultados de la búsqueda pueden ser restringidos a una zona, gracias a Google Local. Por ejemplo, si alguien quiere consultar por "Waffles in Ottawa" (en español, Waffles en Ottawa), para encontrar restaurantes que sirven waffles cerca de la ciudad. Las búsquedas pueden encontrar una amplia gama de restaurantes, hoteles, teatros y negocios generales.

Como otros servicios de mapa, Google Maps permite la creación de pasos para llegar a alguna dirección. Esto permite al usuario crear una lista paso a paso para saber el cómo llegar a su destino, calculando el tiempo necesario y la distancia recorrida entre las ubicaciones co-arqueólogas.

b. OpenStreetMap

OpenStreetMap es un proyecto de software libre, el objetivo de este proyecto es brindar una solución de mapas digitales parecido al servicio de Google Maps, pero con la opción de nutrir los mapas de manera comunitaria, subiendo las trazas de GPS al portal o con distintas herramientas con las que cuenta este proyecto para el cargado y edición de datos sobre los servidores.

Gracias al aporte que realiza la comunidad de usuarios, este proyecto ha crecido bastante permitiendo así contar con mapas de distintas partes incluso de pueblos y localidades pequeñas, en este mapa podemos

encontrar la ciudad de Loja de una manera detallada mucho mejor que la que se puede encontrar en Google Maps, gracias a la facilidad todas las personas que cuenten con un GPS pueden hacer sus aportaciones de una manera rápida y precisa.

c. MGMaps Lib SKD

MGMaps es una API libre escrita en Java que proporciona una interfaz para mostrar mapas geocodificados de manera sencilla. Esta plataforma está disponible para uso en Android, BlackBerry y J2ME. Permite mostrar mapas, definir rutas, definir puntos de interés, búsquedas por geocodificación (direcciones, lugares…). Su licencia GPL permite usarla para proyectos de código abierto, aunque también dispone una licencia comercial.

1.3.2. Geocodificación

La geocodificación es el proceso de asignar coordenadas geográficas a puntos del mapa (que pueden representar diferentes puntos de interés). Con esta información, el SIG es capaz de especular generando información que no dispone a través de la que conoce. Por ejemplo, si se conoce algunos números de casas de una calle, se puede interpolar, suponiendo que entre casa y casa hay la misma distancia, la posición de los otros números.

Otro sistema parecido es la geocodificación inversa. En este caso, lo que se hace es asignar un punto de interés a unas coordenadas concretas. Un ejemplo, podría ser, con una aplicación móvil, estando en frente de un museo, guardar en una base de datos su posición.

1.3.2.1. SIG

Los SIG son sistemas de mapas digitalizados que guardan datos geográficos diseñados para que, a través de un software, se puedan capturar, almacenar, editar, estudiar y analizar de manera sencilla haciendo uso de las nuevas tecnologías. Generalizando, se podría decir que se trata de una base de datos, sobre la cual se pueden hacer consultas de manera sencilla sobre datos

geográficos. Versiones conocidas de un SIG son por ejemplo, aplicaciones como Google Maps, o Google Earth. Los SIG tienen múltiples funciones en diferentes ámbitos, como por ejemplo, calcular las rutas más cortas, calcular niveles freáticos del suelo o análisis catastróficos.

Funcionan a través de lo que se denominan capas. Usando un mismo sistema de coordenadas, cada capa tiene un nivel diferente de información agrupada, como por ejemplo, una capa de carreteras, otra de ríos, otra de información de altitud. Esto permite añadir o quitar información asociada de manera sencilla para casa caso. También permite, focalizando en un punto, obtener información de cada una de las capas. Los SIG pueden resolver:

- Localización: Propiedades de una coordenada
- Condición: Se lanza una condición para ver qué zonas la cumplen y cuáles no.
- Tendencia: Comparación entre dos posiciones distintas.
- Rutas: Cálculos de rutas óptimas.
- Modelos: Generar modelos para ver como una determinada acción influye en el mapa.

Representación de datos

Los elementos de una capa de un SIG se pueden representar de diferentes maneras, según su uso, los dos más conocidos son el método Raster, y el método Vectorial, aunque el método vectorial es el usado más comúnmente.

a. El método Raster

Se basa en definir toda la zona que abarca un mapa en una malla. Así, nos quedan un gran número de pequeñas celdas regulares, a cada cual se le asigna una propiedad dentro de la capa. Tiene diferentes sistemas de almacenamiento, pudiendo representarse en ficheros binarios

(BLOB), o en ficheros de imagen JPEG, asignando un color a cada propiedad, por lo tanto, asignando colores a las celdas.

b. El método Vectorial

Este es el sistema más utilizado a la hora de representar los datos en capas. Más que definir propiedades de pequeñas celdas, lo que busca es definir con precisión la localización de diferentes elementos geográficos. Para guardar esta información, se utilizan tres elementos geométricos, el punto, la línea y el polígono. En el punto, se asigna a una coordenada específica una propiedad. Esto no influye pues, a las coordenadas que le envuelven, como pasaría en el método raster. Suelen ser usadas para definir puntos de interés. En una línea, se asigna una propiedad a todas las coordenadas que van desde un cierto punto hasta otro. Suelen ser usadas para definir ríos, carreteras, líneas ferroviarias, etc. Suelen ser usados para definir edificios, parcelas, lagos, etc.

c. Renderizado

Los sistemas de renderizado son aquellos que, a través de la información de las capas de un SIG, son capaces de mostrarnos esta información generando una imagen. Un ejemplo es Open Street Maps, que genera sus mapas a través de diferentes técnicas de renderizado , que podemos seleccionar desde su aplicación web. El funcionamiento de estos sistemas en apariencia es parecido, se le ofrecen una serie de capas de información, se define un tamaño de la vista, se define una escala de la vista, y se genera la imagen. Pero cada una de ellas acepta tipos de datos diferentes, genera ficheros diferentes y es accesible en plataformas diferentes.

d. Osmarender

Es un sistema basado en reglas para generar ficheros SVG (ficheros de gráficos vectoriales que interpretan los navegadores). Realmente no

renderiza una imagen, sino que aplica una transformación del formato de un fichero a SVG. Está escrito en JavaScript y Python, y corre sobre un servidor web. Osmarender es usado por Open Street Maps, por lo que se puede convertir de .osm a SVG.

e. Mapnik

Es un programa gratuito escrito en C++ y Phyton que permite generar mapas a través de información de capas, de las cuales soporta los formatos ESRI shapefiles, PostGIS, TIFF rasters, .osm, GDAL o OGR, gracias a convertir cualquier formato a PostGIS (su nativo), a través de la aplicación osm2pgsql. Es el sistema principal que usa Open Street Maps para renderizar sus mapas a través de su servidor "tile.openstreetmap.org".

1.3.3. GEOLOCALIZACIÓN

1.3.3.1. Definición

El termino geolocalización hace referencia a la identificación de la posición geográfica en tiempo real de un objeto o persona, ya sea un dispositivo conectado a Internet, un teléfono móvil o cualquier otro aparato que sea posible rastrear (Bover, 2010). Para determinar la ubicación, existen varias maneras de hacerlo, entre ellas están la identificación del router al que se te encuentra conectado, la red del proveedor, celular o directamente por el receptor interno de GPS del dispositivo.

En los últimos años, diferentes tipos de tecnologías han apostado por la geolocalización, siendo extraordinario el auge de esta en las tecnologías móviles de última generación.

1.3.3.2. Sistemas de localización para dispositivos móviles

Existen diferentes maneras de localizar un dispositivo móvil, pero la efectividad del método dependerá de algunas variables como el medio o la disponibilidad de esta medición en el terminal.

Es posible clasificar los diferentes sistemas en tres grandes grupos:

a. Basados en la red: Estos sistemas utilizan los sistemas del proveedor de servicios para determinar la posición del terminal, por lo que no necesitamos ninguna aplicación específica funcionando en el móvil. El problema principal de este sistema es que es preciso estar cerca del proveedor para que funcione.

b. Basados en el terminal: Los dispositivos que utilizan estos sistemas disponen de un receptor de señales y un software cliente para determinar la posición del terminal a través de las señales externas. Cabe destacar que es preciso instalar una aplicación en el móvil, haciendo que el funcionamiento de esta dependa de la adaptación de los diferentes sistemas operativos.

c. Híbridos: Los sistemas híbridos son una combinación de sistemas basados en el terminal y sistemas basados en la red; aunque contenga los métodos más fiables, también adquiere los problemas de los dos grupos anteriores.

1.4. ANDROID

1.4.1. Definición

Android es un software para dispositivos móviles que incluye un sistema operativo, y una pila de software como aplicaciones, framework y middleware, todos juntos forman el sistema completo (Burnette Ed, 2009). El SDK de Android proporciona las herramientas y APIs necesarios para comenzar el desarrollo de aplicaciones en la plataforma Android usando el lenguaje de programación Java.

1.4.2. Características

Android se caracteriza por cubrir muchas aéreas, como el desarrollo de aplicaciones basada en sensores, conectividad y medios de comunicación (Android Developers, 2012). Es por tal motivo que seguidamente se mencionan las características más importantes de la plataforma:

- Application Framework para reutilizar y sustituir componentes.
- Maquina Virtual Dalvik optimizada para dispositivos móviles.

- SQLite para almacenamiento de datos estructurados.
- Sensores de cámara, gps, redes 3G, wifi.
- Emulador de dispositivos.
- Conectividad Bluetooth, WIFI, WIMAX.
- Mensajería SMS, MMS.

1.4.3. Arquitectura

Android maneja una arquitectura distribuida en cinco capas (ver fig. 3), mencionadas a continuación:

1. Capa de Aplicaciones

Es la capa más alta en la arquitectura de Android, y básicamente se refiere a todas las aplicaciones con las que interactúa el usuario sin conocer todas las acciones que ocurren por debajo (Android Developers, 2012).

Entre las aplicaciones que vienen por defecto tenemos: Navegador-Buscador, Contactos, Android Market, Servicio de llamadas y mensajes.

2. Capa de Frameworks

Esta capa permite a los desarrolladores tener acceso al conjunto de APIs utilizadas por las aplicaciones principales. La arquitectura está diseñada para simplificar la reutilización de componentes, donde cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación podrá entonces hacer uso de esas capacidades (sujeto a restricciones de seguridad impuestas por la plataforma). Este mismo mecanismo permite que los componentes sean reemplazados por el usuario (Android Developers, 2012).

3. Capa de Librerías

Se incluye un conjunto de bibliotecas C/C++ utilizado por diversos componentes del sistema. Estas capacidades están expuestas a los desarrolladores a través de su estructura de aplicaciones (Android Developers, 2012). Entre las bibliotecas más conocidas tenemos: System C library, Surface Manager, SGL, SQLite, etc.

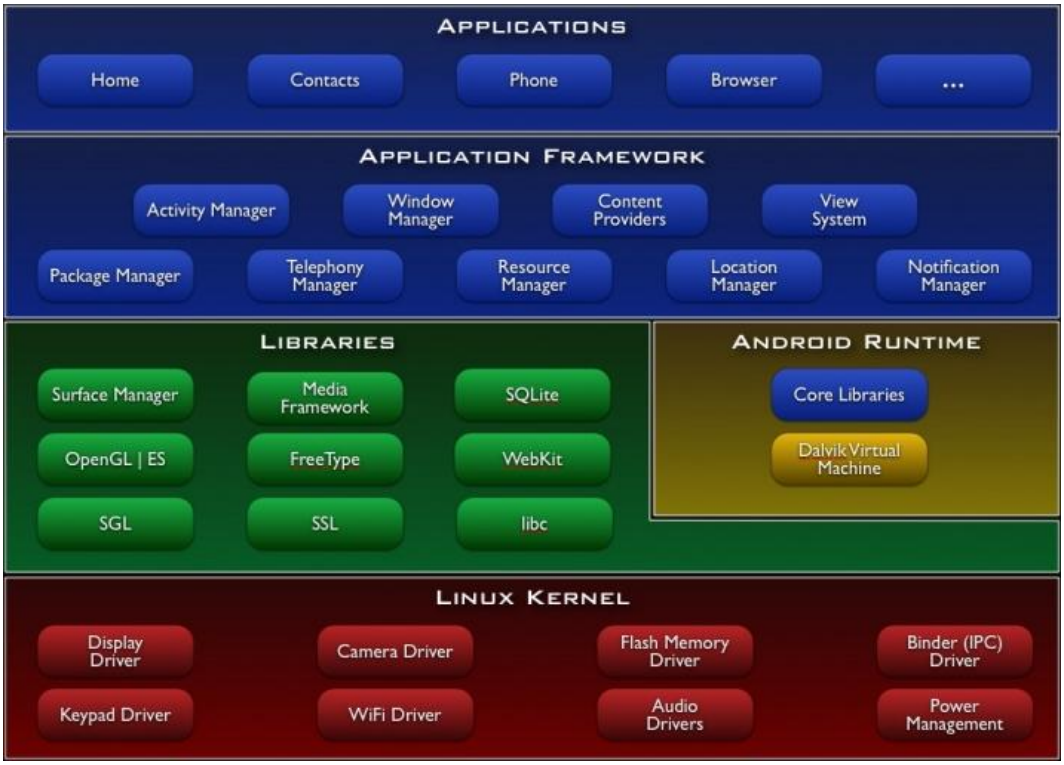

Fig. 3. Arquitectura del Sistema Operativo Android

4. Android Runtime

Android incluye un conjunto de bibliotecas del núcleo que proporciona la mayor parte de la funcionalidad disponible en las bibliotecas del núcleo del lenguaje de programación Java. En Android cada aplicación se ejecuta en su propio proceso, con su propia instancia de la maquina virtual que ha sido diseñada para que el dispositivo pueda ejecutar múltiples maquinas virtuales de manera eficiente.

5. Kernel de Linux

Android se basa en la versión 2.6 de Linux para el sistema de servicios básicos, tales como la seguridad, la gestión de memoria, gestión de procesos, pila de red, y el modelo del controlador. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila de software.

1.4.4. Versiones

Android inicio a partir del año 2007, desde aquel entonces han existido muchas versiones (Android Developers, 2012), sin embargo a partir de la versión 1.5 cada una recibía un nombre peculiar, además de que cada versión era mejorada.

A continuación se ha hecho un listado de las características más importantes de cada versión[5], con respecto al enfoque del tema de investigación y funcionalidades relevantes entre versiones:

Android 1.5 (Cupcake, abril 2009)

- Rediseño completo de todos los elementos de la interfaz
- Mejoras en velocidad de cámara.
- Menor tiempo de búsqueda de satélites GPS.
- Interprete Javascript.
- Inclusión de teclado en pantalla.

Android 1.6 (Donut, Septiembre 2009)

- Conexión de redes VPN.
- Nueva pantalla para el control de batería.
- Motor de texto a voz.

Android 2.0 / 2.1 (Eclair, Octubre 2009)

- Rediseño de la Interfaz del Navegador.
- Soporte Flash de la Cámara.
- Bluetooth 2.1
- Mejoras Google maps.
- Mejoras Teclado Virtual.
- Mejoras duración batería.

[5] Android version history: http://en.wikipedia.org/wiki/Android_version_history

Android 2.2 (Froyo, Mayo 2010)

- Actualizaciones automáticas para aplicaciones.
- Soporte Wifi IEEE 802.11n.
- Soporte API Gráfica OpenGL.
- Compilador JIT (Just in time).
- Cloud to Device API.
- Android Market.

Android 2.3 (Gingerbread, Diciembre 2010)

- Refinamientos de Interfaz de Usuario.
- Teclado más suave y multi-táctil.
- Administrador de Aplicaciones.
- Soporte nativo para telefonía VoIP SIP (Llamadas por Internet).
- Soporte para reproducción de videos web.
- Administrador de descargas.

Android 3.0 - 3.1 - 3.2 (Honeycomb, 2011)

- Soporte para Tablets.
- Escritorio 3D con Widgets Rediseñados.
- Sistema mejorado Multitarea.
- Mejoras soporte redes Wifi.
- Soporte para periféricos de salida.

Android 4.0 (IceCream Sandwich, 2011)

- Soporte para Lapiz Táctil Stylus.
- Reconocimiento Facial.
- Corrector de Texto mejorado.
- Reconocimiento de voz de usuario.
- Android Beam para compartir contenido entre teléfonos.
- Captura de pantalla.
- Creación mejorada de directorios.

1.4.4.1. Distribución de las versiones

La página de desarrolladores de Android cada 15 días emite un informe porcentual y gráfico referente a los productos activos que ejecutan una determinada versión de la plataforma Android (Android Developers, 2012). Lo que permite entender el panorama de la distribución del dispositivo y decidir la forma de priorizar el desarrollo de las funciones de su aplicación en relación a las versiones de SO más popular.

La figura 4 representa la proporción de dispositivos que han ingresado a Google Play en el mes de febrero 2012. Donde podemos observar que la versión que más accesos ha tenido es la 2.3.3 (Gingerbread), seguida por la popular 2.2 (Froyo). En la tabla 1 se observa en mayor detalle los porcentajes correspondientes a cada versión.

Plataforma	Nivel API	Distribución
Android 1.5	3	0.6%
Android 1.6	4	1.0%
Android 2.1	7	7.6%
Android 2.2	8	27.8%
Android 2.3 - 2.3.2	9	0.5%
Android 2.3.3 - 2.3.7	10	58.1%
Android 3.0	11	0.1%
Android 3.1	12	1.4%
Android 3.2	13	1.9%
Android 4.0 - 4.0.2	14	0.3%
Android 4.0.3	15	0.7%

Tabla 1. Distribución de las versiones Android (Febrero 2012)

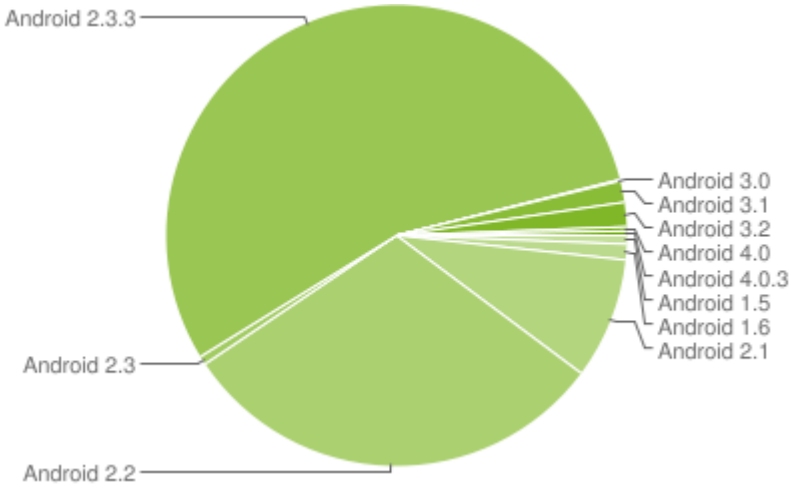

Fig. 4. Representación gráfica – Accesos Android Market por versiones

1.4.5. Herramientas de Desarrollo

1.4.5.1. SDK de ANDROID

El SDK de Android para desarrolladores provee un conjunto de herramientas que facilita el desarrollo de aplicaciones para múltiples clientes. Entre estos servicios encontramos el emulador de Teléfono Android (AVD) que permite la ejecución de nuestras pequeñas aplicaciones en un entorno que simula el teléfono real. No obstante existen ciertas limitantes, por tal motivo algunas pruebas como la cámara o GPS se realizaron en un equipo con el SO, ya que nos permite tener una visión real de la funcionalidad de nuestra aplicación.

1.4.5.1.1. Android Virtual Device

Este dispositivo virtual se utiliza para simular las compilaciones de software y las especificaciones de hardware disponibles en diferentes dispositivos. Esto permite probar la aplicación en una variedad de plataformas de hardware sin la necesidad de adquirir una variedad de teléfonos. Cada dispositivo virtual está configurado con un nombre, un destino de generación de Android (basado en la versión SDK compatible), con una capacidad de tarjeta SD, y la resolución de la pantalla.

1.4.5.1.2. Android Emulator

El Emulador es una de las herramientas que ayuda en gran medida al desarrollador para ejecutar y depurar las aplicaciones. Una implementación de la máquina virtual Dalvik, por lo que es válido como una plataforma para correr aplicaciones Android, como si se lo realizara en un teléfono Android real.

Debido a que es disociado de cualquier hardware en particular, es una base excelente para usar y probar las aplicaciones. Proporciona plena conectividad de red, junto con la habilidad de ajustar la velocidad de conexión a Internet y la latencia durante la depuración de sus aplicaciones. También se puede simular realizando y recibiendo llamadas de voz y mensajes SMS.

El ADTplug-in se integra al emulador de Eclipse para que sea enganchado de forma automática dentro de la AVD seleccionando ejecutar o depurar sus proyectos.

Para ejecutar el emulador previamente se debe crear un dispositivo virtual. El emulador iniciará el dispositivo virtual y ejecutará una instancia de Dalvik dentro de ella.

1.4.6. Comparación entre Plataformas

En la web podemos encontrar mucho material acerca de comparaciones técnicas entre los SO más utilizados para dispositivos móviles: iOS, Android y Windows Mobile. En la tabla 2 se pueden observar las características más relevantes en relación al enfoque de estudio.

Según estadísticas referidas a la penetración de los smartphones en el mercado de la telefonía móvil (Comscore, 2011), se pudo apreciar que este tipo de dispositivos ocupa la minoría en el mercado con tan sólo un 27%, y las principales plataformas se resumen en: Android, iOS, Blackberry, Symbian, Windows Phone, etc.

Sin embargo, se afirma al SO Android como la plataforma más usada en telefonía inteligente, con un 11% de teléfonos en todo el mundo; iOS se queda con sólo un 5%, al nivel de un Symbian que continúa decayendo progresivamente.

Funciones	iOS 5	Android 4.0	Windows Mobile 7.5
Núcleo (Kernel)	OS X	Linux	Windows CE 7
Estándares de soporte	GSM, CDMA	GSM, CDMA	GSM, CDMA
Hardware de soporte	Sólo iPhone, iPad y iPod touch	Una amplia variedad de dispositivos	Una limitada variedad de dispositivos
Seguridad	Sí	Susceptible a Malwares	Sí
Integración con redes sociales	Sólo Twitter	Facebook, Twitter	Facebook, Twitter, LinkedIn, Windows Live
Navegador	Safari Móvil	Basado en Chrome	Internet Explorer 9
Soporte para Flash	No	Sí, lo tiene	No
Motor de búsqueda predeterminado	Google	Google	Bing
Sincronización con WiFi	Sí	Sólo con aplicaciones de terceros	Sí
Soporte con tablets	Sí	Sí	No
Actualizaciones	Sí	Sí	Sí
Soporte con La Nube	iCloud	Google Sync	SkyDrive
Personalización	Limitado (con jailbreaks)	Profunda personalización	Ninguna
Aplicaciones	Más de 500,000	Más de 450,000	Más de 30,000
Posibilidad de realizar una captura de pantalla	Sí	Sí	No
Librería	iBooks	Google Books	No tiene
Respaldo inalámbrico en Línea	Sí (5 GB gratis)	No	No

Tabla 2. Cuadro comparativo entre los SO: Android, iOS y Windows Mobile. [6]

La mayor ventaja en cuanto a la aparición del Sistema Operativo Android se la llevan los desarrolladores, ya que al no existir las restricciones impuestas por los sistemas operativos propietarios sobre el desarrollo de aplicaciones por terceros en Android, son libres de escribir aplicaciones que aprovechen al máximo el hardware de estos dispositivos para distribuirlas en un mercado abierto.

1.4.7. Programación en android

En este apartado se detallan algunos temas relevantes en cuanto a la programación de la aplicación de realidad aumentada y geolocalización utilizando el SDK de Android, herramientas como el IDE de Eclipse y el lenguaje de programación Java

[6] Comparativa: iOS 5 vs. Android 4.0 Ice Cream Sandwich vs. Windows Phone 7.5 Mango: http://appleDiario.com/2011/10/25/comparativa-ios-5-vs-android-4-0-ice-cream-sandwich-vs-windows-phone-7-5-mango/

para crear aplicaciones sencillas que puedan introducir al lector en el mundo del desarrollo de aplicaciones para Android.

Todo lo que se necesita para empezar a escribir aplicaciones para Android es una copia del SDK de Android y el kit de desarrollo Java y el Eclipse IDE el cual es recomendado por Android, sin embargo se puede hacer en cualquier otro IDE de Java.

Es recomendable tener conocimientos o experiencia en el desarrollo con Java, ya que las técnicas, la sintaxis y la gramática se traducirán directamente en Android, aunque algunas de las técnicas de optimización pueden parecer contradictorias.

Para el desarrollo de la aplicación móvil se utilizaron las siguientes herramientas y APIs:

1. Eclipse IDE for Java Developers [7]

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Eclipse dispone de un Editor de texto con resaltado de sintaxis. La compilación es en tiempo real. Tiene pruebas unitarias, depurador, asistentes (wizards) para creación de proyectos, clases, tests, etc., y refactorización.

2. Wikitude API 2.3 [8]

Wikitude es una plataforma que acoge a miles de proveedores de contenido cada uno ofrece un servicio diferente. El API fue desarrollado por la empresa austriaca Mobilizy creadora de la Aplicación Wikitude. El API de Wikitude permite crear aplicaciones con Realidad Aumentada y ser adapta a sus necesidades. Para el uso del API se requiere que la aplicación Wikitude oficial esté instalada en el dispositivo. En caso de no ser así, se presenta un mensaje que advierte de este hecho, y permite ir abrir el Android Market para poder instalarla.

[7] Eclipse Software: http://es.wikipedia.org/wiki/Eclipse_(software)

[8] Wikitude: http://www.wikitude.com/tour/wikitude-world-browser

3. GoogleDirections API[9]

Es un servicio que utiliza una solicitud HTTP para calcular rutas para llegar de una ubicación a otra. Las rutas pueden especificar los orígenes, los destinos y los hitos como cadenas de texto (por ejemplo, "Chicago, IL" o "Darwin, NT, Australia") o como coordenadas de latitud/longitud. El API puede devolver rutas segmentadas mediante una serie de hitos. Por lo general, este servicio está diseñado para calcular rutas a partir de direcciones estáticas (conocidas previamente) para la ubicación del contenido de la aplicación en un mapa. Sin embargo, este servicio no está diseñado para responder en tiempo real a la información introducida por el usuario, por ejemplo. Para calcular rutas dinámicas (por ejemplo, en un elemento de interfaz de usuario), consulta la documentación sobre el servicio de rutas de la versión 3 de JavaScript API.

4. Netbeans IDE [10]

Es un entorno de desarrollo integrado, una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas. Está escrito en Java - pero puede servir para cualquier otro lenguaje de programación. Existe además un número importante de módulos para extender el IDE NetBeans. El IDE NetBeans es un producto libre y gratuito sin restricciones de uso. El NetBeans IDE soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Entre sus características se encuentra un sistema de proyectos basado en Ant, control de versiones y refactoring.

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

[9] Google Directions API: https://developers.google.com/maps/documentation/directions/?hl=es

[10] Adaptado de Netbeans IDE - Source Code Editor: http://netbeans.org/features/ide/editor.html

5. Glassfish[11]

Es un servidor de aplicaciones de software libre desarrollado por Sun Microsystems, compañía adquirida por Oracle Corporation, que implementa las tecnologías definidas en la plataforma Java EE y permite ejecutar aplicaciones que siguen esta especificación. Es gratuito y de código libre, se distribuye bajo un licenciamiento dual a través de la licencia CDDL y la GNU GPL.

6. MySQL[12]

MySQL es un sistema de gestión de bases de datos relacional multiusuario, con más de seis millones de instalaciones. MySQL es un software libre basado en un esquema de licenciamiento dual. Herramienta muy utilizada en aplicaciones web, múltiples plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL.

1.4.7.1. Tipos de aplicaciones

Las aplicaciones complejas son difíciles de encasillar en una sola categoría, por lo general incluyen elementos de cada uno de estos tipos. Al crear una aplicación se debe tener en cuenta, cómo es probable que la utilicen y luego diseñarla.La mayoría de las aplicaciones que se creen en Android se ubicará en una de las siguientes categorías (Android Developers 2012):

Foreground Applications (De Primer Plano): Estas aplicaciones solo se utilizan cuando el usuario las ve y las usa. En cuanto la minimiza o cierra, deja de actuar. Los juegos son un ejemplo típico de estas aplicaciones. Al crear aplicaciones de primer plano se necesita considerar cuidadosamente el ciclo de vida de actividad, para que la actividad pueda cambiar a la perfección entre el primer plano y el trasfondo.

[11] Glassfish (Wikipedia): http://es.wikipedia.org/wiki/GlassFish

[12] MySQL (Wikipedia): http://es.wikipedia.org/wiki/MySQL

Background Servicese Intent Receivers (De fondo): Estas aplicaciones tienen una interacción limitada con el usuario, pasan la mayoría de su vida escondidas, esperando algún evento. Aplicaciones para responder SMS o interactuar con llamadas son ejemplos. Es posible crear servicios totalmente invisibles, pero en la práctica es mejor proporcionar al menos algún tipo de control de usuario. Como mínimo se debe permitir a los usuarios confirmar que el servicio está en funcionamiento y se debe dejar de configurar, poner en pausa, o darlo por terminado, según sea necesario.

Intermittent Applications (Intermitentes): Son aplicaciones que son interactivas, pero hacen mucho de su trabajo de fondo, notificando a los usuarios cuando es apropiado. Servicios de Chat y Música son ejemplos.

Estas aplicaciones son por lo general de un conjunto de Actividades y Servicios

Widgets: Son aplicaciones que viven solamente en el home-screen del dispositivo. En algunos casos, su aplicación puede consistir enteramente de un componente flash. El uso de Widgets, puede crear componentes visuales interactivos que los usuarios pueden añadir a las pantallas de inicio. Las aplicaciones Widgets sólo se utilizan para mostrar información dinámica, como los niveles de batería, las previsiones meteorológicas o la fecha y hora.

Al crear una aplicación, podemos utilizar todas las técnicas mencionadas anteriormente, pero es importante considerar como la utilizará el usuario y diseñar la interfaz apropiadamente.

1.4.7.2. Componentes de una aplicación Android

El componente de la actividad posiblemente es el más importante de los bloques de construcción para Android, la clase actividad, es la base de todas sus pantallas de interfaz de usuario. Las aplicaciones para Android se componen de elementos de acoplamiento flexible, vinculado con un manifiesto de proyecto que se describe a continuación, cada componente y cómo interactúan.

Existen componentes que proporcionan los recursos necesarios para nuestras aplicaciones, entre ellos se mencionan los más conocidos y utilizados en la aplicación:

- Activities (Actividades): Se encuentra en la capa de presentación de una aplicación. Cada pantalla de la aplicación será una extensión de la clase Actividad. Las Actividades usan vistas para formar interfaces de usuario gráficas que muestran la información y respuesta a las acciones del usuario (Android Developers, 2012).

- Intents: Puede transmitir mensajes a todo el sistema o en una actividad de destino o servicio, indicando su intención de tener una acción realizada (Android Developers, 2012).Entonces, el sistema determinará el destino que llevará a cabo cualquier acción en su caso.

- Notificaciones: Permite la señal de los usuarios sin robar el enfoque o interrupción de las actividades actuales. Son las técnicas requeridas para conseguir la atención de los usuarios dentro de un servicio de receptor o difusión. Por ejemplo cuando el dispositivo recibe un mensaje de texto o llamada entrante, esta actividad las recibe por medio de la emisión de luces, haciendo sonidos, enviando iconos, o mostrando mensajes de dialogo. Se puede desencadenar este mismo hecho o evento desde tu propia aplicación usando notificaciones (Android Developers, 2012).

1.4.7.3. Interfaces de usuario

Las interfaces en Android se pueden construir de dos maneras, mediante código en formato XML o código Java. La definición de la estructura GUI en XML es muy preferible, ya que la interfaz de usuario siempre debe ser separada de la lógica del programa. Además la adaptación de un programa de resolución de pantalla de uno a otro es mucho más fácil. La definición de una interfaz de usuario en XML es muy similar a crear un documento HTML común. Lo

mismo que en Android XML-Diseños (Android Developers, 2012). Todo está bien estructurado y puede ser expresada por estructuras arbóreas:

Jerarquía de los Elementos de la pantalla

La unidad básica funcional de una aplicación Android es la activity, una actividad que puede hacer muchas cosas. Para iniciar la actividad en pantalla se trabaja con View y Viewgroups que son unidades básicas de la expresión de la interfaz de usuario en la plataforma Android.

View: Es una estructura de datos cuyas propiedades almacenan el diseño y el contenido de un área específica rectangular de la pantalla. Un objeto de vista se encarga de la medida, su diseño, dibujo, el desplazamiento y gestos para el área de la pantalla que representa

Viewgroup: Es un tipo especial de objeto de la vista, cuya función es contener y gestionar un conjunto de puntos de vista y subordinados. Un Viewgroup permite agregar a la estructura de la interfaz de usuario, la acumulación de elementos complejos de pantalla que puede ser abordado como una sola entidad.

A Tree Structured UI: En la plataforma Android, se define la interfaz de usuario de una actividad con un árbol de vista y nodos Viewgroup, como se muestra en la figura 5. El árbol puede ser tan simple o tan complejo, dependiendo que se necesite para hacerlo.

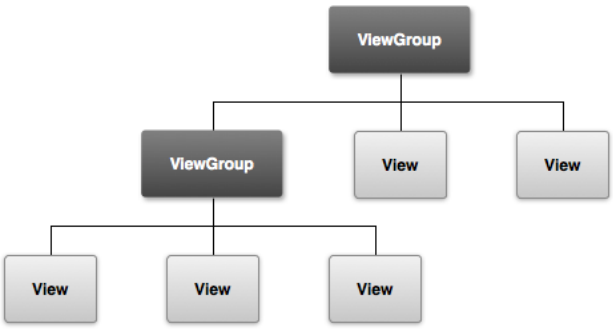

Fig. 5. Jerarquía de viewgroup

1.4.7.4. Desarrollo de aplicaciones

En el transcurso del presente tema de investigación se creyó conveniente realizar un conjunto de pruebas de concepto relacionadas al enfoque del proyecto; con la finalidad de lograr una mayor familiarización y adaptación al entorno de trabajo de esta plataforma.

En el **Anexo A**, se presenta la estructura de una aplicación básica, se detalla el conjunto de archivos, ficheros y componentes necesarios para el correcto funcionamiento de la misma.

En el **Anexo B**, se detalla el desarrollo de un conjunto de aplicaciones referentes a temas de: interfaz gráfica, localización basada en gps, mapas, overlays y el consumo de webservices.

El desarrollo de cada una de estas aplicaciones se realizó siguiendo la Guía para Desarrolladores Android (Android Developers, 2012) y un blog sobre aplicaciones Android.

1.5. SERVICIOS WEB

1.5.1. Definición

Un servicio web es un componente de software que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos. Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web. Para mejorar la interoperabilidad entre distintas implementaciones de servicios Web se ha creado el organismo WS-I, encargado de desarrollar diversos perfiles para definir de manera más exhaustiva estos estándares. (Papazoglou M.P , Georgakopoulos D, 2003)

Existen dos filosofías a la hora de escribir servicios web (Motta David, 2011), cada uno con sus características:

REST: utiliza XML, JSON y HTTP. Cada URL representa un objeto sobre el que puedes realizar POST, GET, PUT y DELETE (las operaciones típicas del HTTP). Este tipo de servicio es ligero ya que utiliza objetos JSON o XML, presenta resultados legibles y es fácil de implementar ya que no hacen falta herramientas específicas.

SOAP: es toda una infraestructura basada en XML. Donde cada objeto puede tener métodos definidos por el programador con los parámetros que sean necesarios. Este tipo de servicio es fácil de consumir, es rígido ya que contiene un tipado fuerte, sigue un contrato y existen muchas herramientas de desarrollo para estos servicios.

Para el presente proyecto se implementará un servidor web REST, que permita consumir sus servicios desde un dispositivo Android, es por eso que a continuación se mencionan algunos de los conceptos y hechos más importantes.

1.5.2. Servicios REST

Este tipo de servicio ha obtenido mayor adopción en la web como una alternativa más sencilla a SOAP y a los servicios web basados en el Lenguage de Descripción de Servicios Web (Web Services Descripcion Language - WSDL). Es por eso que hoy en día, grandes proveedores de Web 2.0 están migrando a esta tecnología orientada a los recursos, incluyendo a Twitter, Google y Facebook, quienes marcaron como obsoletos a sus servicios SOAP y WSDL.

Una implementación concreta de un servicio web REST sigue cuatro principios de diseño (Rodríguez Alex, 2008) fundamentales:

- Utiliza los métodos HTTP de manera explícita
- No mantiene estado
- Expone URIs con forma de directorios
- Transfiere XML, JavaScript Object Notation (JSON), o ambos

1.5.2.1. Características

Una de las características claves de los servicios web REST es el uso explícito de los métodos HTTP; así resulta consistente con la definición del protocolo. Este principio de diseño básico establece una asociación uno-a-uno entre las

operaciones de crear, leer, actualizar y borrar y los métodos HTTP. De acuerdo a esta asociación:

POST: se utiliza para crear un recurso en el servidor.

GET: se utiliza para obtener un recurso.

PUT : se utiliza para cambiar el estado de un recurso o actualizarlo

DELETE: se usa para eliminar un recurso.

Los servicios web REST necesitan escalar para poder satisfacer una demanda en constante crecimiento, es por ello que se usan clústeres de servidores con balanceadores de carga y alta disponibilidad, que trabajan en función de transferir peticiones de un equipo a otro para disminuyendo el tiempo total de respuesta de una invocación al servicio web.

Una característica de los servicios web REST es que las URIs a utilizar deben ser sencillas a tal punto de que sea fácil adivinarlas; es decir que necesite de muy poca o ninguna explicación o referencia para que un desarrollador pueda comprender a lo que apunta, y a los recursos derivados relacionados.

Algunos de los tipos MIME más usados para los servicios web REST son:

MIME-Type	Content-Type
JSON	application/json
XML	application/xml
XHTML	application/xhtml+xml

Así el servicio puede ser utilizado por diferentes clientes y multiples lenguajes, corriendo en diversas plataformas y dispositivos.

REST no mantiene estado, los servicios sin estado son mucho más simples de diseñar, escribir y distribuir a través de múltiples servidores. Un servicio sin estado no sólo funciona mejor, sino que además mueve la responsabilidad de mantener el estado al cliente de la aplicación. En un servicio web REST, el

servidor es responsable de generar las respuestas y proveer una interfaz que le permita al cliente mantener el estado de la aplicación por su cuenta. (Rodriguez Alex, 2008)

1.5.4. Consumo de Servicios REST desde Android

Android no puede consumir directamente un Servicio REST, sin embargo posee algunas librerías que permiten realizar conexiones HTTP, y como este tipo de Servicios trabajan en función de este protocolo, es posible consumir los servicios a través de un cliente HTTP. Los clientes HTTP encapsulan una mezcla de distintos objetos necesarios para ejecutar las peticiones HTTP. Al final de cuentas será tan fácil como crear un objeto Document en función de un Recurso de la Web. (Android Developers, 2012)

A su vez existen varios frameworks REST para JAVA y Android, entre los que se destacan:

Restlet (Android, Java).

Spring Android (Android, Java): Spring Android Rest Template es un robusto y popular cliente REST basado en Java, y Spring Android provee una versión RestTemplate que trabaja en entornos Android.

JBOSS RESTEasy (Java) [13]

Es un proyecto que proporciona diversos marcos para ayudar la implementación de REST Web Services y aplicaciones Java REST. Una aplicación totalmente certificada y portátil de la especificación JAX-RS. Proporciona una API de Java para REST Web Services a través del protocolo HTTP.

Apache CFX (Java) [14]

[13] Adaptado de RESTEasy - JBoss Community. Disponible en: http://www.jboss.org/resteasy

[14] Adaptado de Apache CXF: An Open-Source Services Framework. Disponible en: http://cxf.apache.org/

Es un framework open source de serviciosCXF le ayuda a construir y desarrollar servicios que utilicen las API de interfaz de programación, como JAX-WS y JAX-RS. Estos servicios pueden hablar una variedad de protocolos como SOAP, XML / HTTP, REST HTTP, o CORBA y el trabajo sobre una variedad de medios de transporte tales como HTTP, JMS o JBI.

CAPITULO II

2. PROBLEMÁTICA

2.1. Descripción

La Universidad Técnica Particular de Loja cada ciclo académico acoge gran cantidad de visitantes, los mismos que en su mayoría no conocen las instalaciones del campus universitario, muchas veces necesitan trasladarse de un punto especifico a otro, más aun cuando existen eventos, congresos, talleres, seminarios. Muchos estudiantes requieren conocer que paradas de bus UTPL se encuentran cerca del sector. Estudiantes de la modalidad abierta necesitan consultar información sobre un centro universitario. Además que muchos turistas necesitan conocer los sitios de interés en la ciudad. Sin embargo no existe una herramienta tecnológica que permita cubrir las necesidades mencionadas de manera fácil, rápida y accesible.

Es por tal motivo que nace la necesidad de crear una aplicación para dispositivos móviles, que facilite esta información a estudiantes y visitantes de una manera rápida y oportuna.

2.2. Justificación

Hoy en día el uso de teléfonos inteligentes ha tenido un enorme crecimiento dentro del mundo actual, esto se debe a las grandes prestaciones que brindan los mismos, un claro ejemplo es la conectividad, y si a ello se suma las últimas tecnologías como la geolocalización y realidad aumentada, se pueden desarrollar excelentes aplicaciones: livianas, rápidas y potentes. Además de lo mencionado existen grandes repositorios y bases de conocimiento que contienen información georeferenciada mismos que permiten explotar aun más el servicio de este tipo de aplicaciones.

2.3. Objetivos

2.3.1. General

Desarrollar una aplicación orientada a principios de Realidad Aumentada en base a la Infraestructura del Campus UTPL y servicios que se ofrecen, para clientes móviles con SO Android y consumo de Servicios REST.

2.3.2. Específicos

- Mostrar una vista con realidad aumentada sobres los puntos de interés del campus UTPL.

- Mostrar a través de mapas la ubicación actual del usuario así como los puntos de interés UTPL, centros universitarios, paradas de bus, y sitios de interés de la ciudad.

- Crear un servicio web que permita acceder a los datos del sistema desde el dispositivo móvil.

- Consultar datos de la DBpedia para los puntos de interés que contengan información.

- Crear un sitio de administración de datos de la aplicación.

CAPITULO III

3. DISEÑO Y DESARROLLO DE LA APLICACIÓN

En este capítulo se detallan los procesos, fases, herramientas y APIs utilizadas en el desarrollo de la aplicación propuesta. Dichos conceptos abarcan bases teóricas que han facilitado la construcción de la aplicación.

3.1.VISIONAMIENTO

La Universidad Técnica Particular de Loja como una institución de educación superior del Ecuador, cada ciclo académico recibe la visita de estudiantes, empresarios y turistas. Los mismos que en muchos casos necesitan trasladarse de un punto especifico a otro; ya sea para eventos como: seminarios, congresos, grados, reuniones, presentaciones, etc. Este tipo de información se puede solicitar en un lugar de información; sin embargo no existe algún medio o herramienta tecnológica que facilite este contenido necesario.

Por tal razón se ha creído conveniente el desarrollo de una aplicación de Realidad Aumentada y Geolocalización que permita recorrer las instalaciones del campus UTPL , conocer los sitos más importantes de la ciudad de Loja, consultar información sobre el servicio de bus y centros universitarios; todo esto a través de un dispositivo móvil con sistema operativo Android. A fin de facilitar este contenido informativo de manera oportuna, agilizar procesos, lo que brindaría una mejor visita.

3.2. METODOLOGÍA

La metodología utilizada en todo el proceso de desarrollo del sistema es ICONIX, misma que maneja un conjunto de artefactos, actividades y procesos requeridos en el desarrollo de aplicaciones que manejan tiempos reducidos y pocos recursos. Sus características principales son: Iterativo-incremental, permite trazabilidad, y maneja una UML dinámica.

Para llevar a cabo las fases de desarrollo se han empleado distintos artefactos, entre ellos especificación de requerimientos, diagrama y especificación de casos de uso, arquitectura de la aplicación, diagramas de clases y otros.

3.2.1. Etapas de desarrollo

Como se menciona anteriormente, ICONIX maneja una serie de fases continuas, es por ello que a continuación se ha elaborado una tabla correspondiente a la fase, tareas y los artefactos a utilizar en cada una:

Fase	Tareas	Artefactos
Análisis de Requisitos	a) Establecer los requisitos funcionales y no funcionales del sistema. b) Identificar los objetivos y necesidades del sistema. c) Identificar los casos de uso del sistema mostrando los actores involucrados.	• Especificación de Requerimientos. • Modelo de Dominio • Modelo de casos de uso
Análisis y Diseño Preliminar	a) Terminar el modelo estático, adicionando los detalles del diseño en el diagrama de clases. b) Verificar si el diseño satisface todos los requisitos identificados.	• Especificación de Casos de Uso
Diseño	a) Establecer la arquitectura del sistema. b) Identificar los componentes del sistema. c) Especificar el comportamiento a través del diagrama de secuencia. d) Finalizar el diagrama de clases	• Arquitectura del sistema • Diagramas de Secuencia • Diagrama de Clase
Implementación	a) Escribir código del sistema. B) Registrar un historial de prototipos del sistema. b) Realizar casos de prueba.	• Código generado, • Historial de prototipos • Caso de pruebas

Tabla 3. Fases de desarrollo del Buscador OCW

3.3. DESARROLLO

3.3.1. Análisis de Requisitos

3.3.1.1. Especificación de Requerimientos

Para una mayor especificación de la Aplicación se han descrito los requisitos funcionales y no funcionales del sistema UTPLAR (Aplicación Móvil y Servidor).

Los requerimientos funcionales definen el comportamiento interno del sistema: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran como los casos de uso serán llevados a la práctica. Los requerimientos no funcionales por su parte especifican criterios que pueden usarse para juzgar la calidad de operación de un sistema y no sólo sus comportamientos específicos.

Cód.	Requisito	Descripción	Prioridad
RF01	Cargar Vista RA	El cliente debe mostrar la vista de realidad aumentada sobre la captura de video en la cámara del dispositivo móvil.	Alta
RF02	Cargar Mapas	El cliente debe mostrar el mapa en pantalla provisto por Google Maps.	Alta
RF03	Mostrar detalles	El sistema debe mostrar el detalle de un POI seleccionado por el usuario tanto en RA como en Mapas.	Alta
RF04	Obtener Ubicación Actual	El cliente deberá automáticamente determinar la posición del usuario, siempre y cuando exista señal de GPS e Internet.	Media
RF05	WebService UTPLAR	El cliente debe recuperar datos que el Webservice le proporciona.	Alta

RF06	Consultas DbPedia	Consultas a DBpedia para recursos existentes.	Media
RF07	Autenticación UTPLAR Web	Autentificarse para utilizar la administración web UTPLAR.	Media
RF08	Agregar Nuevos POIs	Funcionalidad que permite insertar nuevos puntos de interés: sitios, centros universitarios, lugares.	Alta
RF09	Actualizar POIs	Funcionalidad que permite actualizar puntos de interés existentes.	Media
RF10	Eliminar POIs	Funcionalidad que permite eliminar un poi.	Baja

Tabla 4. Requerimientos Funcionales

3.3.1.2. Modelo de Dominio

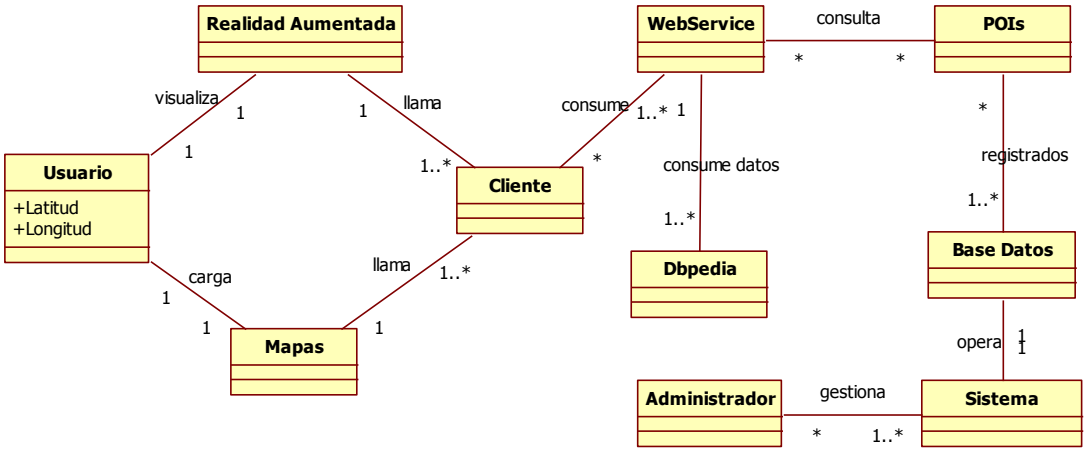

Fig. 6. Modelo de Dominio

3.3.1.3. Modelo de casos de uso

El objeto principal de los casos de uso es describir de forma natural las funcionalidades de un sistema a desarrollar y su empleo se realiza en el proceso de especificación de requisitos del sistema.

En este apartado utilizaremos los casos de uso como forma de acercar al lector las funcionalidades que la aplicación va a desempeñar frente al usuario. Sin embargo, no vamos a profundizar demasiado en las posibilidades que ofrece esta técnica, sino que vamos a limitarnos en aquellos aspectos que son más ilustrativos que le ayuden a comprender mejor qué es lo que realiza la aplicación.

Con el objeto de especificar a mayor detalle la relación entre requisitos y casos de uso se adjunta la siguiente tabla:

Cód.	**Requisito**	**Caso de uso relacionado**
RF01	Cargar Vista RA	Cargar Realidad Aumentada
RF02	Cargar Mapas	Cargar Mapas
RF03	Mostrar detalles	Mostrar Detalles
RF04	Obtener Ubicación Actual	Obtener Ubicación
RF05	WebService UTPLAR	Webservice
RF06	Consultas DbPedia	Consultas DbPedia
RF07	Autenticación UTPLAR Web	Autentificación del Sistema UTPLAR
RF08	Agregar Nuevos POIs	Nuevo POI
RF09	Actualizar POIs	Actualizar POI
RF10	Eliminar POIs	Eliminar POI

Tabla 5. Relación de Requisitos y casos de uso

En la figura 7 se representa el diagrama de casos de uso de la Aplicación.

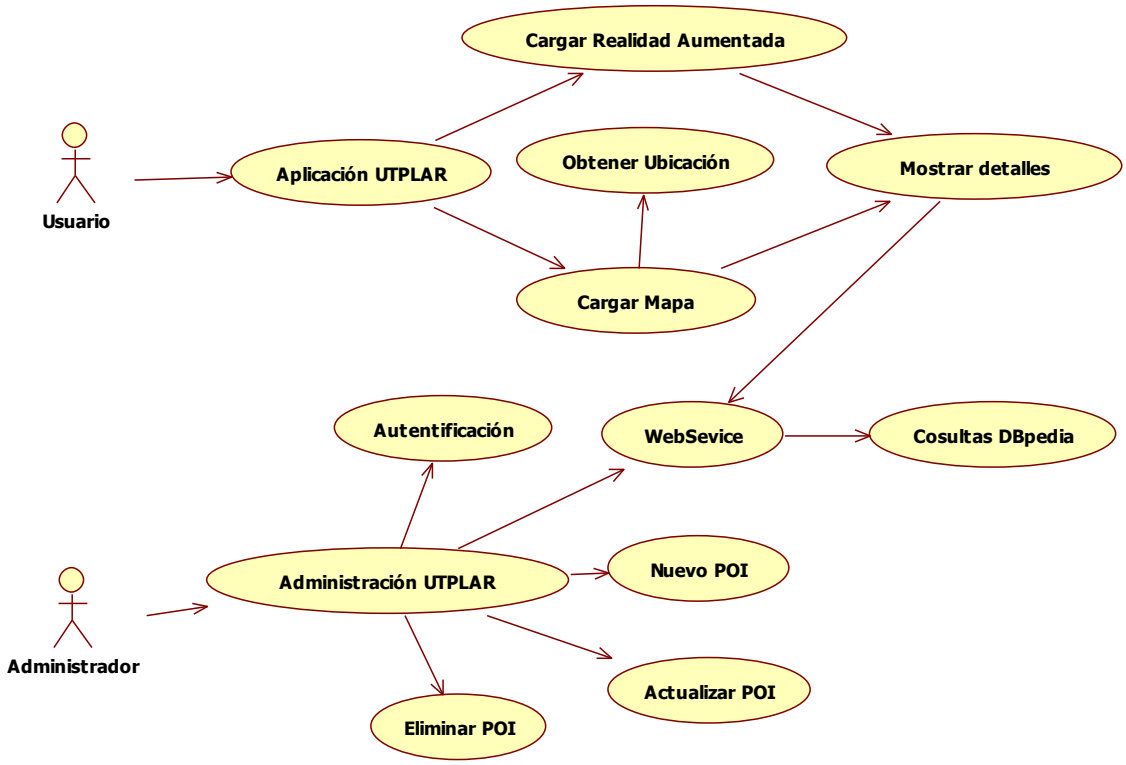

Fig. 7. Casos de Uso UTPLAR

3.3.2. Análisis y Diseño Preliminar

3.3.2.1. Descripción de los casos de uso

Cargar Realidad Aumentada

Permite cargar la vista en realidad aumentada para sitios y centros UTPL, así como otros sitios cercanos.

Nombre	Cargar Realidad Aumentada
Req. que satisface	RF01 - Cargar Vista RA
Actor	Usuario
Descripción	Carga el servicio de Realidad Aumentada mediante íconos.
Precondiciones	1. El usuario debe haber iniciado la Aplicación 2. Señales del GPS. 3. Conexión estable a Internet.

	4. Existencia de POIs en el sistema.
Postcondiciones	1. Vista de Realidad Aumentada mediante iconos de acuerdo al punto de interés (sitios, centros, lugares) que desee visualizar.
Flujo	1. Iniciar la aplicación 2. Comprobar señales GPS e Internet 4. Llamar a la vista de cámara. 4. Consultar datos de POIs del webservice. 5. Presentar POIs mediante íconos.

Tabla 6. Caso de Uso Carga Vista RA

Cargar Mapas

Permite cargar un mapa digital y visualizar cada uno de los POI distribuidos en el mismo.

Nombre	Cargar Mapas
Req. que satisface	RF02 – Cargar Mapas
Actor	Usuario
Descripción	Cargar un mapa junto a iconos que representen cada POI almacenado en el Sistema.
Precondiciones	1. El usuario debe haber iniciado la Aplicación 2. Señales del GPS. 3. Conexión estable a Internet. 4. Existencia de POIs en el sistema.
Postcondiciones	1. Cargar Mapa Digital. 2. Cargar los POIs distribuidos sobre el mapa.
Flujo	1. Iniciar la aplicación 2. Comprobar señales GPS e Internet 4. Cargar Mapa. 5. Consultar datos de POIs del webservice. 6. Cargar Overlay de POIs.

Tabla 7. Caso de Uso Cargar Mapas

Mostrar Detalles de un POI

Permite visualizar mayor información sobre un POI específico.

Nombre	Mostrar Detalles
Req. que satisface	RF03 – Mostrar Detalles
Actor	Usuario
Descripción	Mostrar el detalle de un POI seleccionado por el usuario. Esta funcionalidad debe permitirse tanto accediendo desde el mapa, como también desde la vista RA.
Precondiciones	1. Cargar iconos de POIs. 2. Existencia de detalles de cada POI en el sistema.
Postcondiciones	1. El usuario podrá visualizar información específica de un POI a través de una ventana temporal.
Flujo	1. Iniciar algún servicio de RA o Mapas. 2. Seleccionar un ícono (POI). 3. Cargar un Popup temporal. 4. Presentar los datos del POI.

Tabla 8. Caso de Uso Mostrar Detalles POI

Obtener ubicación actual

Permite conocer la ubicación actual del usuario dentro de un mapa cargado.

Nombre	Obtener Ubicación
Req. que satisface	RF04 - Obtener Ubicación
Actor	Usuario
Descripción	Permite reconocer la ubicación actual del usuario con respecto a un mapa cargado.
Precondiciones	1. El usuario debe haber iniciado el servicio de mapas. 2. Señales del GPS. 3. Conexión estable a Internet.
Postcondiciones	1. Cargar un icono representativo de la ubicación

	actual del usuario.
Flujo	1. Iniciar un servicio de mapas. 2. Obtener coordenadas con el GPS. 3. Mostrar un ícono con la posición actual.

Tabla 9. Caso de Uso Obtener Ubicación

Construcción de Servicios Web

Permite al cliente móvil obtener los POIs almacenados en el Sistema.

Nombre	Webservice
Req. que satisface	RF05 – WebService UTPLAR
Actor	Administrador
Descripción	Servicio Web RestFull que consulta datos del Sistema
Precondiciones	1. Existencia de POIs en el Sistema UTPLAR. 2. Acceso a la DB del sistema.
Postcondiciones	1. El Webservice realizará consulta de datos y los mostrará en formato xml.
Flujo	1. Cargar el url del Servicio. 2. Enviar parámetros para consultas específicas de POI.

Tabla 10. Caso de Uso Construcción del Servicio Web

Consultas a DBpedia

Permite realizar consultas sobre un recurso en DBpedia y complementar información en POIs de lugares de Loja.

Nombre	Consultas DBpedia
Req. que satisface	RF06 - Consultas DBpedia
Actor	Administrador
Descripción	Realizar consultas a DBpedia de acuerdo al POI de la ciudad seleccionado.
Precondiciones	1. Existencia del recurso en DBpedia. 2. El POI debe estar referenciado a un recurso en

	DBpedia.
Postcondiciones	El Servicio devuelve los siguientes predicados: label, abstract y comment, en el idioma que el usuario haya enviado como parámetro.
Flujo	1. Utilizar el servicio sparql de DBpedia. 2. Enviar la consulta SPARQL dentro de la url. 3. Tratar los datos obtenidos. 4. Presentarlos en nuestro servicio en formato xml.

Tabla 11. Caso de Uso Consultas a DBpedia.

Autentificación del Sistema UTPLAR

Permite realizar consultas sobre un recurso en DBpedia y complementar información en POIs de lugares de Loja.

Nombre	Autentificación del Sistema UTPLAR
Req. que satisface	RF07 – Autentificación
Actor	Administrador
Descripción	Permite realizar una autenticación para ingresar al sistema UTPLAR de Administración.
Precondiciones	1. El usuario debe estar registrado en el sistema. 2. Debe conocer usuario y contraseña.
Postcondiciones	1. El usuario puede entrar a la Administración del Sistema si su login es válido.
Flujo	1. Ingresar nombre de usuario 2. Ingresar contraseña 3. Validar datos en el sistema. 4. Mostrar el menú de Administración.

Tabla 12. Caso de Uso Autentificación al Sistema UTPLAR

Agregar nuevos POIs

Permite agregar un nuevo punto de interés a la base de datos del sistema.

Nombre	Nuevo POI
Req. que satisface	RF08 – Agregar nuevos POIs
Actor	Administrador
Descripción	Funcionalidad para agregar un nuevo POI.
Precondiciones	1. El usuario debe autentificarse. 2. Seleccionar un tipo de POI.
Postcondiciones	1. Presentar formulario para agregar un nuevo POI.
Flujo	1. Ingresar a la administración UTPLAR. 2. Seleccionar un tipo de POI. 3. Llenar el formulario. 4. Guardar los datos.

Tabla 13. Caso de Uso Agregar nuevos POIs

Actualizar POIs

Permite actualizar un punto de interés existente en la base de datos del sistema.

Nombre	Actualizar POI
Req. que satisface	RF09 – Actualizar POI
Actor	Administrador
Descripción	Actualizar un punto de interés determinado (sitios, centros, lugares).
Precondiciones	1. El usuario debe autentificarse. 2. Existencia del POI.
Postcondiciones	1. Listado de POIs existentes. 2. Formulario de edición de atributos del POI.
Flujo	1. Ingresar a la administración UTPLAR. 2. Seleccionar un tipo de POI. 3. Elegir la opción de Actualizar. 4. Guardar cambios.

Tabla 14. Caso de Uso Actualizar POIs

Eliminar POIs

Permite eliminar un punto de interés existente en la base de datos del sistema.

Nombre	Eliminar POI
Req. que satisface	RF10 – Eliminar POI.
Actor	Administrador
Descripción	Eliminar un POI registrado en el Sistema.
Precondiciones	1. El usuario debe autentificarse. 2. Existencia del POI.
Postcondiciones	Listado de POIs existentes. Confirmar eliminación.
Flujo	1. Ingresar a la administración UTPLAR. 2. Seleccionar un tipo de POI. 3. Elegir la opción de Eliminar. 4. Confirmar Eliminación.

Tabla 15. Caso de Uso Eliminar POIs

3.3.3. Diseño

3.3.3.1. Arquitectura

La arquitectura de la aplicación permite visualizar como está diseñada la solución a nivel global. La solución propuesta en el proyecto se basa en la Arquitectura LOCWD Mobile que se observa en la figura 8; una arquitectura distribuida diseñada con la finalidad de reducir tiempos de carga en el cliente móvil, agilizar procesos y aprovechar recursos de la web.

En cuanto a la arquitectura física tenemos varios componentes incluyendo: Sistema GPS, Red 3G, Servidores propios y externos; como se puede observar en la figura 9.

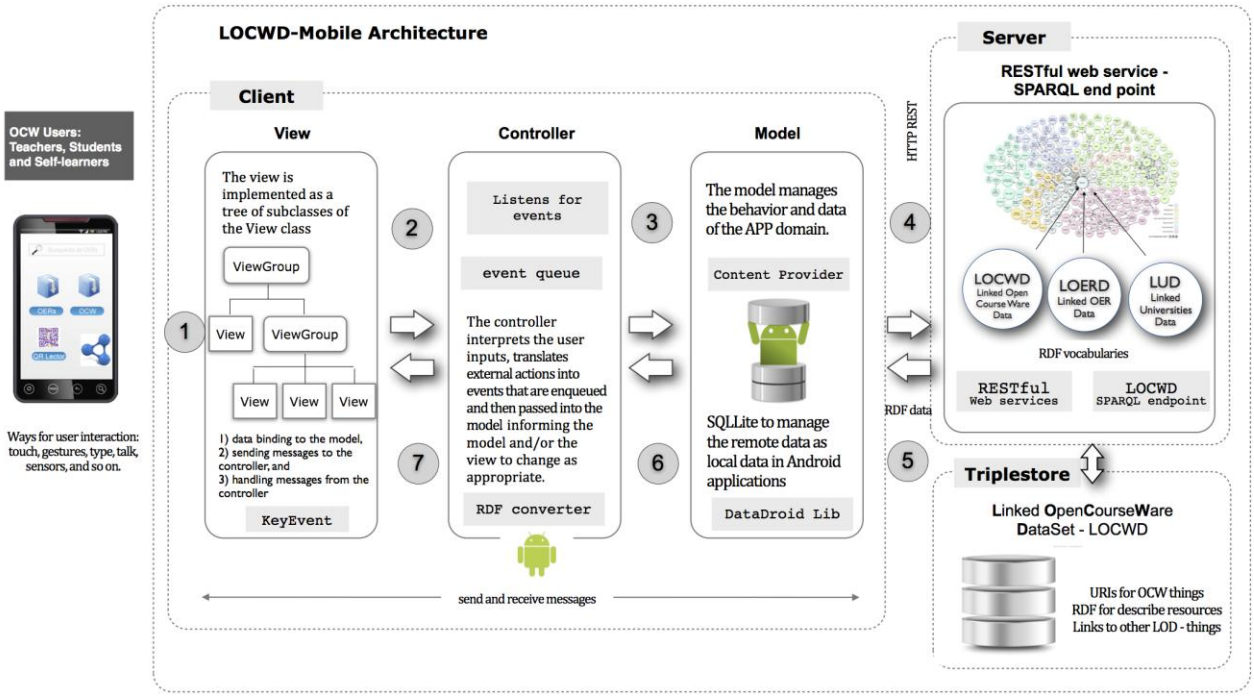

Fig. 8. Arquitectura LOCWD-Mobile

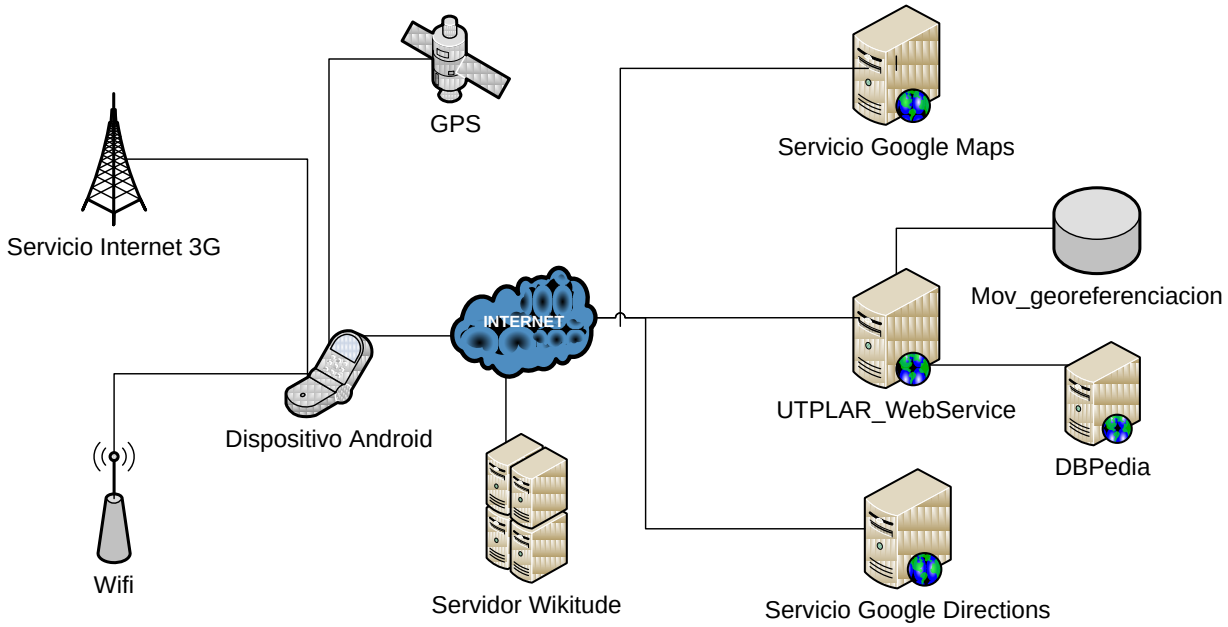

Fig. 9. Arquitectura física de UTPLAR

3.3.3.2. Componentes

La arquitectura implementada se estructura de varios componentes (Ver Fig. 9), mencionados a continuación:

SERVIDOR

Encargada de dar respuesta a las peticiones del cliente móvil; dentro del proyecto se utilizarán varios servidores, servicios web y Apis; entre ellos tenemos:

Base de Datos: Se utiliza un servidor MySQL para el almacenamiento de información de los POI UTPL (sitios, paradas, centros a distancia), así como sus relaciones.

Servicios Web: Se utiliza un servidor web Glassfish para el montado del Web Service RestFUL; intercambio de datos en formato XML.

Servicios Web Externos: Se consume uno de los servicios de IRBU (Mora C, 2012) para obtener las paradas más cercanas dentro de un rango establecido por el usuario.

Servicio Google Maps: Servicio utilizado para visualizar los mapas dentro de la aplicación, similar al servicio web.

API Google Directions: El API Google Directions nos permitirá obtener los datos en formato XML sobre la ruta entre dos puntos.

DBpedia: Versión Semántica del contenido de Wikipedia.

CLIENTE

La aplicación UTPLAR se estructura de dos componentes principales, estos son:

Capa Presentación: Está compuesta por todos los layouts o vistas creadas en el proyecto. Su principal objetivo es presentar una interfaz manejable a la interacción del usuario.

Capa de Negocio: Constituida por cada Activity, y cada uno de los controladores que permiten manejar los eventos a partir de acciones que realice el usuario. Su función principal es establecer comunicación entre la presentación y la capa de datos.

3.3.3.3. Diagramas de secuencia

Para mejorar la percepción en cuanto a las funcionalidades de la aplicación se han elaborado los siguientes diagramas de secuencia:

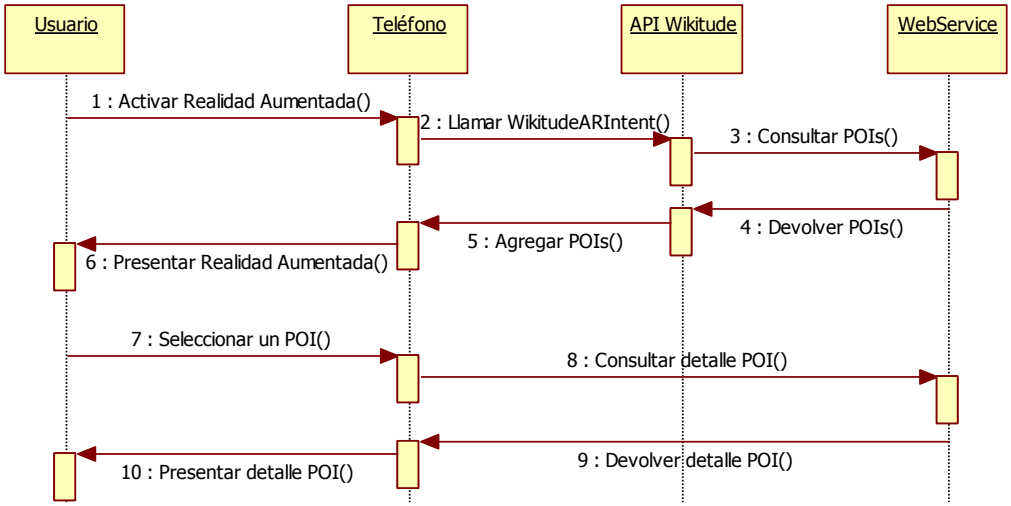

Fig. 10. Diagrama de secuencia – Realidad Aumentada y Mostrar Detalles

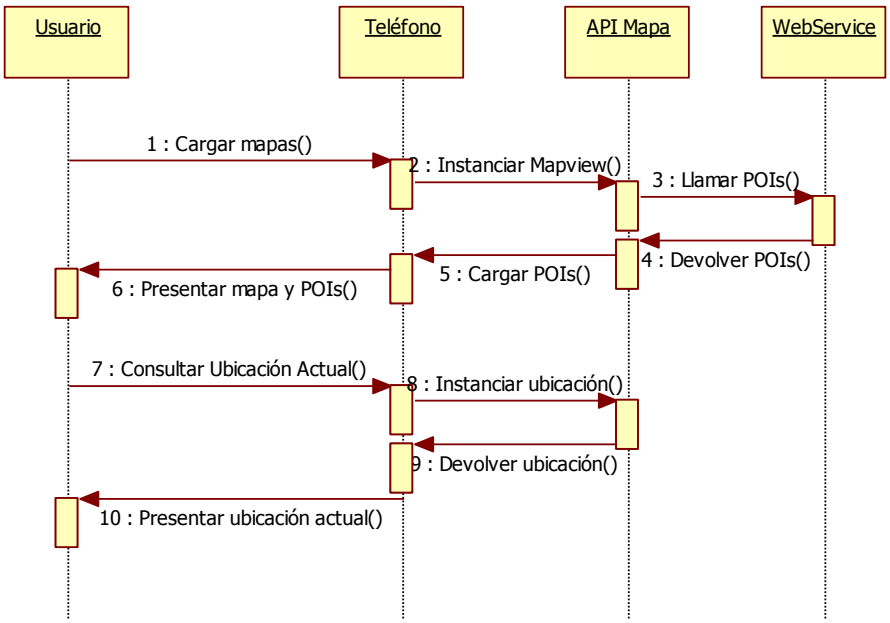

Fig. 11. Diagrama de secuencia – Cargar Mapas y Ubicación actual

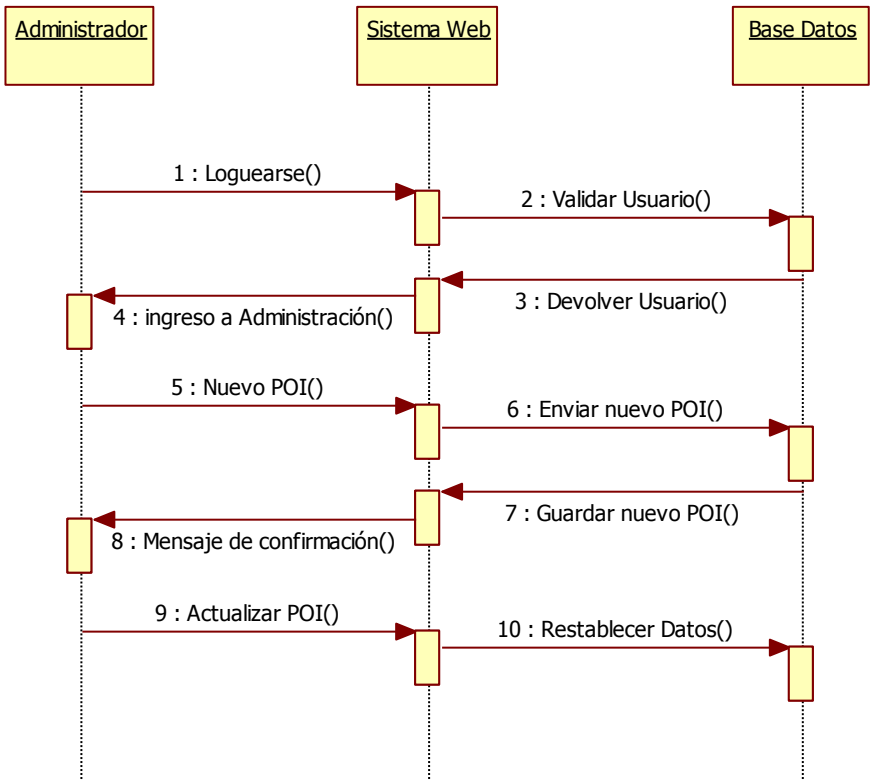

Fig. 12. Diagrama de secuencia – Administración UTPLAR

(Autenticación, Nuevo/Actualizar/Eliminar POI)

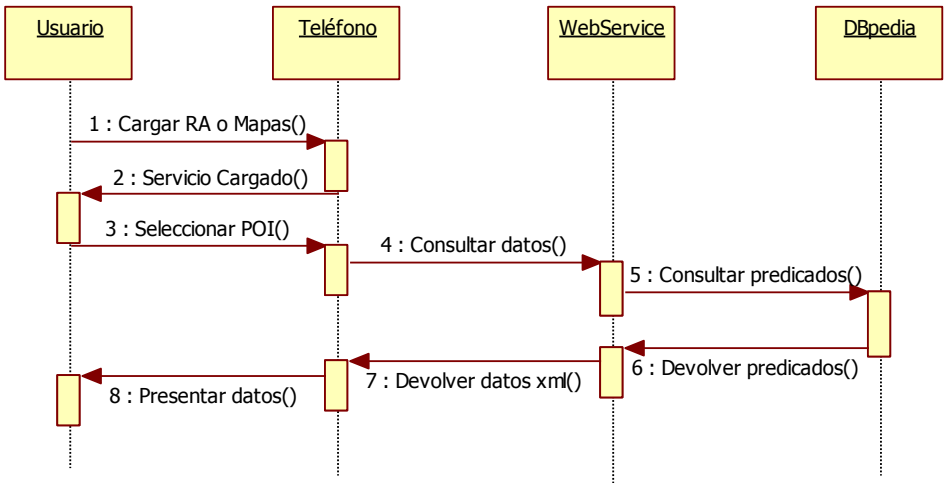

Fig. 13. Diagrama de secuencia – Consulta DBpedia

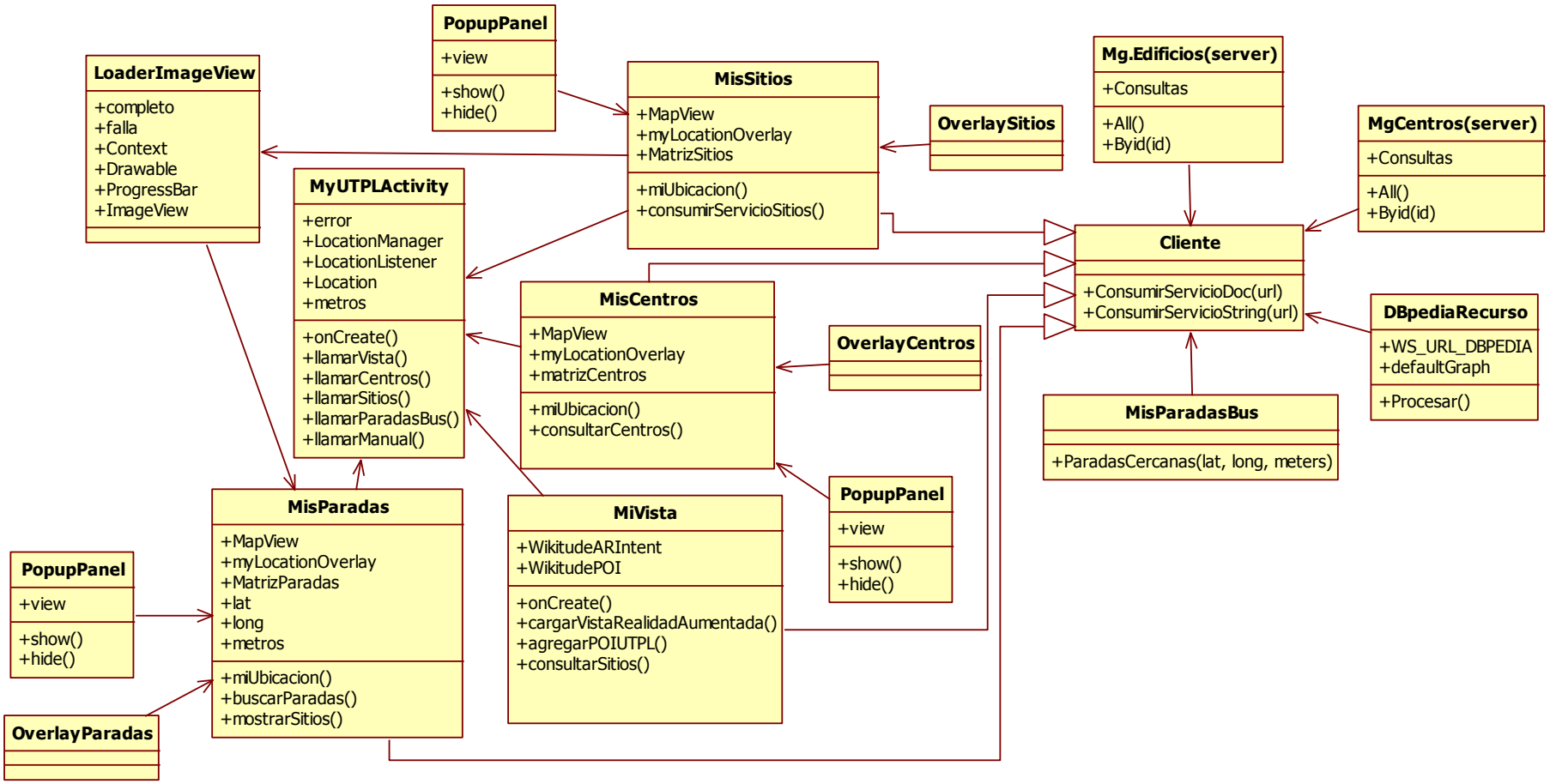

Fig. 14. Diagrama de clases (cliente-servidor)

3.3.3.4. Diagrama de clases

En función de los requisitos y problema resolver, se ha definido un modelo de clases, atributos y operaciones. Mediante este diagrama se representa un esquema de la futura implementación del sistema, así como permite dar una idea bastante cercana a la forma en la que se ha de resolver el problema. El diagrama de clases del sistema UTPLAR se puede observar en la figura 14.

3.3.4. Implementación

3.3.4.1. UTPLAR

UTPLAR es el nombre de la aplicación de Realidad Aumentada para la Universidad Técnica Particular de Loja; cuyo objetivo principal es brindar información en tiempo real de los puntos de interés del campus, centros universitarios, lugares importantes de la ciudad y servicios de transporte UTPL a través de realidad aumentada y geolocalización en mapas.

Las funcionalidades de la aplicación se detallan a continuación:

Menú Principal

Menú o interfaz principal, mismo que permite llamar a todas las funciones integradas en la Aplicación. Este menú está conformado por 6 botones:

1. Realidad Aumentada UTPL
2. Sitios UTPL
3. Paradas de Bus UTPL
4. Centros UTPL
5. Otros Sitios Cercanos
6. Ayuda

El menú principal es la primera vista en cargar al acceder a la aplicación UTPLAR. Las funciones están colocadas de forma vertical, y cada función es representada con un ícono descriptivo, así como el nombre del servicio. (Ver fig. 15)

Antes de iniciar cualquiera de los servicios se realiza una validación del estado del GPS y conectividad de datos (internet). Dependiendo del tamaño de la pantalla se activa un scrollbar en caso de requerirlo.

Además de las funciones, existe un menú para: activar/desactivar GPS, configurar metros para buscar paradas, ver información del autor.

Fig. 15. Interfaz Principal UTPLAR

a. Realidad Aumentada UTPL

La vista con realidad aumentada es la funcionalidad principal de la aplicación. La misma que carga el navegador de Wikitude, y dependiendo de la dirección apuntada de la cámara presenta íconos representativos de cada lugar enfocado (figura 16). Para ver información más detalla del icono seleccionado se debe pulsar sobre el icono y aparecerá la información sobre el lugar, la distancia hacia el lugar, así como una flecha indicando la dirección del sitio para llegar.

Para hacer uso se debe verificar el funcionamiento de la cámara, gps y wifi del dispositivo móvil; además que se requiere tener instalada la aplicación Wikitude ya que el API consume los servicios de RA de esta aplicación.

Fig. 16. Función: Mi vista UTPLAR

b. Sitios UTPL

Al pulsar esta opción se carga un mapa de GoogleMaps, junto a un conjunto de íconos que representan cada uno de los sitios UTPL registrados en la base de datos, así como un ícono azul que representa la ubicación actual del dispositivo (fig. 17).

Dentro de las funciones que brinda el mapa una vez cargado se tiene: zoom in, zoom out y desplazar el mapa.

Otra de las funcionalidades del servicio es que al pulsar un ícono de algún sitio presenta información específica del mismo: nombre, descripción e imagen.

c. Paradas de Bus UTPL

El servicio de Parada de Buses UTPL al igual que el servicio anterior carga un mapa digital para la visualización de las paradas cercanas según los metros de distancia seleccionados; este valor por defecto es de 200 m. Sin embargo se puede configurar desde la vista principal.

En este servicio se muestran datos referentes a las paradas de bus UTPL, mismos que son obtenidos mediante una petición POST al servidor IRBUS luego de pulsar un icono específico de una parada (fig. 18):

Fig. 17. Función Sitios UTPL – UTPLAR

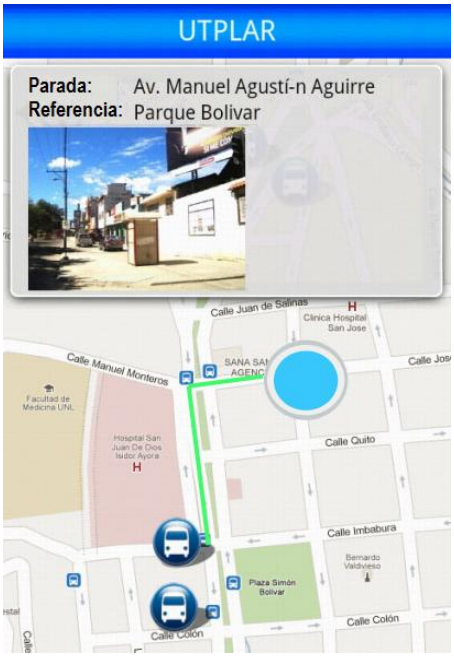

Fig. 18. Función Paradas Bus UTPL – UTPLAR

Parada: Es la ubicación real de la parada, aquí se muestran nombres de las calles o sus intersecciones.

Referencia: Nombre de lugares representativos cerca de la parada para fácil ubicación.

Foto de la Parada: Una imagen gráfica del lugar donde los estudiantes esperan el arribo del bus.

Además de la información obtenida se agregó el servicio para graficar la ruta de llegada a una parada específica. Esto se realiza a través del Servicio Google Directions, que obtiene la ruta en formato xml. Y se lo representa a través de un algoritmo en el mapa actual cargado.

d. Centros UTPL

El servicio de Centros UTPL es similar a Sitios UTPL con la diferencia del área cubierta, ya que esta es para todo el territorio ecuatoriano y poder visualizar los centros asociados y oficinas UTPL de todo el país (fig. 19). A más de ello podemos visualizar información detallada de un centro por ejemplo: la ciudad, el coordinador, dirección y teléfono.

Fig. 19. Función: Mis Centros - UTPLAR

e. Otros Sitios Cercanos

El último servicio de mapas es sitios cercanos, similar a todos los servicios de mapas, con la diferencia que este servicio se enfoca en

mostrar algunos sitios importantes de la ciudad (fig. 20), además que implementa también realidad aumentada.

Otro de los servicios de esta funcionalidad es consultar información de la DBpedia con respecto al sitio de interés: label, abstract y comment; en los idiomas: inglés y español. (fig. 21)

Fig. 20. Otros Sitios Cercanos

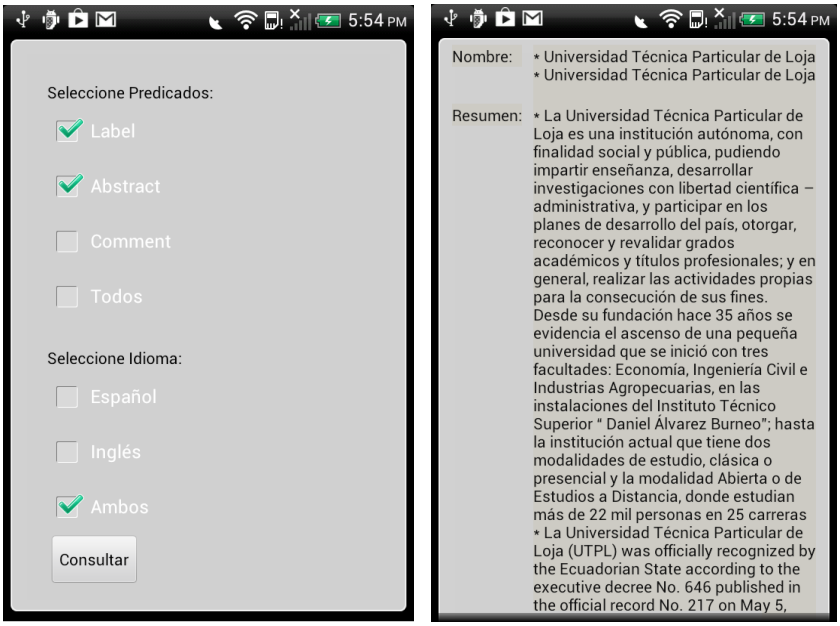

Fig. 21. Consulta del sitio en DBpedia

Menú de Configuración

Se ha creído conveniente disponer de un menú de configuración, sus opciones son:

Servicios: Permite acceder al menú de GPS y WIFI para activar o desactivar estos servicios.

Paradas: Seleccionar la cantidad de metros de distancia mínima para obtener las paradas más cercanas según la ubicación actual del dispositivo.

Salir: Permite salir de la aplicación, cerrando cualquier proceso de la misma.

3.3.4.1.1. Historial de prototipos

Como parte del desarrollo de la aplicación se entregaron varios prototipos en función de cada requisito requerido, es por eso que a continuación se describen las características de cada versión de propotipo.

Prototipo 1

El primer prototipo se enfocó en hacer pruebas de mapas y geolocalización de POIs; por ende se desarrollo una pequeña aplicación que cargaba un mapa y un conjunto de iconos que representaba cada POI y al pulsar se presentaba un mensaje instantáneo con el nombre del punto. Para desarrollar este prototipo se hizo uso de librerías de google maps, un mapview, overlays y mensajes toast provistos por la plataforma.

Prototipo 2

El segundo prototipo se implementó el servicio de realidad aumentada con 6 puntos de interés del campus: UGTI, Cafeteria, M. Abierta, Octógono, Cruz, E. Central, Centro Convenciones, se mostraban los primeros íconos tomados como referencia del lugar. Para este prototipo se realizaron pruebas con varias APIs de realidad aumentada, entre ellas: ARToolkit, Mixare, ARDroid y Wikitude (api seleccionada). Cabe

recordar que en los prototipos mencionados se trabajan con datos agregados desde código.

Prototipo 3

El tercer prototipo mejoró la funcionalidad de mapas y realidad aumentada ya que una vez construidos los webservice y recopilación de datos. Se tenían más datos para representar, entre ellos: sitios UTPL, centros universitarios, paradas de bus (WebService IRBU). Además que se implementó paneles para presentar ventanas de información del POI donde se podía incluir texto e imágenes del POI. Además que dentro del servicio de paradas de bus se agregó la graficación de rutas para llegar a una parada. En este prototipo se desarrollaron webservices xml desde netbeans, y se usó un api de google maps (solo es web).

Prototipo 4

En el cuarto prototipo se trabajó en la mejora de interfaz principal, creación de íconos y estilos, así como validar los servicios de GPS e internet antes de iniciar algún servicio. En este prototipo se crearon funciones para manejar la conectividad a internet y obtención de coordenadas.

Prototipo 5

En el último prototipo se agregó la funcionalidad para consultar datos sobre sitios de interés existentes en BDpedia, por ejemplo: nombre, resumen, comentarios en inglés y español. Para ello se creó un servicio web que recibe el uri y el lenguaje a consultar, donde internamente realiza una consulta a DBpedia y arroja los resultados que solicitamos en formato xml.

Para finalizar se adjunta una tabla comparativa de cada una de las versiones de prototipo de UTPLAR.

Prototipo	Aprendizaje	Herramientas	Tiempo
1	Utilización de GPS. Utilización Google maps API. Creación de Overlays.	Google maps API	2 semanas
2	Utilización básica de Wikitude API	Wikitude API	2 semanas
3	Construcción de Servicios Web RestFull from database. Maximizar funcionalidades de Wikitude API. Utilización API Google Directions. Cargar imágenes en ImageView desde un servidor web.	RestFull WebServices Wikitude API	8 semanas
4	Validación de conectividad internet y GPS. Uso de scrollbars.		2 semanas
5	Consumir datos en formato json de DBpedia.	DBpedia	1 semana

Tabla 16. Resumen Historial Prototipos

3.3.4.2. Administrador Web

El administrador web facilita el manejo de POIS del sistema, permitiendo realizar algunas operaciones con respecto a un sitio de interés en caso de ser requerido. Entres sus funcionalidades tenemos:

- Autenticación del Sistema
- Administración de usuarios
- Administración de sitios
- Administración lugares cercanos
- Administración de centros universitarios

A continuación se describe cada uno de ellos:

a. Autenticación del Sistema

Servicio que permite acceder a un administrador al sistema, y facilitar la ejecución de operaciones con la base de datos (fig. 22).

Fig. 22 Autenticación del sistema

b. Usuarios

La administración de usuarios permite crear nuevos administradores, actualizar o eliminar alguno de ellos.

Fig. 23 Administración de usuarios

c. Sitios UTPL

La administración de sitios o POIS UTPL permite crear un nuevo sitio, actualizar o eliminar alguno de ellos.

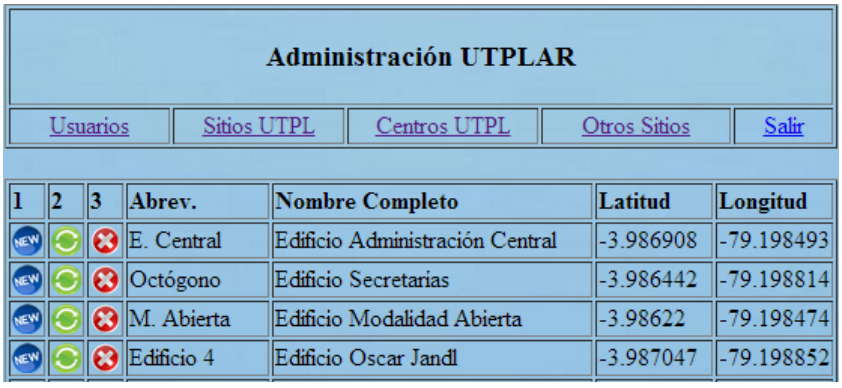

Fig. 24 Administración de sitios UTPL

d. Centros UTPL

La administración de centros universitarios permite registrar un nuevo centro, actualizar o eliminar alguno si es requerido.

Administración UTPLAR

Usuarios | Sitios UTPL | Centros UTPL | Otros Sitios | Salir

1	2	3	Tipo	Ciudad	Dirección	Teléfono
			Centro Regional	Quito	Av. 6 de Diciembre y Alpallana	2558239
			Centro Regional	Guayaquil	Av. Kennedy N°333 entre la Av. San Jorge y calle F.	2397901
			Centro Provincial	Ambato	Avenida Guaytambos y Manzanas esquina, SECTOR FICOA	2822714

Fig. 25 Administración de centros universitarios

e. Otros lugares

La administración de otros sitios cercanos permite agregar nuevos lugares de la ciudad, actualizar o eliminar alguno de ellos.

Administración UTPLAR

Usuarios | Sitios UTPL | Centros UTPL | Otros Sitios | Salir

1	2	3	Abrev.	Nombre Completo	Latitud	Longitud
			P. Ciudad	Monumento Puerta de la Ciudad	-3.98953	-79.204024
			H. Ayora	Hospital Isidro Ayora	-3.993619	-79.206106
			Catedral	Iglesia Catedral	-3.995599	-79.201143
			Podocarpus	Parque Nacional Podocarpus	-4.361474	-78.974052
			Jipiro	Parque Recreacional Jipiro	-3.972106	-79.203536

Fig. 26 Administración de lugares cercanos

3.3.4.3. Pruebas

3.3.4.3.1. Casos de prueba

El presente plan de pruebas se basa en los casos de uso funcionales descritos en la sección 3.3.2.1.1, mismos que son los servicios principales y se detallan a continuación:

Nro.		1	
Nombre Caso Prueba		Cargar Vista RA	
Realizado por		Usuario	
Descripción		Testear el servicio de realidad aumentada.	
Precondiciones		1. El usuario debe haber iniciado la Aplicación 2. Señales del GPS. 3. Conexión estable a Internet. 4. Existencia de POIs en el sistema.	
Postcondiciones		1. Visualizar vista de realidad aumentada.	
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Clic en un servicio de Realidad Aumentada.	Cargando el servicio de Realidad Aumentada.	OK
2	Validar GPS	Si el GPS esta desactivado presentar mensaje de error de GPS. Caso contrario continuar siguiente paso.	OK
3	Validar conexión Internet	Si no existe conexión presentar mensaje de error de Red. Caso contrario continuar paso 4.	OK
4	Cargar cámara	Presentar enfoque de cámara.	OK
5	Cargar POIs	Presentar POIS en la pantalla del dispositivo.	OK

Tabla 17. Caso de Prueba Cargar Vista RA

Nro.	2
Nombre Caso Prueba	Cargar Mapas
Realizado por	Usuario
Descripción	Testear el servicio de geolocalización de POIs en mapas.
Precondiciones	1. El usuario debe haber iniciado la Aplicación 2. Señales del GPS. 3. Conexión estable a Internet. 4. Existencia de POIs en el sistema.
Postcondiciones	1. Visualizar el mapa y los POIs localizados.

Paso	Acción	Respuesta Esperada del sistema	Res.
1	Clic en un servicio de Mapas.	Instanciar el servicio de Realidad Aumentada.	OK
2	Validar GPS	Si el GPS esta desactivado presentar mensaje de error de GPS. Caso contrario continuar siguiente paso.	OK
3	Validar conexión Internet	Si no existe conexión presentar mensaje de error de Red. Caso contrario continuar paso 4.	OK
4	Cargar mapa	Presentar mapa y POIs.	OK
5	Desplazar	Desplazamiento del mapa.	OK
6	Agrandar	Zoom del mapa.	OK
7	Reducir	Zoom in del mapa.	OK

Tabla 18. Caso de Prueba Cargar Mapas

Nro.	3		
Nombre Caso Prueba	Mostrar Detalles de un POI		
Realizado por	Usuario		
Descripción	Testear el servicio de mostrar detalles de un POI.		
Precondiciones	1. El usuario debe haber iniciado un servicio de realidad aumentada o mapas. 2. Conexión estable a Internet.		
Postcondiciones	1. Visualizar una ventana con mayor información del POI.		
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Clic en un POI	Mostrar el detalle de un POI.	OK

Tabla 19. Caso de Prueba Mostrar Detalles

Nro.		4	
Nombre Caso Prueba		Obtener Ubicación	
Realizado por		Usuario	
Descripción		Testear el servicio de ubicación en tiempo real.	
Precondiciones		1. El usuario debe haber iniciado un servicio de mapas. 2. Señales del GPS. 3. Conexión estable a Internet.	
Postcondiciones		1. Visualizar la ubicación del usuario a través de un ícono o figura.	
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Clic en un servicio de mapas	Iniciar el servicio de mapas.	OK
2	Buscar señales GPS	Mostrar la ubicación	OK

Tabla 20. Caso de Prueba Ubicación Actual

Nro.		5	
Nombre Caso Prueba		Webservice	
Realizado por		Administrador	
Descripción		Testear el webservice del sistema.	
Precondiciones		1. Existencia de POIs en el Sistema UTPLAR. 2. Acceso a la DB del sistema.	
Postcondiciones		1. Presentar POIs en formato xml.	
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Escribir url del servicio	Cargar datos de POIs en xml	OK
2	Escribir url del servicio+parámetro	Cargar datos de un POI en xml.	OK

Tabla 21. Caso de Prueba Webservice

Nro.		6	
Nombre Caso Prueba		Consultas DBpedia	
Realizado por		Usuario	
Descripción		Testear las consultas de predicados a DBpedia.	
Precondiciones		1. Existencia de datos del POI en DBpedia.	
Postcondiciones		1. Presentar predicados consultados en formato xml.	
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Clic en un POI de sitios cercanos.	Nueva ventana informativa.	OK
2	Clic en Boton DBpedia	Mostrar menú de consulta personalizado.	OK
3	Seleccionar Opciones	Mostrar opciones seleccionadas	OK
4	Clic en botón Consultar	Mostrar nueva ventana con lo solicitado.	OK

Tabla 22. Caso de Prueba Consultas a DBpedia

Nro.		7	
Nombre Caso Prueba		Autenticación	
Realizado por		Administrador	
Descripción		Testear el servicio de autentificación del sistema de administración UTPLAR.	
Precondiciones		1. Usuario registrado	
Postcondiciones		1. Ingreso a la administración de la aplicación.	
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Entrar al Sitio de Administración	Cargar formulario de logueo.	OK
2	Ingresar usuario y contraseña incorrecta	Presentar nuevamente el formulario.	OK
3	Ingresar usuario y contraseña correcta	Presentar el menú de administración.	OK

Tabla 23. Caso de Prueba Autenticación

Nro.		8	
Nombre Caso Prueba		Nuevo POI	
Realizado por		Administrador	
Descripción		Testear el servicio para ingresar nuevos POIs.	
Precondiciones		1. Usuario logueado.	
Postcondiciones		1. Formulario para insertar un nuevo POI.	
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Seleccionar un tipo de POI.	Cargar formulario para el nuevo POI.	OK
2	Enviar datos POI.	Presentar mensaje de confirmación de la inserción.	OK

Tabla 24. Caso de Prueba Nuevo POI

Nro.		9	
Nombre Caso Prueba		Actualizar POI	
Realizado por		Administrador	
Descripción		Testear el servicio para actualizar un POI existente.	
Precondiciones		1. Usuario logueado.	
Postcondiciones		1. Formulario con campos editables de los datos del POI.	
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Seleccionar un tipo de POI.	Cargar POIs registrados.	OK
2	Seleccionar actualizar.	Presentar formulario con campos editables.	OK
3	Enviar nuevos datos.	Presentar mensaje de confirmación de actualización.	OK

Tabla 25. Caso de Prueba Actualizar POI

Nro.		10	
Nombre Caso Prueba		Eliminar POI	
Realizado por		Administrador	
Descripción		Testear el servicio para eliminar un POI.	
Precondiciones		1. Usuario logueado.	
Postcondiciones		1. Boton para eliminar un POI.	
Paso	**Acción**	**Respuesta Esperada del sistema**	**Res.**
1	Seleccionar un tipo de POI.	Cargar POIs registrados.	OK
2	Seleccionar eliminar.	Presentar mensaje para confirmar la eliminación.	OK
3	Confirmar operación.	Presentar mensaje de la operación realizada.	OK

Tabla 26. Caso de Prueba Eliminar POI

3.3.4.3.2. Dispositivos y versiones

La aplicación fue diseñada bajo en la versión 2.2, ya que es el target inicial y valido para el correcto desempeño de este tipo de aplicaciones y es compatible con las versiones superiores. Sin embargo es necesario realizar pruebas en diferentes versiones a fin de evaluar cada una de las funciones.

A nivel general la aplicación tiene un nivel de compatibilidad muy bueno, es por eso que a continuación se detalla una tabla comparativa entre distintas versiones probadas:

DISPOSITIVOS	**Versión Android**			
	2.2	**2.3.4**	**2.3.6**	**4.0.2**
Galaxy ACE		X	X	
Galaxy SII			X	
HTC EVO 3D		X		X
Galaxy Nexus				X
Motorola DEFY mb526			X	
Sony Ericsson x10 mini	X			

Tabla 27. Pruebas en diferentes dispositivos móviles

3.3.4.3.3. Precisión y estabilidad de servicios

En la presente aplicación es de vital importancia la precisión de los datos a mostrar, así como el grado de estabilidad de la aplicación; es por eso que se ha realizado un plan de pruebas con respecto a los servicios principales seleccionados por el creador. La tabla se muestra a continuación:

FUNCIONES Y SERVICIOS	Precisión	Estabilidad
Carga de Vista RA	80-90%	95%
Carga de Mapa	100%	100%
Carga de POIs mapa	100%	100%
Dibujar Ruta	90%	100%
Mostrar Detalles	100%	100%
Consulta DBpedia	100%	100%
Administración Web	100%	100%
WebServices	100%	100%

Tabla 28. Pruebas de Precisión y Estabilidad

Cabe recordar que el tiempo de carga de cada uno de los servicios depende de la conexión de datos del dispositivo. Puesto que los datos se deben consultar desde el WebService.

3.3.4.3.4. Usabilidad

Dentro del conjunto de pruebas de usabilidad se califican algunos factores; en el presente proyecto se han seleccionado únicamente: facilidad de uso, la utilidad hacia el usuario y el diseño de la aplicación (no se ha tomado en cuenta el servidor debido a que este será administrado por el creador o una persona delegada, más no por un usuario común). Es por eso que a continuación se ha realizado una tabla luego de las pruebas realizadas:

Servicio	Complejidad de Uso	Conformidad Diseño
Interfaz Principal	Fácil	Excelente
Realidad Aumentada	Medio	Excelente

Mapa de Sitios UTPL	Fácil	Bueno
Mapa de Paradas Bus	Fácil	Excelente
Mapa de Centros	Fácil	Excelente
Mapa de Sitios Cercanos	Fácil	Excelente
Mi ubicación	Fácil	Excelente
Graficar ruta	Fácil	Bueno
Consulta DBpedia	Medio	Bueno

Tabla 29. Pruebas de Usabilidad

3.3.4.3.5. Pruebas de Stress

Un factor importante en estas pruebas es el rendimiento de la aplicación frente al número de usuarios que necesiten el servicio al mismo tiempo. En nuestro caso al contar con una arquitectura distribuida; el servidor es el encargado de devolver los datos de la base de datos a los clientes móviles. Lo que significa que el rendimiento de la app en sí depende del webservice.

En tal caso existe un administrador que mantiene el servidor activo quien configura el número de conexiones del mismo con el objeto de obtener un mejor rendimiento del servicio.

CAPITULO IV

4. CONCLUSIONES Y RECOMENDACIONES

4.1. CONCLUSIONES

- Uno de los principales desafíos del sistema de realidad aumentada UTPLAR es el registro preciso de los objetos virtuales con el entorno real. Esta funcionalidad tiene una exactitud entre el 90%, y depende de factores como la señal de red disponible, así como la exactitud del GPS, cabe recordar que se trabajó con un margen de error[15] de 10 a 20 metros.

- Las funcionalidades que trabajan con google maps son más estables en cuanto a la colocación de POIs, ya que no necesitan del servicio de GPS para esta función; sin embargo para servicios de direcciones es fundamental tanto el servicio de internet como el GPS, puesto que se realiza la petición en tiempo real.

- El api googles direction en relación al mapa de la ciudad de Loja maneja un 90% de precisión a nivel general debido a que el mapa aun está en desarrollo, sin embargo para la zona céntrica la precisión aumenta a un 95%.

- UTPLAR al utilizar una arquitectura distribuida, ha permitido maximizar recursos, reducir tiempos de procesamiento y sobre todo publicar y explotar datos de la web.

- UTPLAR se proyecta a ser una aplicación móvil que facilite contenido informativo de la UTPL de manera oportuna y novedosa, reducir tiempos de búsqueda de información y sobre todo brindar un mejor servicio.

[15] Margen de Error GPS:
http://www.madrimasd.org/cienciaysociedad/ateneo/dossier/galileo/mundogps/mundogps2.htm

4.2 RECOMENDACIONES

- En la actualidad existen muchas librerías y APIs que agilizan el proceso de desarrollo de una aplicación Android, es por eso que resulta de gran utilidad a la hora de desarrollar nuestras aplicaciones, no obstante se debe analizar si es necesario o no.

- Verificar que nuestro dispositivo funcione a perfección, más aun cuando se utilizan sensores de movimiento y localización. Ya que de esto dependerá el éxito de desarrollo y pruebas de nuestra aplicación.

- Revisar los permisos necesarios en el archivo de configuración AndroidManifest.xml para utilizar servicios como: Internet, Localización, etc.

- Realizar pruebas de la aplicación desarrollada en múltiples dispositivos móviles o versiones de la plataforma, permitirá conocer el grado de funcionamiento y portabilidad a nivel de hardware y software.

BIBLIOGRAFÍA

Azuma, R. (1995). A survey of augmented reality.

Bover, A. (2010). *Aplicación de Gestión de Información Geolocalizada en Android*. Obtenido de http://upcommons.upc.edu/pfc/bitstream/2099.1/11482/1/69369.pdf

ComPixels. (05 de 06 de 2010). *Top 5 Augmented Reality Apps for Android.* Obtenido de http://compixels.com/441/top-5-augmented-reality-apps-for-android

CONSUMER, E. (2011). *Realidad aumentada.* Obtenido de http://www.90grados.info/: http://static.consumer.es/www/tecnologia/infografias/swf//realidad-aumentada.swf

Doreen, M. J. (2003). *The Eye Magic Book.* . Obtenido de http://www.mindspacesolutions.com/demos/eyeMagicWorkShopReport.pdf

Handheld Augmented Reality, Christian Doppler Laboratory. (2011). Obtenido de http://handheldar.icg.tugraz.at/

HARP. (2012). Obtenido de http://isites.harvard.edu/icb/icb.do?keyword=harp

HiddenCreative. (2011). *AUGMENTED REALITY ACTIVE MARKET PARTICIPATION STATISTICS.* Obtenido de http://www.slideshare.net/HiddenCreative/marketing-with-augmented-reality-an-infographic

KAUFMANN, H., & MEYER, B. (2010). *Simulating Educational Physical Experiments in Augmented Reality.* Obtenido de http://www.ims.tuwien.ac.at/media/documents/publications/Physics_Experiments_AR.pdf

LABEIN. (2011). *Sistema de posicionamiento 3D sin marcadores basado en visión por computador.* Obtenido de http://www.labein.es/rasmap-w.nsf/Subproyectos_ceit.html

Lazar, A., Nikolić, N., Nikolić, S., & Predrag, R. (s.f.). *ARMED-Augmented Reality for the Medicine.* Obtenido de http://msacademic.rs/download/imaginecup/2011/projects/01%20AdHoc%20Team%20-%20ARMed.pdf

Sielhors, T., Obst, T., Burgkart, R., & Riener, R. &. (2004). *An Augmented Reality Delivery Simulator for Medical Training.* Obtenido de http://ami2004.loria.fr/PAPERS/26obetoebiel.pdf

Piedra, N., Chicaiza, J., & Lopez, J. (2012). *2012 04-19 (educon2012) emadrid upm combining linked data mobiles improve access ocw.* Obtenido de http://www.slideshare.net/emadridnet/2012-0419-educon2012-emadrid-upm-combining-linked-data-mobiles-improve-access-ocw-12744073

Android Developers: Android Developers Guide. (2012). Disponible en: http://developer.android.com/index.html

ANEXOS

ANEXO A

Estructura de una Aplicación

Al momento de crear un proyecto en Android automáticamente se generan un conjunto de directorios y archivos que permiten posteriormente generar la aplicación. Muchas veces será necesario crear uno o más directorios en caso de ser requeridos.

1. Directorio src

Contiene el código fuente de la aplicación, el código de la interzaz gráfica, clases auxiliares, etc. Al crear un proyecto Android en Eclipse se crea automáticamente una Activity principal, del tipo .java.

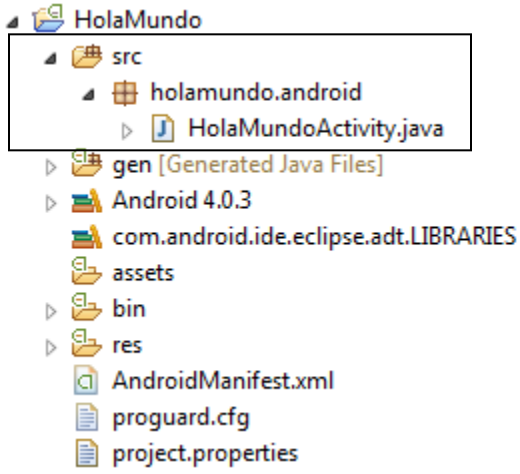

Fig. 27. Archivos básicos de un proyecto Android

2. Directorio Recursos

Los recursos son archivos externos usados en su mayoría para dar una mejor presentación a una aplicación Android. Entre los archivos que se soportan se incluyen XML, PNG, JPEG. El archivo XML tiene muchos formatos diferentes dependiendo en qué se escriba; por ejemplo se pueden crear layouts, menus, strings en este formato.

Directorio res: Contiene todos los recursos necesarios para el proyecto: imágenes, strings, estilos, etc; los mismos que se organizan de la siguiente manera:

Subdirectorio drawable: Almacena los archivos del tipo imagen (jpg, png) y shape. Existen tres tipos ldpi, mdpi,, hdpi que se usarán dependiendo de la resolución del dispositivo.

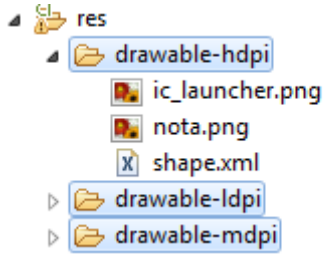

Fig. 28. Directorio de recursos

Subdirectorio layout: Contiene todos los ficheros que componen la interfaz gráfica de la aplicación.

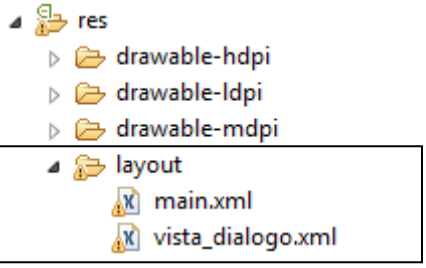

Fig. 29. Directorio de recursos - Vistas

Subdirectorio values: Almacena todos los ficheros relacionados a las cadenas de texto y estilos utilizados en la aplicación.

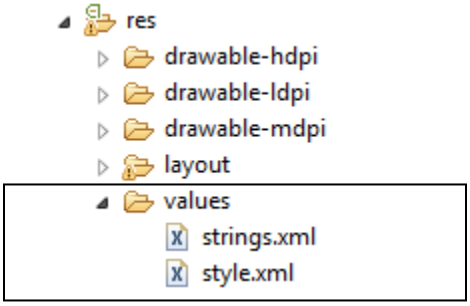

Fig. 30. Directorio de recursos - Valores

3. Carpeta gen

Contiene una serie de elementos de código generados automáticamente al compilar el proyecto. Cada vez que se genera el proyecto, la maquinaria de compilación de Android genera una serie de ficheros fuente en java, dirigidos al control de los recursos de la aplicación, como se muestra en la figura 26.

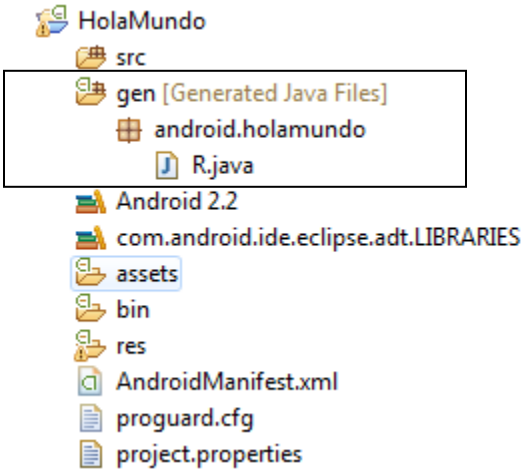

Fig. 31. Directorio gen

4. Clase R.java

Es uno de los archivos, ya que contiene código generado automáticamente por el proyecto. Esta clase R de detallan una serie de constantes con los ID de todos los recursos de la aplicación incluidos en la carpeta /r es/, de forma que podamos acceder fácilmente a estos recursos desde nuestro código a través de este dato.

5. Archivo de configuración AndroidManifest.xml

Todo proyecto en Android incluye un archivo AndroidManifest.xml, que almacena información relevante del proyecto. Este archivo permite definir la estructura e ingresar datos de las aplicaciones, y de los componentes. El sistema de información debe tener antes de que pueda ejecutar cualquiera de código de la aplicación. Entre otras cosas, el manifiesto hace lo siguiente (Android Developers, 2012):

- Se nombra el paquete de Java para la aplicación. El nombre del paquete sirve como un identificador único para la aplicación.

- Describe los componentes de la aplicación - las actividades, servicios, receptores de radiodifusión, y proveedores de contenido que la aplicación se compone. Además de las clases que implementan cada uno de los componentes y sus capacidades.

- Se declaran los permisos de la aplicación debe tener para acceder a las partes protegidas de la API e interactuar con otras aplicaciones.

- También declara que los permisos que los demás están obligados a tener a fin de interactuar con los componentes de la aplicación.

- Se declara el nivel mínimo de la API de Android que requiere la aplicación.

- En él se enumeran las bibliotecas que la aplicación debe estar vinculado en contra.

ANEXO B

Desarrollo de Aplicaciones Android

1. Creación de un nuevo proyecto Android

Para crear un nuevo proyecto Android en Eclipse, se deben seguir los siguientes pasos:

1. Seleccionar Archivo → Nuevo → "Android Project" (fig. 27).

2. En el cuadro de diálogo (fig. 28), inicialmente se establece los detalles del nuevo proyecto. El "Nombre del proyecto".

3. Luego se selecciona la versión instalada del "Build Target" (fig. 29).

4. En la vista final del cuadro (fig. 30) especificamos "Nombre de la Aplicación", "Nombre del paquete", "Nombre de la actividad", y escogemos el SKD mínimo (se selecciona un por defecto).

5. Cuando se haya introducido los datos, se hace clic en Finalizar.

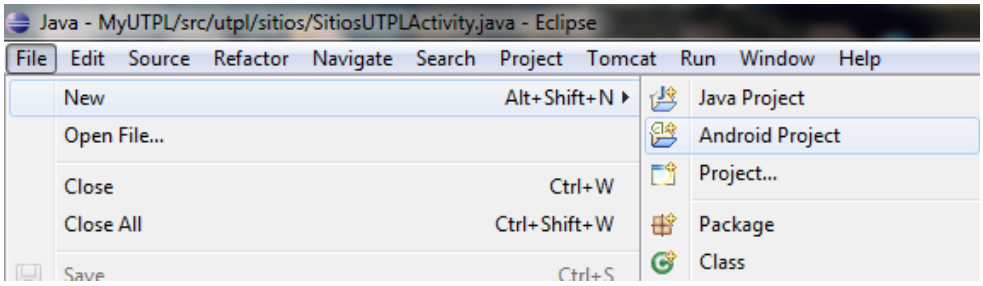

Fig. 32. Creando nuevo Proyecto

Luego de realizar el proceso, se generan automáticamente los archivos y recursos (fig. 31) de la primera aplicación.

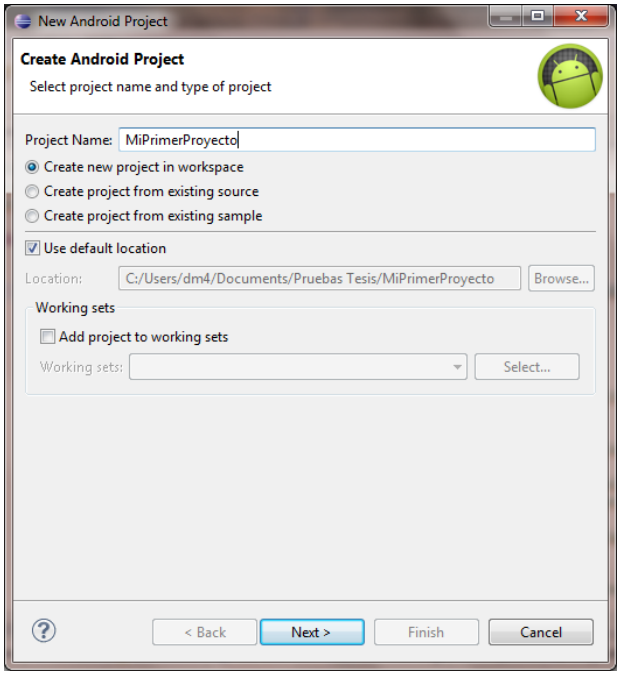

Fig. 33. Seleccionar nombre y tipo de proyecto

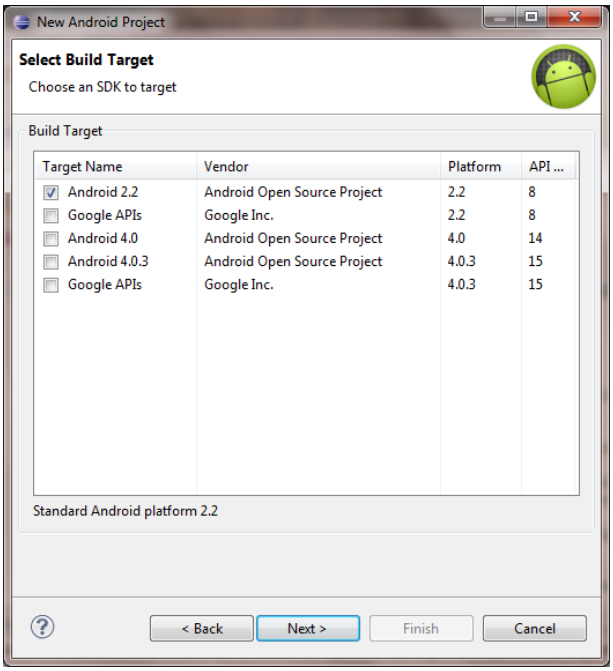

Fig. 34. Seleccionar versión Android

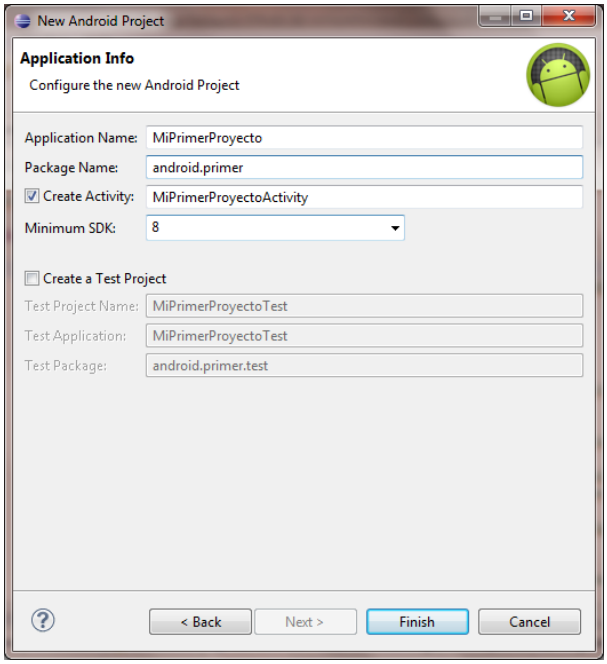

Fig. 35. Configurar información del proyecto

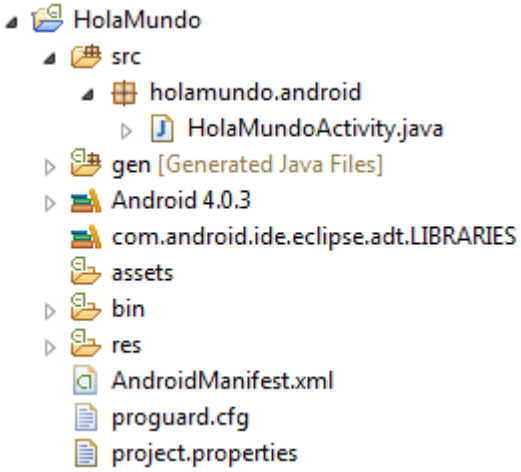

Fig. 36. Proyecto Android creado

Configuración inicial

Las configuraciones de inicio permiten establecer algunos parámetros como el tiempo de ejecución y depurar aplicaciones. El uso de una configuración inicial puede especificar lo siguiente:

- El proyecto y la actividad de lanzamiento.
- Las opciones de emulador a usar.

- Entrada / salida de la configuración (incluido por defecto de la consola).

Se puede especificar diferentes configuraciones iníciales de confianza para ejecutar. Los pasos siguientes muestran cómo crear una configuración inicial para una aplicación de Android:

1. Seleccionar el proyecto y hacer clic derecho.

2. Seleccionar la opción “Run as”, seguidamente de “Run configurations”.

3. En el cuadro de diálogo pestaña “Android” (fig. 32) seleccionar la activity inicial, caso contrario se ejecutará la que viene por defecto.

4. En la pestaña target (fig. 33), es posible seleccionar el AVD para ejecutar nuestra aplicación.

5. Por último, se establece las propiedades adicionales en la pestaña “Common”.

6. Se hacer clic en aplicar, y la configuración de inicialización se guardará.

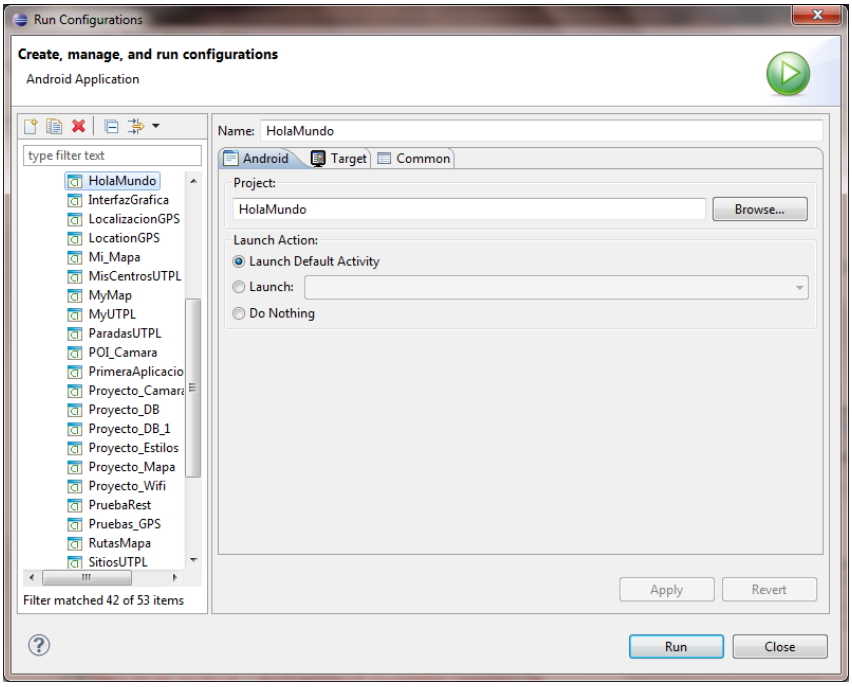

Fig. 37. Configuración de Ejecución – Seleccionar Activity

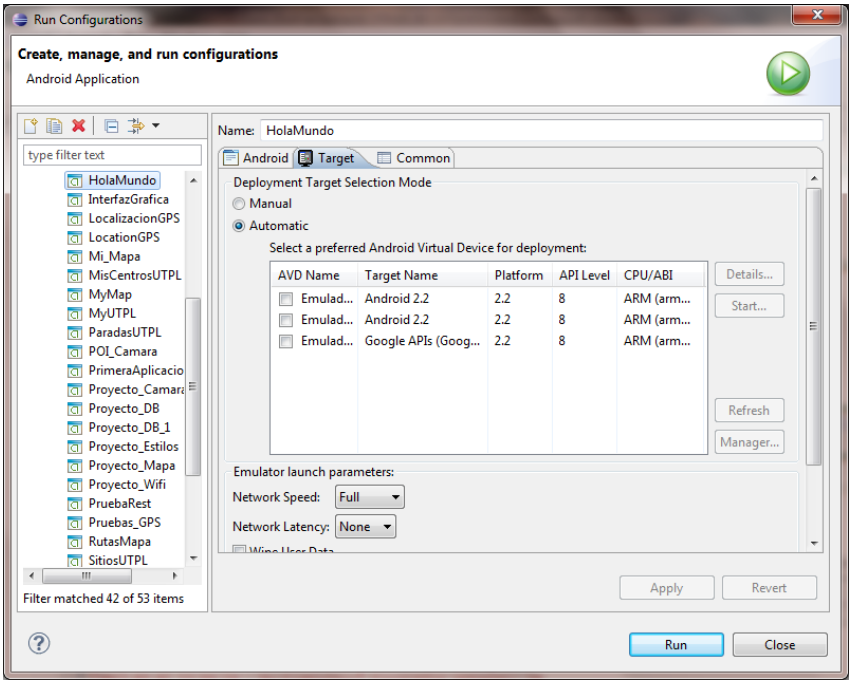

Fig. 38. Configuración de Ejecución - Seleccionar el emulador

Ejecución de la Aplicación

Una forma sencilla de ejecutar una aplicación Android se resume en los siguientes pasos:

1. Seleccionar el proyecto y hacer clic derecho.

2. Seleccionar la opción “Run as” (fig. 34), seguidamente de “Android Application”.

3. Empieza a cargar la aplicación en el emulador (fig. 35).

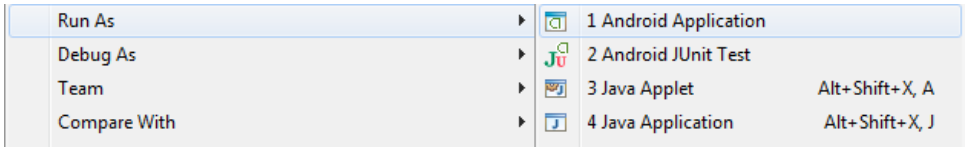

Fig. 39. Ejecutar aplicación Android

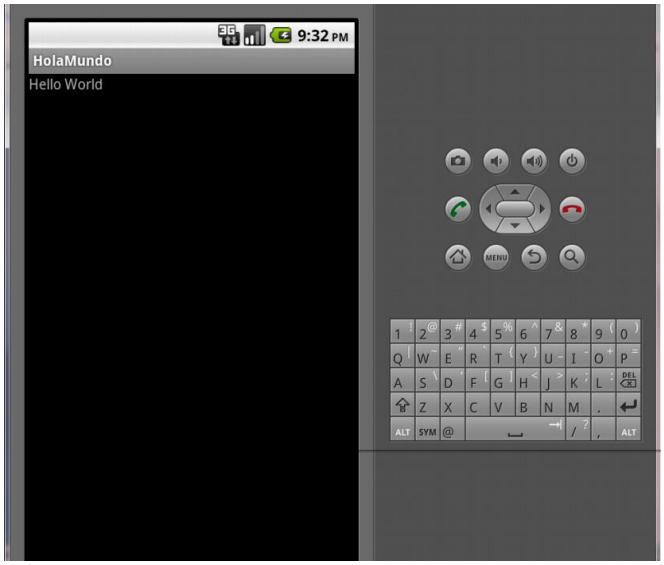

Fig. 40. Aplicación ejecutada

La fase ejecución de una aplicación Android en Eclipse sigue una serie de procesos que se describen a continuación:

1. Compila el proyecto actual y lo convierte en un archivo ejecutable Android (.dex).
2. Se crean los paquetes de los recursos ejecutables y externos en un paquete de Android (.apk).
3. Inicia el emulador.
4. Instala la aplicación en el emulador.
5. Inicia la aplicación (fig. 35).

2. Creación de una Actividad (Activity)

Todo proyecto Android genera una clase principal del tipo Activity donde se define la interfaz de usuario e implementa su funcionalidad. La estructura básica de una nueva actividad se muestra a continuación:

```
package android.holamundo;

import android.app.Activity;

public class HolaMundoActivity extends Activity {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

    }
}
```

Como se puede observar, el presente es un activity vacío. Sin embargo al llamar a una vista se establece el contenido para la aplicación; por tal motivo la primera cosa que se debe hacer es crear la interfaz de usuario con vistas y diseños.

Las vistas o layouts definen la interfaz de usuario que permiten presentar datos y proporcionar interacción con el usuario. Android proporciona varias clases de diseño, llamada ViewGroups, que puede contener múltiples puntos de vista para ayudar a diseñar las interfaces de usuario. Para asignar una interfaz de usuario de un Activity, se llama al método setContentView de onCreate de su Activity.

3. Creación de una Interfaz de Usuario

En toda aplicación es fundamental la creación de interfaces de usuario atractivas e intuitivas para el usuario final; asegurando su facilidad de uso.

Los elementos de una interfaz de usuario para Android están dispuestos en la pantalla por medio de una variedad de controladores de distribución derivados de ViewGroup. El uso correcto de los diseños es esencial para la creación de interfaces (Android Developers, 2012).

Capas o Layouts

Los Layouts son elementos no visuales destinados a controlar la distribución, posición y dimensión de los controles que se insertan en su interior. Estos componentes extienden a la clase base ViewGroup, como muchos otros componentes contenedores, es decir, capaces de contener a otros controles. Dentro de los layouts más conocidos se encuentran:

Frame Layout

En un Frame Layout sus controles hijos se alinean a su esquina superior izquierda, de forma que cada control queda oculto por el control siguiente (fig. 36). Por ello, suele utilizarse para mostrar un único control en su interior, a modo de contenedor (placeholder) sencillo para un sólo elemento sustituible, por ejemplo una imagen (Android Developers, 2012).

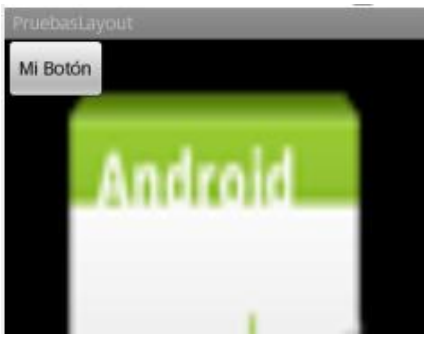

Fig. 41. Ejemplo FrameLayout sencillo

LinearLayout

Este Layout apila uno tras otro todos sus elementos hijos de forma horizontal o vertical según se establezca su propiedad de orientación (android:orientation).

A pesar de la simple vista resulta ser lo suficiente versátil como para ser de utilidad en muchas ocasiones. Se puede ver un ejemplo de un LinearLayout en la figura 37.

Fig. 42. Layout vertical y horizontal

Table Layout

Un TableLayout permite distribuir sus elementos hijos de forma tabular, definiendo las filas y columnas necesarias, además la posición de cada componente dentro de la tabla (fig. 38).

La estructura de la tabla se define de forma similar al código HTML, es decir, indicando las filas que compondrán la tabla (objetos TableRow), y dentro de cada fila las columnas necesarias, sin embargo no existe ningún objeto especial para definir una columna (similar a un TableColumn) sino que directamente se inserta los controles necesarios dentro del TableRow y cada componente insertado corresponderá a una columna de la tabla (Android Developers, 2012).

Fig. 43. Ejemplo TableLayout

RelativeLayout

Este Layout permite especificar la posición de cada elemento de forma relativa a su elemento padre o a cualquier otro elemento incluido en el propio Layout (Android Developers, 2012). De esta forma, al incluir un nuevo elemento X se podrá indicar por ejemplo que se debe colocar debajo del elemento Y alineado a la derecha del Layout padre.

En un Relative Layout se tendrá un sinfín de propiedades para colocar cada control justo donde se requiere como se muestra en la figura 39.

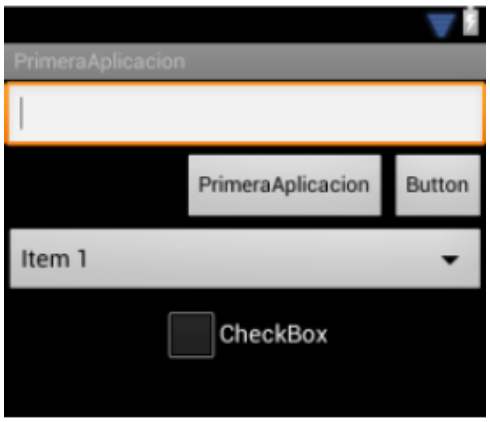

Fig. 44. Ejemplo Relative Layout

4. Comunicación entre Intents

Los Intents son utilizados como un mecanismo de paso de mensajes que funciona tanto en su aplicación, y entre otras aplicaciones. Los Intents se pueden utilizar para:

- Declarar su intención de que una actividad o servicio comenzó a realizar una acción, por lo general con una parte de los datos.
- Difusión que un evento se ha producido.
- Explícitamente iniciar un servicio o actividad.

Se usa Intents para apoyar la interacción entre cualquiera de los componentes de la aplicación instalada en un dispositivo Android, independientemente de la aplicación.

Esto convierte el dispositivo desde una plataforma que contiene una colección de componentes independientes en un único sistema interconectado.

Uno de los usos más comunes de Intents es iniciar nuevas actividades, ya sea explícitamente (mediante la especificación de la clase de carga) o implícitamente (Android Developers, 2012).

En este último caso, la acción no tiene que ser realizada por una actividad dentro de la aplicación que llama.

Ejemplo: Paso de mensajes entre intents

Luego de crear una Aplicación común. Se deberá crear una segunda actividad y luego agregar dentro del archivo androidmanifest.xml una nueva actividad de ésta manera:

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

El .mensaje es la nueva actividad creada.

En el evento del botón de la actividad principal se coloca lo siguiente:

```
//crea hacia donde se va a dirigir la nueva actividad
  Intent i = new Intent(IntentsActivity.this, mensaje.class);
//Como vamos a pasar un mensaje a otro intent lo hacemos con una key
  i.putExtra("key", TextoMensaje.getText().toString());
//empieza la actividad
  startActivity(i);
```

En el evento onCreate de la actividad secundaria mensaje.java se escribe lo siguiente:

```
mostrar = (TextView)findViewById(R.id.txtMensaje);
// Texto que toma los valores con la clave pasada de la otra actividad
String text = this.getIntent().getExtras().getString("key");
//Muestra el texto que se a pasado
mostrar.setText(text);
```

El objeto de la aplicación es enviar un mensaje desde la actividad principal a la secundaria (fig. 40):

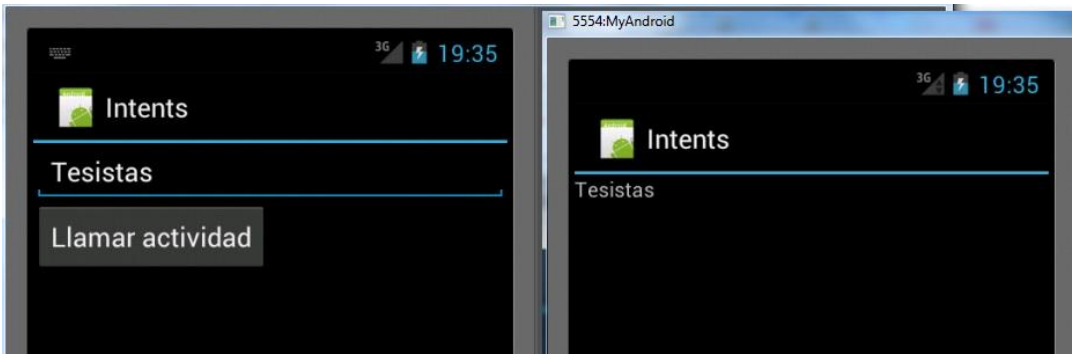

Fig. 45. Comunicación entre Intents

5. Aplicación utilizando el GPS

En la presente aplicación se mostrará cómo utilizar el GPS integrado de un dispositivo móvil Android. Esto nos permitirá obtener nuestra ubicación en tiepo real, teniendo un margen de error mínimo que se traduce en unos cuantos metros en relación al lugar en el que realmente nos encontramos. Todo esto basado en las coordenadas de latitud y longitud.

Para ello creamos un proyecto nuevo, seguidamente se debe editar el archivo main.xml, y agregar los controles necesarios para la aplicación, en este caso un botón y dos textview para mostrar los resultados (fig. 41).

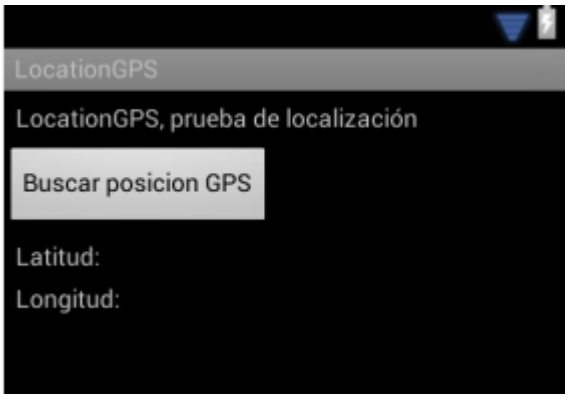

Fig. 46. Interfaz Aplicación GPS

Seguidamente se debe establecer **permisos** de usuario en el archivo Manifest.xml para servicios de Localización:

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

Dentro del Activity se deben definir algunos objetos y clases, entre ellos el objeto **LocationManager**, al cual asignaremos un escuchador (LocationListener) para que nos informe cada vez que cambia el estado del GPS mediante el método onLocationChanged de la clase MyLocationListener que implementa el escuchador LocationListener.

```
Prívate class MyLocationListener implements LocationListener{

//Método Sobrescrito

public void onLocationChanged(Location loc) {
            if (loc != null) {
//Para presentar mediante un msj instantaneo si la señal gps fue recibida
                Toast.makeText(getBaseContext(),
                    getResources().getString(R.string.gps_signal_found),
                    Toast.LENGTH_LONG).show();
                setCurrentLocation(loc);
```

```
                handler.sendEmptyMessage(0);
            }
 }
}
```

Se debe rescribir el método **run** dentro de la clase. Aquí se utilizan algunos objetos como el LocationManager, LocationListener y MyLocationListerner de la clase privada. Aquí utilizamos un proveedor denominado GPS_PROVIDER que nos permite obtener la posición actual. Además de trabajar con Threads que usan un objeto Looper, con las funciones prepare() para iniciar el bucle de threads y loop() al final.

```
public void run() {
            //Estableciendo un valor obtenido desde el Sistema
            mLocationManager =
(LocationManager)getSystemService(Context.LOCATION_SERVICE);

            //Comprueba Si el proveedor de servicios de Localización esta en estado
activado
            //GPS_PROVIDER = PROVEEDOR

            if (mLocationManager.isProviderEnabled(LocationManager.GPS_PROVIDER)) {

                  //Inicializa un bucle
                  Looper.prepare();

                  //Se crea una nueva instancia de la clase MyLocationListener
                  mLocationListener = new MyLocationListener();

                  //Registra la actividad actual y notifica periódicamente en base
a los
                  //parámetros especificados como el proveedor y el escuchador
                  mLocationManager.requestLocationUpdates(
                     LocationManager.GPS_PROVIDER, 0, 0, mLocationListener);

                  //Ejecuta la cola de msjs dentro del bucle
```

```
                Looper.loop();
                //Finaliza el bucle
                Looper.myLooper().quit();

            } else {

                //Muestra un msj instantaneo si no se han recibido señales gps
            Toast.makeText(getBaseContext(),
                getResources().getString(R.string.gps_signal_not_found),
                Toast.LENGTH_LONG).show();
            }
        }
```

También tenemos un objeto de tipo **Handler** al cual llamaremos cuando tengamos la localización de nuestro GPS y este se encargará de cerrar la ventana en proceso y establece los valores de longitud y latitud en el TextView.

```
private Handler handler = new Handler() {
            @Override
            public void handleMessage(Message msg) {

            //Cerrando el ProcessDialog
                pd.dismiss();
            //Remueve el registro actual de actualizaciones de ubicación de la
                // actividad actual con el LocationListener dado.
                mLocationManager.removeUpdates(mLocationListener);

                //Si currentLocation tiene un valor
            if (currentLocation!=null) {

                //Imprime los valores latitud y longitud
                outlat.setText("Latitude: " + currentLocation.getLatitude());
                outlong.setText("Longitude: " + currentLocation.getLongitude());
            }
            }
        };
```

Finalmente se ha creado un método que permite lanzar un ProcessDialog y un Thread que permita iniciar el método run de la Aplicación.

```
private void escribirSenalGPS() {

      DialogInterface.OnCancelListener dialogCancel = new
DialogInterface.OnCancelListener() {

            //Método que permite cancelar la operación y cierra el ProcessDialog
           public void onCancel(DialogInterface dialog) {

            //Muestra un  msj instantaneo con el msj señal no encontrada
            Toast.makeText(getBaseContext(),
                      getResources().getString(R.string.gps_signal_not_found),
                      Toast.LENGTH_LONG).show();
               handler.sendEmptyMessage(0);
           }

       };

       //Lanza un ProcessDialog que estará activo mientras no haya una respuesta o
se termine el bucle
            pd = ProgressDialog.show(this,
this.getResources().getString(R.string.search),
                        this.getResources().getString(R.string.search_signal_gps),
true, true, dialogCancel);

            //Creando un nuevo thread (bucle)
            Thread thread = new Thread(this);

            //Iniciando y llamando al método run()
            thread.start();
    }
```

Como ultima parte debemos llamar a este método desde un botón. Nuestro resultado al iniciar el servicio GPS debe ser similar a la figura 42.

Fig. 47. Ejecución inicial de la Aplicación GPS

Para comprobar el funcionamiento de la aplicación desde el emulador se debe simular una señal de GPS, para ello Eclipse tiene un apartado denominado DDMS donde hay una opción donde se puede enviar la latitud y longitud al GPS.

Sin embargo se la puede hacer también a través de la consola de Windows. Primero se realiza una conexión con el emulador mediante el comando:

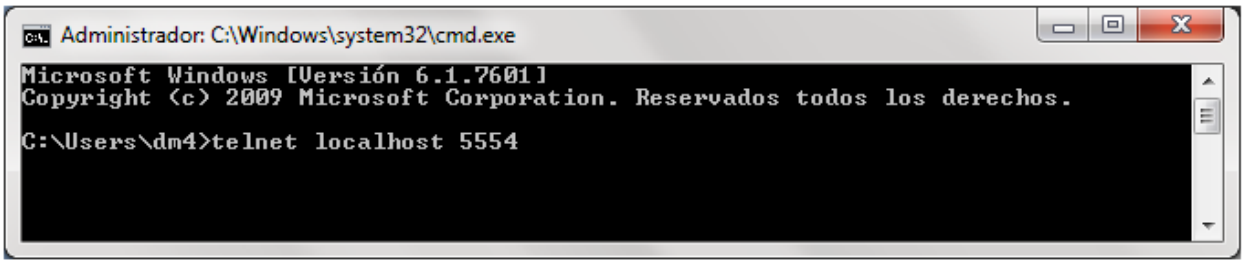

Fig. 48. Conexión telnet con el emulador local

Y para enviar las coordenadas el comando geo fix (fig. 44):

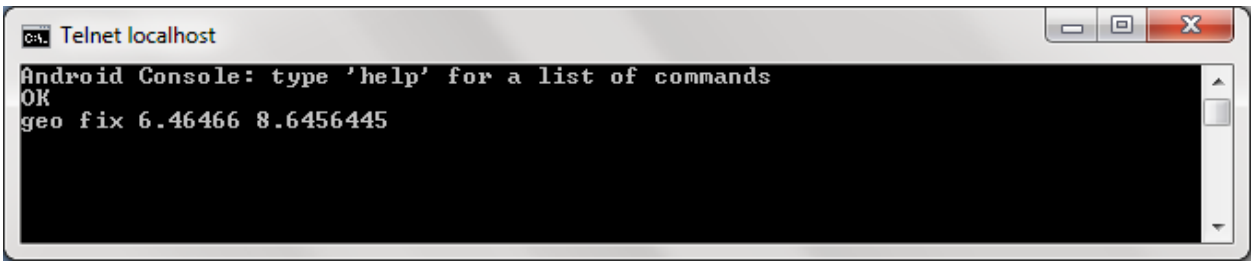

Fig. 49. Envío de señales GPS (latitud y longitud)

Donde el primer valor es la latitud y el segundo la longitud.

La aplicación desarrollada escuchará esta simulación de coordenadas y nos mostrará las señales enviadas (fig. 45).

Fig. 50. Resultados de la Aplicación GPS

En dispositivos móviles luego de que se ejecuta nos devolverá la ubicación actual, siempre y cuando estén activos los servicios de GPS y en correcto funcionamiento.

6. Aplicación con Mapas y OverLays

Antes de iniciar el desarrollo se debe descargar el Google API para el SDK, así como crear un AVD para ejecutar aplicaciones de este tipo, en caso de no contar con dispositivo real; los dispositivos reales fácilmente ejecutan aplicaciones que usen el API. Para realizar uso del API se debe conseguir su API Key, para el uso del control Mapview que se describirá luego (Ver Generación del API Key para usar el servicio de Mapas).

Google facilita el uso de sus servicios dentro de nuestras aplicaciones Android. Sin embargo estas no pueden ser para uso comercial.

Antes de iniciar a desarrollar, es necesario configurar previamente el entorno (Ver Configuración de Eclipse para el uso de mapas). En esta aplicación se hará uso de un nuevo control conocido como MapView que permite gestionar un mapa dentro del layout de nuestra aplicación. Sin embargo, puesto que Eclipse no lo incluye en la vista Graphical Layout, deberemos editar directamente el fichero main.xml, tal y como se muestra a continuación:

```
<?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">

<com.google.android.maps.MapView
    android:id="@+id/mapa"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:apiKey="0FjufuKaAsrOCnjEWl09NXkqeOWqSdocNk6EuoA"
    android:clickable="true" />

</LinearLayout>
```

La propiedad apiKey se establece con la clave que hemos obtenido anteriormente, y añadimos además la propiedad clickable para poder interactuar con el mapa (por ejemplo desplazarnos).

Por otra parte, la clase MapView también incluye los siguientes métodos para controlar el tipo de mapa que queramos mostrar.

- setSatellite(boolean on)
- setStreetView(boolean on)
- setTraffic(boolean on)

Podemos igualmente consultar el tipo de mapa que en ese momento está desplegado mediante los siguientes métodos:

- isSatellite()
- isStreetView()
- isTraffic()

Para gestionar el MapView desde código fuente, se crea una instancia y luego se establecen los controles:

```
mapa = (MapView)findViewById(R.id.mapa);
mapa.setBuiltInZoomControls(true);
```

Seguidamente se deben agregar algunos permisos (fig. 46) y librerías (fig. 47) en el AndroidManifest.xml, así:

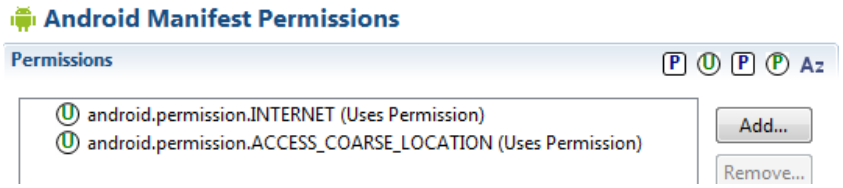

Fig. 51. Permisos de usuario agregados

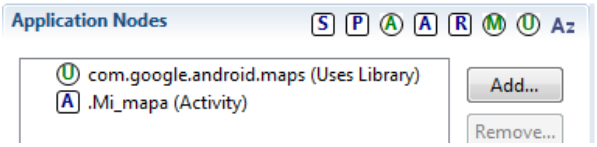

Fig. 52. Librería a utilizar para Mapas.

Finalmente al ejecutar la aplicación se obtendrá algo similar a la figura 48.

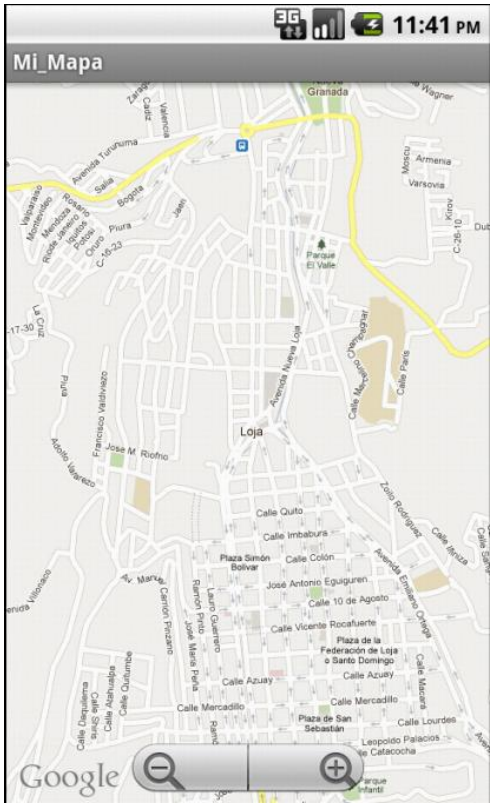

Fig. 53. Vista de un MapView

Configuración de Eclipse para el uso de mapas

En primer lugar deberemos asegurarnos de tener instalado el paquete de mapas en Eclipse que nos permitirá interactuar con los mismos. El nombre habitual de este paquete será: Google APIs by Google, Android API x, revisión y ´para comprobar si lo tenemos correctamente instalado, y en caso negativo poder instalarlo debemos acceder al Android SDK and AVD Manager y observar si aparece en la sección de paquetes instalados, tal y como se observa en la figura 49.

Por otra parte, si queremos probar la aplicación en el emulador tenemos también que crear un nuevo AVD que utilice el paquete en cuestión como \target", tal y como podemos ver en la figura 50.

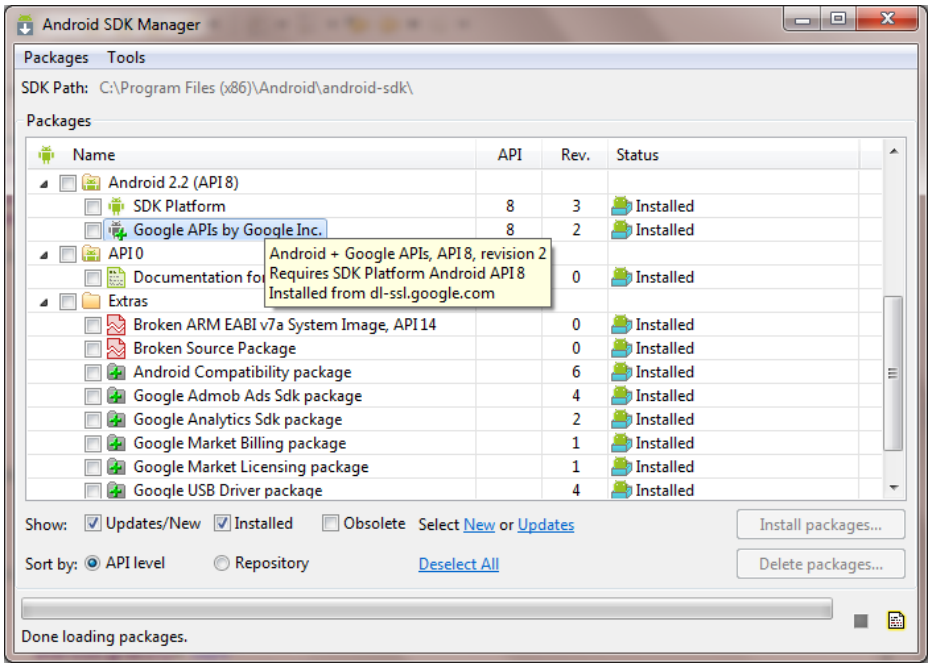

Fig. 54. Google API instalada - Android SDK Manager

Antes de finalizar, hay que recordar que cuando se crea un proyecto de este tipo, se debe configurar el target a Google API, en las propiedades del mismo, tal y como podemos comprobar en la figura 51.

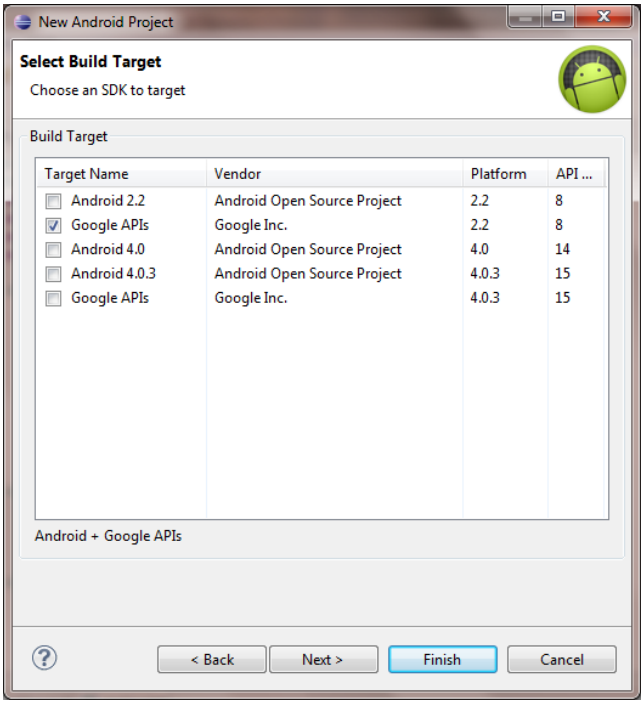

Fig. 55. Selección Target del Proyecto

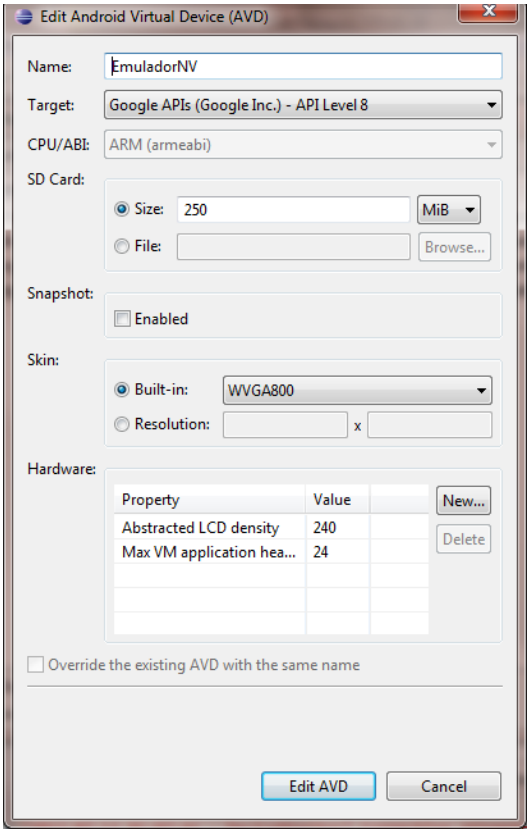

Fig. 56. Creando un AVD usando Google API 2.2

Generación del API Key para usar el servicio de Mapas

Android exige que toda aplicación instalada en un dispositivo (y que por tanto se pueda comercializar en Android Market) esté armada digitalmente con un certificado digital cuya clave privada sea propiedad del desarrollador. De esta forma se consigue que el autor de la aplicación esté correctamente identificado.

Entonces, para poder hacer uso del Android Maps API necesitamos que Google nos proporcione una clave para dicho API, la cual estará asociada al certificado con el que armaremos digitalmente nuestras aplicaciones que utilicen mapas.

Para solicitar dicha clave debemos acceder a la siguiente dirección web: http://code.google.com/intl/es-ES/android/maps-api-signup.html

Como podemos comprobar, para obtener dicha clave necesitamos proporcionarle a Google la huella digital de nuestro certificado. Para ello localizaremos el archivo donde se almacenan los datos del certificado de depuración creado por defecto, llamado debug.keystore.

A su vez, para conocer la localización de dicho certificado, abriremos la ventana Windows Preferences de Eclipse, y dentro del apartado Android Build encontraremos la ruta del certificado en cuestión, tal y como se puede comprobar en la figura 52.

Una vez que se conoce la ruta del certificado de depuración por defecto, luego se debe obtener su huella digital a través del siguiente comando, en la línea de comandos:

```
keytool -list -alias androiddebugkey -keystore “Ruta del keystore”
-storepass android -keypass android
```

Finalmente se debe copiar dicha huella digital para el uso de un MapView descrito a continuación.

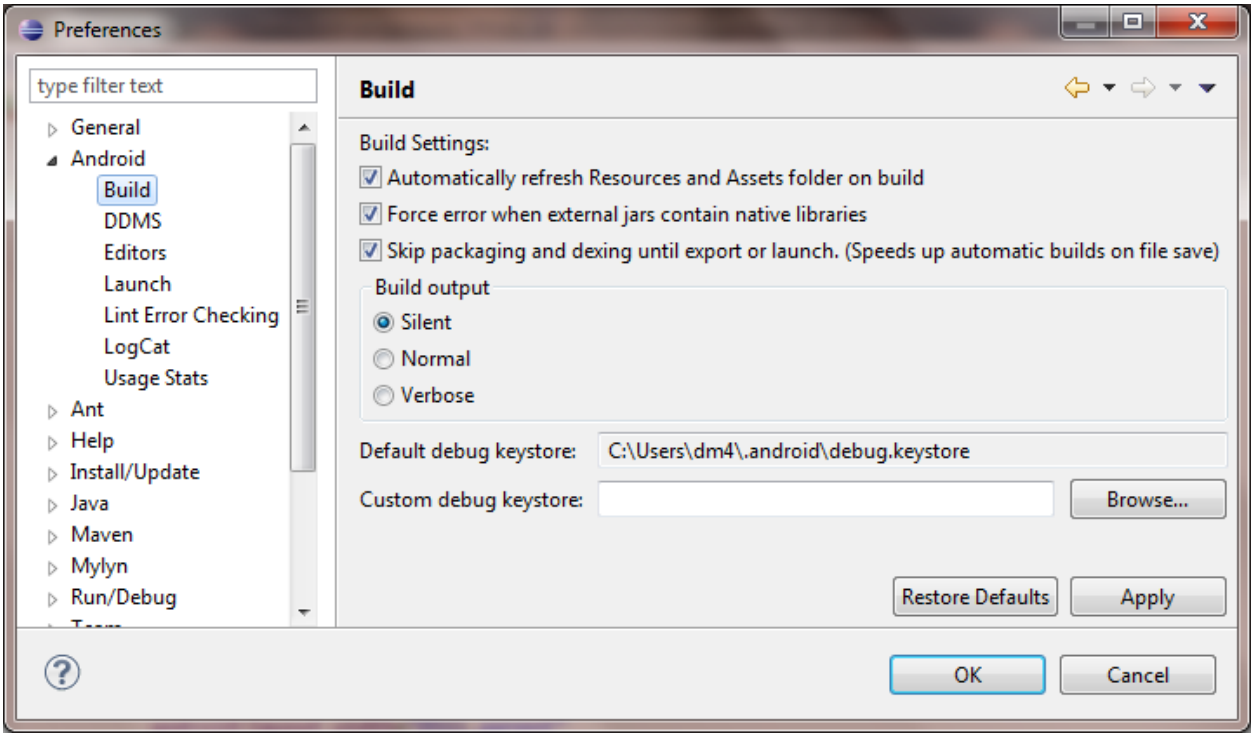

Fig. 57. Localización del archivo debug.keystore

6.1. Creación de un OverLay

Una clase de tipo Overlay permite sobreponer capas encima de un mapa determinado. Esto resulta muy útil cuando es necesario colocar un marcador sobre el mapa que indique por ejemplo la posición de un POI.

Para ello se debe crear una propia clase que herede de Overlay, para luego sobrescribir su método draw(), tal y como se observa a continuación.

```
public class MyOverlay extends Overlay {

	private GeoPoint point;
	private Resources resources;

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

```
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(resources, R.drawable.utpl);
        canvas.drawBitmap(marker, scrnPoint.x - marker.getWidth() / 2,
        scrnPoint.y - marker.getHeight() / 2, null);
        return true;
}
```

Como se puede observar, el cometido del método draw() no es otro que el de convertir el punto que representa la localización del dispositivo mediante la clase GeoPoint() en un punto en el mapa. A partir de los métodos getProjection() y toPixels() se crea un marcador a partir de un recurso (utpl.png) de la aplicación y se dibuja éste en el punto del mapa obtenido anteriormente.

Para dibujar el marcador se debe llamar desde el Activity principal de la siguiente manera:

```
        GeoPoint point = new GeoPoint((int) (location.getLatitude() * 1E6),
        (int) (location.getLongitude() * 1E6));
        mapController.animateTo(point);
        mapController.setZoom(15);
        List<Overlay> mapOverlays = mapView.getOverlays();
        MyOverlay marker = new MyOverlay(point, getResources());
        mapOverlays.add(marker);
        mapView.invalidate();
```

El resultado final de la aplicación se presenta en la figura 53.

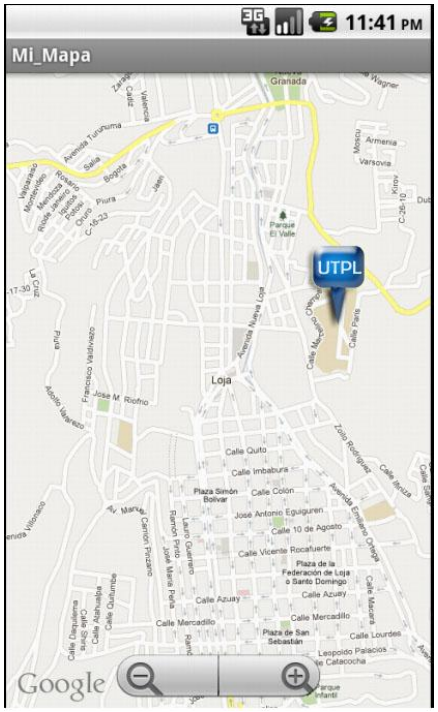

Fig. 58. Ejemplo de un Overlay sobre un MapView

7. Aplicación que consume un WebService

Construir servicios web es una forma de reutilizar código para múltiples clientes. Sin importar la plataforma o tipo de equipo que desee utilizar los mismos. Debido al gran auge de la web se ha creído conveniente utilizar una estructura cliente servidor que permita consumir servicios Restful desde el dispositivo móvil.

Antes de iniciar con el desarrollo de la aplicación móvil se debe contar con el webservice Restful. Para su construcción existen muchos manuales y tutoriales en línea. Para las pruebas realizadas se construyó un Servicio que consulta un dato según un código enviado como parámetro. El webservice reporta sus datos de respuesta en formato xml. A continuación se describen los pasos a seguir para el desarrollo de la aplicación.

1. Crear un nuevo proyecto Android.

2. Crear la interfaz principal (fig. 54).

Fig. 59. Aplicación Consumo Servicios Rest

3. Editar el archivo de Manifiesto, agregando la siguiente línea:

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

4. En el Activity principal escribimos las siguientes constantes globales:

```
static String HOST = "192.168.1.4";
static int PORT = 8080;
static String URL= "/MisServiciosRest/resources/buscarDato?cod=";
```

5. Dentro del botón se codifica lo siguiente:

```
HttpClient httpClient = new DefaultHttpClient();
HttpGet httpGet = new HttpGet("http://"+HOST+":"+PORT+"/"+URL+
parametro);
DocumentBuilder builder;
try {

ResponseHandler<String> responseHandler = new
BasicResponseHandler();
String responseBody = httpClient.execute(httpGet,
responseHandler);
builder =
DocumentBuilderFactory.newInstance().newDocumentBuilder();
```

```
            Document doc=builder.parse(new InputSource(new
            StringReader(responseBody)));
            String dato =
            Integer.parseInt(doc.getElementsByTagName("dato").item(0).getText
            Content());
            txt2.setText(dato); //Caja de texto para mostrar el resultado

            } catch (ClientProtocolException e) {
                  Log.e(getString(R.string.app_name), e.getMessage());
            } catch (IOException e) {
                  Log.e(getString(R.string.app_name), e.getMessage());
            } catch (SAXException e) {
                  Log.e(getString(R.string.app_name), e.getMessage());
            } catch (ParserConfigurationException e) {
                  Log.e(getString(R.string.app_name), e.getMessage());
            } catch (FactoryConfigurationError e) {
                  Log.e(getString(R.string.app_name), e.getMessage());
            }
      }
```

Primeramente se crea una conexión para el cliente móvil (HttpClient), luego un objeto HttpGet para iniciar la solicitud de tipo get, así como DocumentBuilder para manejar posteriormente el documento obtenido del WebService.

6. Ejecutamos la aplicación y tendremos algo similar a la figura 55.

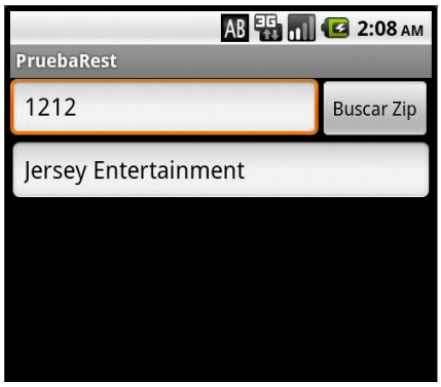

Fig. 60. Resultados Aplicación consumiendo RestFul Services

ANEXO C

Aplicación Móvil Android de Realidad Aumentada y Geolocalización para la UTPL

Rodrigo Saraguro[1], Nelson Piedra[1]

[1] Tecnologías Avanzadas de la Web y Sistemas Basados en el Conocimiento
Departamento de Ciencias de la Computación y Electrónica
Universidad Técnica Particular de Loja
1101608 San Cayetano Alto S/N (Loja-Ecuador)
{rasaraguro, nopiedra}@utpl.edu.ec

Resumen. La Universidad Técnica Particular de Loja, es una institución de educación superior del Ecuador que de forma regular recibe la visita de estudiantes, profesionales, empresarios y turistas; ante esto, existe la necesidad de proveer un mecanismo que brinde información oportuna; para ello se ha desarrollado una aplicación que permita reconocer las instalaciones del campus universitario a través de un dispositivo con SO Android, lo que ayudaría en gran medida a visitantes y estudiantes cuando requieran asistir a un evento, conocer un centro universitario cercano, una parada de bus UTPL o sitios importantes de la ciudad. El estudio inicia con una introducción, el estado del arte referente al tema de realidad aumentada, geolocalización, plataforma Android, así como proyectos similares. Antes de iniciar la fase de desarrollo se menciona el objetivo de cada servicio, para luego continuar con la arquitectura, componentes del proyecto, interfaces, servicios y APIs utilizadas. En la parte final se detallan los resultados obtenidos y conclusiones.

Palabras clave: Android, móvil, realidad aumentada, geolocalización, POI, UTPL.

1 Introducción

La Universidad Técnica Particular de Loja, es una institución de educación superior del Ecuador que ofrece servicios de tercer y cuarto nivel a más de 30000 estudiantes; cuenta con cerca de 80 centros universitarios en el territorio nacional así como en el extranjero; cada ciclo académico acoge un gran número de estudiantes de la Modalidad Abierta, ponentes, participantes de eventos, así como turistas. Los mismos que en muchos casos necesitan trasladarse de un punto especifico a otro. Sin embargo no existe algún medio o herramienta tecnológica que facilite esta información. Por tal motivo se ha creído conveniente el desarrollo de una aplicación basada en Realidad Aumentada y Geolocalización que permita recorrer las instalaciones del campus UTPL a través de un smatphone con SO Android. Con lo cual se pueda obtener información al alcance de

manera oportuna, así se agilizarían muchos procesos, además que se brindaría una mejor visita.

La Realidad Aumentada es una de las tecnologías más llamativas que permite implementar estos servicios para dispositivos móviles, ya que combina elementos reales y virtuales para facilitar nuestra visión del mundo y cambiar la forma de acceder a la información. Con potentes dispositivos móviles es sencillo navegar por nuestro entorno visual y obtener mágicamente más información sobre el mismo, por ejemplo: la altura de un edificio, los departamentos que puede tener, personas importantes que trabajan en él, la distancia al lugar más cercano, tweets que se están realizando en el campus, lista de eventos, etcétera.

2 Estado del Arte

2.1 Realidad aumentada

La realidad aumentada es el término usado para definir un tipo de tecnología donde la visión de la realidad se amplía con elementos virtuales que añaden información digital. Así definimos esta tecnología como un punto intermedio entre la realidad tal y como a conocemos y la realidad virtual. [1]

Un sistema de realidad aumentada cuenta con las siguientes propiedades: combina objetos reales y virtuales, funciona en tiempo real, y se registra en tres dimensiones. Ya que la información virtual añadida normalmente se registra en un lugar del espacio.

Dentro del campo de la RA, existen algunos tipos de reconocimiento con los que se trabaja, siendo esta tarea la parte más costosa de la realidad virtual, y dependiendo de la técnica se requerirá de un tipo de hardware u otro. Estos son: basado en marcadores que utilizan imágenes del entorno como referencias, basado en objetos donde se comparán con una base de datos de objetos según sea su forma para descubrir de qué objeto se trata, y el reconocimiento basado en localización (GPS), y de sistemas que reconozcan la orientación del dispositivo que trabajan en función de las coordenadas, entonces el dispositivo aproxima el objeto de acuerdo a su ángulo de visión, y su distancia. [2]

2.2 Geolocalización

Es la identificación de la posición geográfica en tiempo real de un objeto o persona, ya sea un dispositivo conectado a Internet, un teléfono móvil o cualquier otro aparato que sea posible rastrear [3]. Dicha localización puede ser en un plano de dos dimensiones (por ejemplo, Google Maps), como en un plano de tres dimensiones (GPS).

En los últimos años, diferentes tipos de tecnologías han apostado por la geolocalización, siendo extraordinario el auge de esta en las tecnologías móviles de última generación.

El sistema GPS (Sistema de Posicionamiento Global), funciona a través de una red de satélites y permite mediante un pequeño receptor conocer nuestra ubicación en cualquier parte del mundo, sin requerir pago o inscripción alguna ya que la señal es libre y se puede aplicar a nivel terreste, aéreo, o marítimo.[4]

Un Mapa Digital es un despliegue gráfico sobre información geográfica que permite entender las distancias, referencias y puntos importantes en relación a un lugar y permite mostrar a un usuario sus puntos de interés. Google Maps ofrece la capacidad de hacer acercamientos o alejamientos de un mapa virtual [5]. Permite desplegar información de manera fácil y económica.

2.3 Android

Android es un software para dispositivos móviles que incluye un sistema operativo, y una pila de software como aplicaciones, framework y middleware, todos juntos forman el sistema completo [6].El SDK de Android proporciona las herramientas y APIs necesarios para comenzar el desarrollo de aplicaciones en la plataforma Android usando el lenguaje de programación Java.

Android se caracteriza por cubrir muchas aéreas, como el desarrollo de aplicaciones, conectividad y medios de comunicación. Es por tal motivo que seguidamente se mencionan sus características más importantes [7]:

- Application Framework para reutilizar y sustituir componentes.
- Maquina Virtual Dalvik optimizada para dispositivos móviles.
- SQLite para almacenamiento de datos estructurados.
- Sensores de cámara, gps, redes 3G, wifi.
- Emulador de dispositivos.

2.4. Proyectos Relacionados

La realidad aumentada puede proporcionar grandes beneficios en cuanto al aprendizaje y la formación en campos tan diversos como el comercio, el ejército y la medicina.

Existen muchos proyectos universitarios que han implementado la Realidad Aumentada en sus aplicaciones, entre ellas se conocen: Aplicaciones para el tratamiento de fobias de insectos [8], Aplicación guía para visitas de lugares como el Edificio Histórico de la Universidad de Oviedo [9], Aplicación Realidad Aumentada en el Ámbito Universitario

[10] para proveer información sobre distintos eventos que ofrecen los distintos departamentos de la Universidad Nacional de la Plata como por ejemplo talleres, cursos, conferencias, etc.

3 UTPLAR

UTPLAR es el nombre de la aplicación de Realidad Aumentada para la Universidad Técnica Particular de Loja; cuyo objetivo principal es brindar información en tiempo real de los puntos de interés del campus, centros universitarios y servicios de transporte a través de Realidad Aumentada y Geolocalización en Mapas.

3.1 Funcionalidades

La aplicación de Realidad Aumentada y Geolocalización tendrá las siguientes funcionalidades:

Vista RA Campus UTPL: A través de la cámara y sensores GPS, se implementa la vista en RA para obtener datos en tiempo real de lo que se visualiza; por ejemplo información de POIs a través de íconos.

Mapa Sitios UTPL: Determinar la ubicación actual del usuario dentro del campus UTPL así como los POI dentro del campus en el mapa, para obtener datos de información, y una imagen del POI.

Paradas de Bus UTPL: Mostrar las paradas de bus más cercanas a la ubicación actual del usuario, así como la ruta para llegar a ella y una imagen del POI.

Centros UTPL: Dibujar los centros asociados de todo el país, así como visualizar información de un centro especifico.

Sitios Cercanos: Presentar los POI más importantes de la ciudad a través de RA y Mapas. Además de presentación de Datos de DBpedia.

Administración UTPLAR: Administración de datos del servicio. Insertar nuevos datos, actualizar, eliminar.

3.2 Arquitectura

Para el desarrollo de la aplicación se utilizará la arquitectura representada en la figura 1. Arquitectura basada en LOCWD-Mobile Architecture [11], una arquitectura distribuida con la finalidad de reducir tiempos de carga en el cliente móvil, agilizar procesos y aprovechar recursos de la web.

Componentes: La arquitectura implementada se puede observar en la figura 1; a continuación se mencionan sus componentes con respecto a la parte de Servidores, y APIS utilizadas:

Base de Datos: Se utiliza un servidor MySQL para el almacenamiento de información de los POI UTPL (sitios, paradas, centros a distancia), así como sus relaciones.

Servicios Web UTPLAR: Se utiliza un servidor web Glassfish para el montado del Web Service RestFUL; intercambio de datos en formato XML.

Servicio Google Maps: Servicio utilizado para visualizar los mapas dentro de la aplicación, similar al servicio web.

API Google Directions: El API Google Directions nos permitirá obtener los datos en formato XML sobre la ruta entre dos puntos.

Wikitude Server: El servidor del API Wikitude [12] para desarrolladores quien recibe cada uno de los datos necesarios para el servicio de Realidad Aumentada.

DBpedia: Versión Semántica del contenido de Wikipedia.

Cliente: Es la aplicación objetivo, sus componentes principales son:

Capa Presentación: Está compuesta por todos los layouts creados en el proyecto. Su principal objetivo es facilitar la interacción entre la app y el usuario.

Capa de Negocio: Constituida por cada Activity, y a su vez por cada uno de los controladores para manejar eventos dentro de la app. Su función principal es establecer comunicación entre la presentación y los datos.

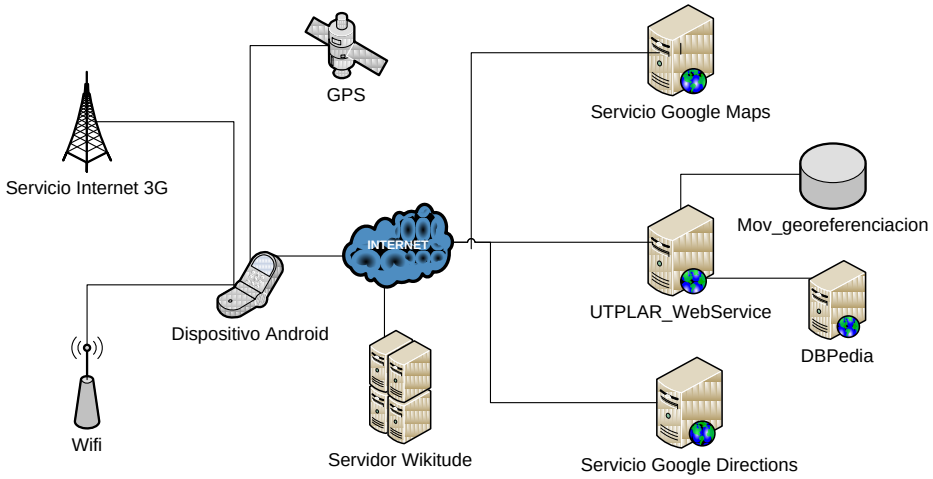

Fig. 1. Arquitectura Física UTPLAR

3.3 Desarrollo

La aplicación UTPLAR basada en la arquitectura antes descrita, fue desarrollada con la metodología ICONIX y los artefactos necesarios, herramientas y APIs de realidad aumentada y geolocalización, siendo estos temas la base fundamental sobre el cual se ha desarrollado la aplicación. A continuación se describen los componentes principales de la aplicación:

Realidad Aumentada UTPL: La funcionalidad de vista de realidad aumentada carga el navegador de Wikitude[12] para visualizar íconos representativos de cada sitio UTPL como se puede ver en la figura 2. Para conocer más a detalle el sitio se debe pulsar sobre el ícono, además se implementa una flecha que indica la dirección del sitio, así como la distancia hacia el mismo lugar.

Sitios UTPL: Al pulsar esta opción se carga un mapa (GoogleMaps), junto a un conjunto de íconos que representan cada uno de los sitios utpl registrados en la Base de Datos. Las funciones de este servicio son: visualizar información referente a un sitio seleccionado: nombre, descripción e imagen, así como la presentar la ubicación actual del dispositivo. El mapa permite alejar, acercar, y desplazar. Dentro de la clase se implementó un Overlay para agregar los POIs dentro del Mapa y una clase para mostrar la información del POI pulsado mediante un panel instantáneo.

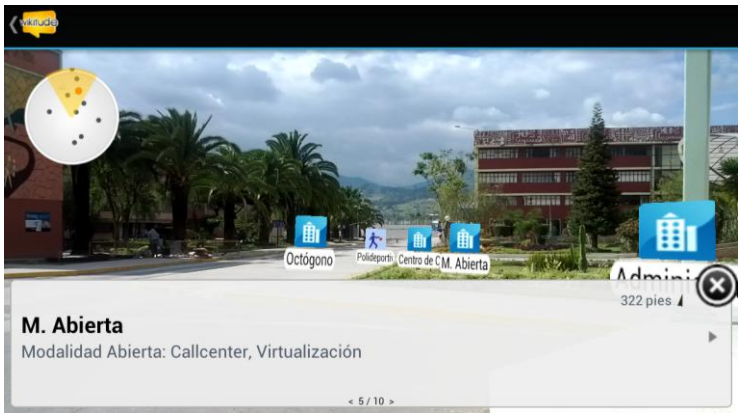

Fig. 2. Servicio: Mi vista AR-UTPLAR

a) Centros UTPL **b) Sitios Cercanos + DBpedia**

Fig. 3. Servicio Mapas- UTPLAR

Paradas de Bus UTPL: El servicio de Parada de Buses UTPL al igual que el servicio anterior carga un mapa digital para la visualización de las paradas cercanas según los metros de distancia seleccionados en la configuración.

En este servicio se muestran datos sobre: ubicación de parada, referencia y la imagen del lugar. Para este servicio se crean clases similares al servicio anterior, sin embargo se consume el API Google Directions para dibujar la mejor ruta de llegada.

Centros UTPL: Este servicio es similar a Sitios UTPL con la diferencia del área cubierta, ya que en esta se visualiza cada uno de los centros asociados y oficinas UTPL registrados en la Base de Datos actual, como se observa en la figura 3 (a).

Otros Sitios Cercanos: Este servicio implementa el servicio de Realidad Aumentada y Mapas, y permite visualizar cada uno de los POIs de Loja (ver figura 3b), además que se realizan consultas SPARQL para los POIs que tengan información en DBpedia.

Administración UTPLAR: Para la administración de datos del sistema UTPLAR se ha diseñado un Administrador Web acoplado a los ServiciosWeb creados; mismo que permitirá insertar, actualizar y eliminar datos.

4 Discusión y resultados

La aplicación fue desarrollada a partir del API Google 2.2, ya que es uno de los targets con mayor demanda en Google Play. Sin embargo fue necesario realizar pruebas en diferentes dispositivos y versiones a fin de evaluar su usabilidad (ver Tabla 1).

Tabla 1. Dispositivos donde se ha probado la aplicación UTPLAR

DISPOSITIVOS	Versión Android			
	2.2	2.3.4	2.3.6	4.0.2
Galaxy ACE		x	x	
Galaxy SII			x	
HTC EVO 3D		x		
Galaxy Nexus				x
Motorola DEFY mb526			x	
Sony Ericsson x10 mini	x			

Luego de realizar los experimentos en múltiples teléfonos y versiones Android. Se ha obtenido un alto grado de funcionamiento; es por tal motivo que se detallan a continuación cada uno de los resultados obtenidos:

La funcionalidad Vista AR tiene una exactitud promedio de 90%, esto depende exclusivamente de la señal de Red disponible, así como la exactitud del GPS, cabe

recordar que existe un margen de error conocido para estos dispositivos de 10 y 20 metros aproximadamente. Las funcionalidad de mapas de Sitios y Centros manejan una exactitud del 100% y su devolución de datos depende en sí de la conexión de datos del dispositivo. En relación al servicio de Paradas de Bus, existe una exactitud del 100% en cuanto a la colocación de cada POI y un 90% de precisión en cuanto a graficar la ruta. El servicio de Paradas de Bus funciona mejor desde un lugar al aire libre, puesto que se necesita señal GPS en tiempo real para mejores resultados.

Si bien es cierto que la velocidad de carga de los servicios disponible en la aplicación depende gran parte de la velocidad de descarga de la conexión de datos, también se puede calificar otro parámetro como la tecnología del dispositivo, puesto que con mejores capacidades se obtienen menos tiempos de carga de resultados.

Luego de finalizar la fase de pruebas se concluye que a nivel general la aplicación maneja un 97.5% de exactitud de datos.

5 Conclusiones

La Universidad Técnica Particular de Loja al contar con la aplicación UTPLAR permitirá no solo guiar a una persona visitante dentro del campus, sino que será un medio que brinde información sobre los centros universitarios, sitios importantes del campus, y la ciudad de Loja, así como información del Servicio de Transporte UTPL, mismo que permitirá reducir tiempos de búsqueda de información por parte del interesado.

6 Agradecimientos

Queremos agradecer a la Universidad Técnica Particular de Loja por el apoyo en la ejecución del programa de investigación "Tecnologías Avanzadas de la Web", que incluye al proyecto " Determinar el impacto del uso de principios de Realidad Aumentada y geolocalización en el acceso a contenidos educativos abiertos a través de dispositivos móviles", el cual ha posibilitado que este trabajo haya podido ser realizado.

Referencias

1. Azuma, R.: A survey of augmented reality. (1995)
2. Bover, A.: Aplicación de Gestión de Información Geolocalizada en Android. (2010). Disponible en: http://upcommons.upc.edu/pfc/bitstream/2099.1/11482/1/69369.pdf
3. Aulet, J. P.: "Geolocalización". (23/09/2011). Disponible en: http://www.sindikos.com/category/internet/geolocalizacion/
4. Valverde Rebaza, J., & Shiguihara Juárez, P.: Comunicaciones por Telefonía Móvil. Disponible en: http://es.scribd.com/doc/19063724/Comunicaciones-por-Telefonia-Movil
5. Peñalver Alonso, P.: Google Maps. (30 de Octubre de 2011). Disponible en: http://proyectoempresarial.wordpress.com/tag/web-2-0/
6. Burnette Ed.: Hello Android, Introducing Google's Mobile Development Platform, Second Edition. (30/12/2008). ISBN: 978-1-93435-617-3
7. Android Developers: Android Developers Guide. (2012). Disponible en: http://developer.android.com/index.html
8. Izquierdo, C. A.: Desarrollo de un sistema de Realidad Aumentada en Dispositivos Móviles. (2010). Disponible en: http://riunet.upv.es/bitstream/handle/10251/8597/PFC%20-%20Desarrollo%20de%20un%20sistema%20de%20Realidad%20Aumentada%20en%20dispositivos%20m%C3%B3viles.pdf
9. Europa Press: La Universidad de Oviedo incorpora la realidad aumentada a su Edificio Histórico. (2010). Disponible en: http://www.europapress.es/asturias/noticia-universidad-oviedo-incorpora-realidad-aumentada-edificio-historico-20120630130605.html
10. Colman, M. R.; Negri, G.: Una aplicación móvil de realidad aumentada para el ámbito universitario. (2010)
11. Piedra, N, Chicaiza, J. Lopez, J.: Combining Linked Data and Mobiles Devices to improve access to OCW. (Abril 2012). ISSN : 2165-9559.
12. Wikitude: Augmented Reality for Developers. (2012). Disponible en: http://www.wikitude.com/developer/wikitude-augmented-reality-for-developers