



Sistemas informáticos

Curso 2010-2011



Facultad de Informática
Universidad Complutense de Madrid

Proyecto FRAGUEL

Framework de realidad aumentada para guías en entornos locales

Alberto Guillén Álvarez
Gabriel Peñas Rodríguez
Bernardo Pericacho Sánchez



SSII - Proyecto FRAGUEL



FRAGUEL

Proyecto de Sistemas Informáticos
Facultad de Informática

Universidad Complutense de Madrid

Autores:

Alberto Guillén Álvarez
Gabriel Peñas Rodríguez
Bernardo Pericacho Sánchez

Profesores directores:

Dr. Pablo Gervás Gómez Navarro
Dr. Gonzalo Méndez Pozo

Curso 2010 / 2011



SSII - Proyecto FRAGUEL



Autorización:

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



SSII - Proyecto FRAGUEL



Palabras clave para búsquedas bibliográficas

- Realidad Aumentada
- Geolocalización
- Android

Keywords

- Augmented Reality
- Geolocation
- Android



SSII - Proyecto FRAGUEL



Queremos expresar nuestro agradecimiento:

- A nuestros familiares que nos infundieron la ética y el rigor que guían nuestro tránsito por la vida.
- A nuestros profesores directores del proyecto por su generosidad al brindarnos la oportunidad de recurrir a su capacidad y experiencia científica en un marco de confianza, afecto y amistad, fundamentales para la concreción de este trabajo.
- A nuestras parejas por concedernos el tiempo y la paciencia necesaria para la consecución del proyecto con éxito.



1. Sobre FRAGUEL

El nombre de FRAGUEL se deduce de FRAmework para GUías en Entornos Locales. Como su nombre indica, el objetivo de esta aplicación es facilitar la creación de puntos de interés y rutas entre dichos puntos, proporcionando un entorno sencillo y dinámico para acceder a los contenidos multimedia de cada uno. Existe una gran cantidad de aplicaciones de información basada en la localización del usuario, pero ninguna de ellas se basa en proporcionar rutas de diverso ámbito (cultural, ocio, etc.) y mucho menos en la creación de las mismas. De hecho, muy pocas permiten que un usuario añada contenido propio en el sistema.

Pudiéndose realizar sobre cualquier zona con cobertura 3G o Wifi (conexión a Internet a través de una red sin cables) y GPS, FRAGUEL, aporta un mecanismo de creación y compartición de rutas y puntos de interés. Todo esto es posible gracias a las características de los últimos terminales móviles basados en el sistema operativo Android. Nuestra plataforma aporta un modo de añadir textos, imágenes, vídeos y modelos 3D para realidad aumentada. Con todas estas herramientas a disposición del usuario se pueden crear desde guías turísticas, creadas por amigos y familiares, hasta guías didácticas de historia y cultura, desarrolladas por alguna institución.



2. About FRAGUEL

The name FRAGUEL stands for Local Environment Guides Framework (in spanish FRAmework para GUías en Entornos Locales). As the title says, the goal of this application is to make the creation of interest points and routes between those points easy, providing a friendly and dynamic environment to access to the multimedia associated to them. There are lots of applications of user location based information, but none of them can provide routes of so different kinds (cultural, spare time, etc.) and neither of them provides the creation of that information. In fact, few of them let the user add their own content into the system.

FRAGUEL application can be performed in any area with 3G or WiFi (wireless Internet through a wireless network) and GPS, and provides a mechanism for creating and sharing routes and points of interest. This is possible due to the features of the latest mobile handsets based on Android operating system. Our platform provides a way to add text, images, videos and 3D models for augmented reality. With all these tools available to users, routes can be created for tourist guides, created by friends and family about spare time, for tutorials on history and culture developed by an institution.



3. Sobre este documento

Este documento está estructurado en cuatro grandes secciones que dan nombre a las diferentes partes de la memoria.

La primera sección es la “introducción”, que es donde se describe y resume la idea básica del proyecto y todas las implicaciones que conlleva, de la propia aplicación del estado del arte en los diferentes ámbitos que atañen a FRAGUEL(Framework de Realidad Aumentada y Guía en Entornos Locales).

La segunda sección, “el producto”, habla del proyecto FRAGUEL como aplicación. Se detalla la arquitectura y diseño funcional bajando hasta el detalle de sus componentes.

En la tercera sección, “el proceso”, se trata la planificación que se ha llevado a cabo para desarrollar el producto formalizado según las propuestas de Pressman para metodología de desarrollo de software.

La cuarta sección, “Conclusiones”, resume los resultados finales obtenidos tras la finalización del proyecto, muestra las conclusiones y posibles líneas a seguir tras lo aprendido, los apéndices de la aplicación y la bibliografía empleada desde el inicio.

El documento en su práctica totalidad (salvo las secciones que la normativa establece) ha sido escrito en castellano por motivos prácticos.

El contenido del documento abarca todas las fases en el desarrollo de FRAGUEL. Desde el análisis y conceptualización previa, el diseño tanto de la arquitectura de la aplicación como el funcional y la finalización del producto.



4. About this document

This document is divided into four main sections that name the different parts of the report.

The first section is the "introduction" which describes and summarizes the basic idea of the project and all the implications of the application itself, the state of the art in various areas pertaining to FRAGUEL (Framework Augmented Reality and Local Environment Guide).

The second section, "the product", is about FRAGUEL project as an application. It details the architecture and its functional design detail down to its own components.

In the third section, "the process", is the planning that has been done to develop the product formalized as Pressman's proposals for software development methodology.

The fourth section, "conclusions," summarizes the final results obtained after completion of the project, shows the conclusions and possible lines to follow in a near future, of the application appendices and the bibliography used since the start.

Practically the whole document, except the sections established in the regulations, has been written in Spanish for practical reasons. The contents of the document cover all the stages in the development of FRAGUEL, from the previous analysis and conceptualization, to the architecture and functional design of the application until the end product.

**-INDICE-****INTRODUCCION**

I. Descripción y objetivos del proyecto.....	1
II. Estado de la cuestión.....	5
1. Smartphones.....	6
2. Realidad aumentada.....	7
2.1. Representación de los objetos.....	8
2.2. Representación del entorno	9
2.3. Posición de objetos y cámara.....	11
3. Posicionamiento y localización.....	13
3.1. GPS.....	13
3.2. Triangulación de antenas de telefonía.....	14
4. Sistemas operativos para dispositivos móviles.....	15
5. Productos similares.....	25

EL PRODUCTO

III. Tecnologías utilizadas.....	27
1. Definición de las tecnologías a emplear.....	28
2. Android OS.....	30
2.1. Desarrollando una aplicación Android.....	30
2.1.1. Ciclo de vida de una aplicación	31
2.1.2. El archivo Manifest.....	34
2.1.3. Interfaz de usuario.....	37
2.1.4. Recursos de una aplicación	45
2.2. Infografía de Android OS	46
3. OpenGL	47
IV. Características de FRAGUEL	53
1. Introducción	54
2. Arquitectura.....	55
2.1. Servidor	55
2.2. Cliente.....	55
3. Mapas y posicionamiento.....	55
3.1. Mapas.....	56
3.2. Puntos de interés.....	57
3.3. Rutas	61
3.4. Gestión de rutas	64
3.5. Localización y geotagging	67
4. Multimedia.....	67
4.1. Imágenes.....	68
4.2. Vídeos.....	69
4.3. Texto a voz.....	71
4.4. Realidad aumentada	72
4.5. Modelado del espacio.....	74

**EL PROCESO**

V. Gestión del proyecto.....	79
1. Planificación del proyecto.....	80
2. Gestión de riesgos.....	83
2.1. Identificación.....	83
2.2. Análisis y planificación.....	84
3. Casos de uso.....	89
4. Análisis de requisitos.....	93
5. Descripción de componentes.....	96
VI. Desarrollo del proyecto.....	103
1. Etapas de desarrollo.....	104
1.1. Primer hito.....	104
1.2. Segundo hito.....	105
1.3. Tercer hito	105
1.4. Cuarto hito	106
2. Proceso de trabajo.....	107
2.1. Creación de tareas.....	107
2.2. Desarrollo de las tareas.....	107
2.3. Revisión de fallos.....	108
CONCLUSIONES	
VII. Resultados.....	109
1. Discusión de los resultados obtenidos.....	110
1.1 Revisión de los objetivos iniciales.....	110
1.2. Resultados positivos.....	112
1.3. Resultados negativos.....	112
2. Descripción de las alternativas a las empleadas.....	113
3. Análisis de la complejidad final.....	113
4. Trabajo futuro.....	114
VIII. Apéndices.....	115
A. Manual de usuario.....	
IX. Referencias y bibliografía.....	146



SSII - Proyecto FRAGUEL



I

Descripción y objetivos del proyecto



El proyecto FRAGUEL nació con el objetivo de crear una aplicación que combine y haga fácilmente accesible al usuario todas las posibilidades multimedia actuales en los dispositivos móviles, las tecnologías de geolocalización y otra en pleno apogeo como lo es la Realidad Aumentada. Con el uso de todas estas tecnologías intentamos ofrecer al usuario una forma de obtener, consumir y compartir información estructurada y localizada mediante rutas situadas en un mapa en el mundo real.

El ámbito de la aplicación no pretende ajustarse al cultural, si no que se pretende que las limitaciones las establezcan los propios usuarios. De esta forma a través del framework podrán crearse desde rutas turísticas hasta rutas que muestren experiencias vividas o información comercial o publicitaria.

En la gran variedad de aplicaciones móviles no existe ninguna cuyo objetivo sea que el usuario aprendiera a través del terminal con información multimedia geolocalizada empleando, además, realidad aumentada. La mayoría del software disponible son juegos y aplicaciones que hacen uso de las características del terminal para mostrar información del entorno, pero dicha información suele ser principalmente comercial. Nuestra idea era combinar de alguna forma lo ya existente y a la vez que el usuario visualizara la información y los contenidos multimedia de una manera diferente empleando las últimas tendencias como lo es la realidad aumentada.

La no disposición de un software que situara dinámicamente al usuario en el mundo virtual de la realidad aumentada aumentó la necesidad de definir e implementar FRAGUEL.

La necesidad de desarrollarlo en un dispositivo móvil surge a la hora de seguir una ruta de forma guiada, ya que será imprescindible para el usuario al tener que desplazarse en espacios abiertos por los diferentes puntos que la formen. La portabilidad de los sistemas empotrados nos permite el movimiento y la interacción con la aplicación sin la necesidad de tener con estar conectados permanentemente a un ordenador de escritorio.



La utilización de las tecnologías de geolocalización nos proporciona la información necesaria para situar todos los elementos de la aplicación sobre el mundo en 2D. La realidad aumentada se beneficia de esta tecnología yendo un paso más allá, dándonos la posibilidad de situar objetos tridimensionales en un plano del mundo en 3D. Así podremos añadir por ejemplo monumentos enteros y emplazarlos en el punto que se quiera de una ruta cualquiera. Además, estos objetos 3D se pintarán en el terminal haciendo uso de OpenGL.

Por tanto, nuestros objetivos a resolver con este proyecto son los siguientes:

- **Implementación del sistema en dispositivos móviles** para que el usuario pueda moverse libremente sin necesidad de estar conectado permanentemente a una conexión eléctrica.
- **Posicionamiento en exteriores** mediante los sistemas de posicionamiento disponibles en los terminales (3G,Wifi y GPS) tratando de minimizar error en la medida de lo posible debido a la necesidad de posicionar al usuario en el mundo virtual que ofrece la realidad aumentada.
- **Uso de la realidad aumentada** (además de la geolocalización) como forma de innovación de las típicas aplicaciones de rutas guiadas. Además se deberá poder emplear **modelos en tres dimensiones** que mejoran el aspecto de esta tecnología.
- **Participación activa del usuario**, haciéndole consciente de que sus acciones son las que desatan los eventos en la aplicación, ya sea por las zonas donde se mueve o por las acciones que lleva a cabo durante la ejecución. Para aumentar más la interacción del usuario, se le brindará la oportunidad de **crear su propio contenido y compartirlo** con los demás de una manera sencilla que no implique tener un avanzado conocimiento en el campo de la informática.
- Emplear **alto contenido multimedia** para que sea más ameno y sencillo la navegación por el contenido informativo.
- Desarrollar el **contenido** suficiente para demostrar que nuestro proyecto es viable.



El enfoque inicial fue realizar un sistema educativo que cumpliera las características anteriores, guiando al usuario por la historia de manera diferente e interactiva.

Por mayor accesibilidad decidimos hacer uso de aquello que hubiera sucedido en el entorno de la Facultad de Informática de la Universidad Complutense de Madrid. Esto nos llevó a los acontecimientos de la Batalla de Madrid, dentro del contexto de la Guerra Civil Española (1936-1939).

Con esta ambientación como objetivo deberíamos buscar toda la información disponible sobre dicha batalla en el entorno de la Ciudad Universitaria. Esto incluye fotos, vídeos y mapas que podríamos mostrar al usuario.

Durante el primer mes de trabajo nos percatamos que debíamos enfocar el proyecto de otra manera, no ciñéndonos simplemente al ámbito educativo y cultural, si no proporcionar un marco en que se pudiesen implementar rutas y contenido de toda índole.



II

Estado de la cuestión



1. Smartphones

El **teléfono inteligente** (smartphone en inglés) es un término comercial para denominar a un teléfono móvil que ofrece más funciones que un teléfono celular común. Entre las funcionalidades extra cabe destacar que la práctica totalidad de los terminales disponen de: localización GPS, conexiones WIFI y 3G, brújula digital, acelerómetros, giróscopos y cámara digital, así como la capacidad de instalar programas para incrementar el procesamiento de datos y la conectividad. Como dato destacar que la totalidad de los terminales desde 2007 hasta la actualidad soportan completamente un cliente de correo electrónico con la funcionalidad completa de un organizador personal.

Este tipo de teléfonos abren una nueva puerta a los desarrolladores de software dado el sorprendente aumento en la potencia de procesamiento y de gráficos integrada. Los nuevos terminales incorporan la multitarea proporcionando aún más rendimiento. Por último también cabe destacar otras capacidades y nuevas características de estos dispositivos tales como navegador, cámara digital integrada, lectura y edición de documentos de negocios en variedad de formatos así como la más importante de ellas ya citada, instalación de programas desarrollados por terceros, el fabricante del dispositivo o por el operador.

Las principales empresas del sector se han apresurado a lanzar distintos sistemas operativos con mayor o menor éxito. De todos ellos, los que tienen actualmente más renombre son Android OS de Google y iOS de Apple, aunque Symbian (Nokia) también tiene una gran cuota de mercado. Éste último, ha tenido la mayor cuota de mercado durante los últimos años debido a la gran aceptación de los teléfonos Nokia, pero se espera que a finales de 2011 Android OS aumente considerablemente su diferencia con su principal competidor.

A continuación se detallan los sistemas operativos más extendidos en este momento:

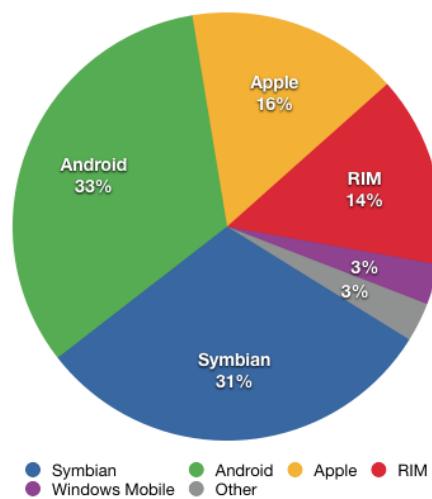


Figura 1: SO en terminales



Sistema operativo	Cuota en 2010 ¹	Cuota en 2009
Android	32,9%	3,5%
Symbian OS	30,6%	44,6%
iOS	16,7%	17,1%
BlackBerry OS	14,6%	20,7%
Windows Phone	2,9%	7,9%
Otros	2,7%	6,5%

Figura 2: Tabla de situación del mercado

2. Realidad aumentada

La realidad aumentada (RA) es el término que se usa para definir una visión directa o indirecta de un entorno físico del mundo real, cuyos elementos se combinan con elementos virtuales para la creación de una realidad mixta a tiempo real. Consiste en un conjunto de dispositivos que añaden información virtual a la información física ya existente, es decir, añadir una parte sintética virtual a lo real. Esta es la principal diferencia con la realidad virtual, puesto que no sustituye la realidad física, sino que sobreimprime los datos informáticos al mundo real.



Figura 3: Realidad aumentada

Esta técnica está en auge debido a la proliferación de los **smartphones** y el considerable aumento tanto en potencia de procesamiento como de sensores disponibles. Con ello se abre un gran campo que supone interesantes retos:

- Usabilidad: la posibilidad de crear aplicaciones que supongan un beneficio para el usuario, ya sea ahorrándole tiempo, educándole...
- Rendimiento: pese a disponer de bastante mejor hardware que con los anteriores teléfonos móviles, sigue siendo imprescindible realizar buenas prácticas de programación para que la aplicación se ejecute de forma fluida y sin errores.



2.1 Representación de los objetos

Actualmente hay dos tendencias referentes a los objetos que se representan en el entorno:

- Objetos 2D: Por simplicidad y menor coste, la mayoría de las aplicaciones existentes optan por esta alternativa. Esto se debe a que dicho software no requiere de representaciones gráficas complejas, basta con alguna imagen representativa en dos dimensiones y realizar escalados para que simule un efecto tridimensional. En este grupo destacan principalmente las aplicaciones de información espacial que proporcionan, por ejemplo, información de tiendas, lugares de ocio, parking... Un buen ejemplo de esto es la aplicación comercial “Layar”.



Figura 4: Realidad aumentada y objetos 2D

- Objetos 3D: Esta rama es más costosa, tanto en el diseño como en la implementación. Sin embargo cada vez tiene más peso debido al aumento de las características de los terminales así como al incremento en el desarrollo de videojuegos para los mismos. En este caso el objetivo es representar mallas tridimensionales haciendo uso de los sistemas proporcionados, como OpenGL. Pocas aplicaciones hacen uso de esta mecánica ya que incrementa considerablemente los costes asociados a todos los ámbitos de desarrollo, pero en el ámbito de los videojuegos es dónde más provecho se le saca.



Figura 5: Videojuegos y realidad aumentada

2.2 Representación del entorno

También cabe destacar las distintas maneras de representar el entorno sobre el que se añaden los objetos. Y, al igual que estos, existen tendencias en dos o tres dimensiones. Este aspecto, sin embargo, tiene mayor repercusión sobre el nivel de atracción del usuario.

- Entorno 2D: Para este método basta con una representación sencilla de la zona, como puede ser un mapa. Es la de uso más extendido dado que su uso es más cómodo en la mayoría de las situaciones de uso. Al igual que sucedía con los objetos 2D simplifica mucho todo el proceso de desarrollo, de hecho la combinación más utilizada es entornos y objetos 2D.

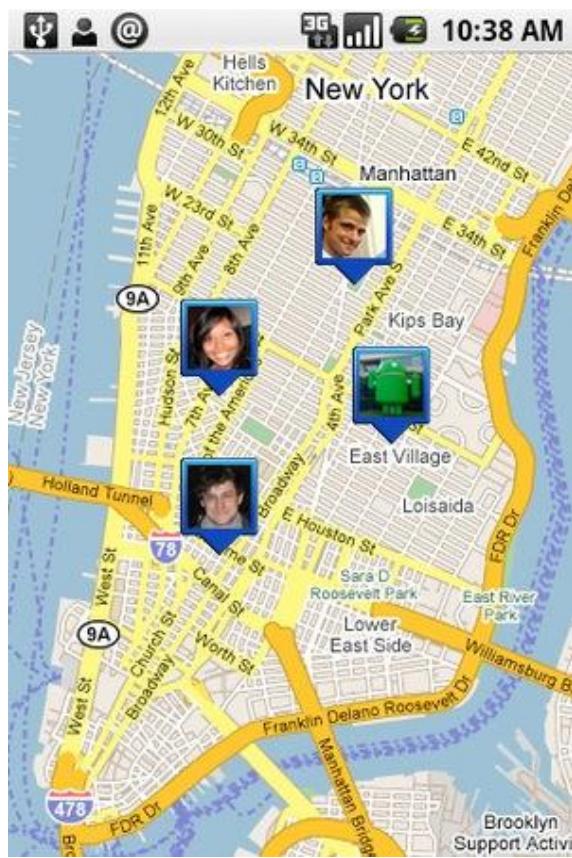


Figura 6: Google Latitude muestra la posición de tus conocidos

- Entorno 3D: Supone la aproximación más interesante de las dos, ya sea combinada con objetos 2D o 3D. Hace uso de la cámara y sensores de rotación y orientación integrados en el terminal para mostrar los objetos en la imagen capturada en tiempo real. También es la más problemática debido a que por ahora los sistemas presentan ciertas incompatibilidades e inestabilidad en algunos casos. Ahora mismo hay un gran filón debido al uso de guías GPS para vehículos que sobreimprimen la ruta a tomar.



Figura 7: Wikitude indica el camino de la forma más intuitiva



2.3 Posición de objetos y cámara

También existen divergencias en el modo de indicar la localización de los distintos elementos que intervienen en el proceso de realidad aumentada. Si bien alguna puede parecer más interesante que otra, todas tienen sus ventajas y desventajas, sin embargo hay que escoger la que mejor se adapte al uso que se requiere. A continuación se explican los dos métodos más utilizados.

- Marcadores: consiste en aplicar técnicas de reconocimiento de imagen para encontrar un determinado patrón. Una vez encontrado se analiza para obtener la orientación, posición e inclinación del observador mediante operaciones matriciales.

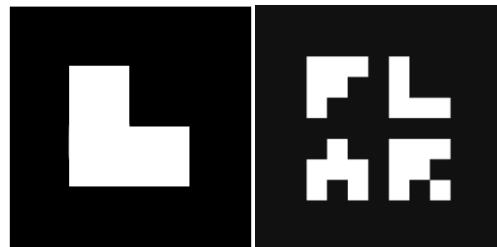


Figura 8: Marcadores sencillos frente a elaborados

El uso de marcadores tiene varias ventajas. Al tener un objetivo en el mundo real hace que tengan mucha precisión y los cálculos necesarios no son extremadamente costos. Otra ventaja es que al cambiar de marcador se puede aplicar una información diferente.



Figura 9: Nintendo también hace uso de realidad aumentada

Por otro lado este método tiene un gran lastre. Al ser necesario un marcador se impide que se pueda mostrar algo en ausencia de este. Sin embargo hay empresas que se dedican a un reconocimiento de imágenes más complejo para que sea más agradable para el usuario.



Figura 10: AR junto a reconocimiento de imágenes

- Sensores: Este método se limita a hacer uso de la información proporcionada por los sensores del terminal. Es una aproximación mucho más atractiva ya que da libertad total al usuario para moverse por el entorno. Y, algo importante, requiere un cómputo claramente inferior a procesar una imagen; como mucho realizar alguna conversión con los valores obtenidos.

Con esta aproximación basta con posicionar los objetos en posiciones fijas y convertir los datos de los sensores en valores de nuestro entorno para localizar la cámara.



Figura 11: Hay mucho interés en torno a la realidad aumentada

Sin embargo, su mayor virtud es al mismo tiempo su mayor defecto. Los sensores de orientación tienen una precisión aceptable, pero los de posición (basados en GPS, triangulación de antenas, etc.) tienen demasiado error y perjudican seriamente al resultado final.



3. Posicionamiento y localización

Un pilar clave de la aplicación es la posibilidad de determinar la posición del terminal en exteriores. Para ello se combinan dos técnicas, que, complementadas, incrementan la precisión obtenida.

3.1 GPS

El GPS (siglas de “Sistema de Posicionamiento Global” en inglés) engloba un conjunto de satélites y tecnologías que permiten determinar la posición de un aparato receptor en cualquier lugar del planeta.

La red consta de un total de 24 satélites en órbita MEO, con trayectorias sincronizadas para optimizar el área de cobertura. Dichos aparatos emiten constantemente una señal que identifica al satélite y la hora de emisión. Con estos datos, y mediante triangulación, es posible determinar la posición del aparato. Por tanto es imprescindible disponer de un mínimo de tres satélites visibles para realizar los cálculos necesarios.

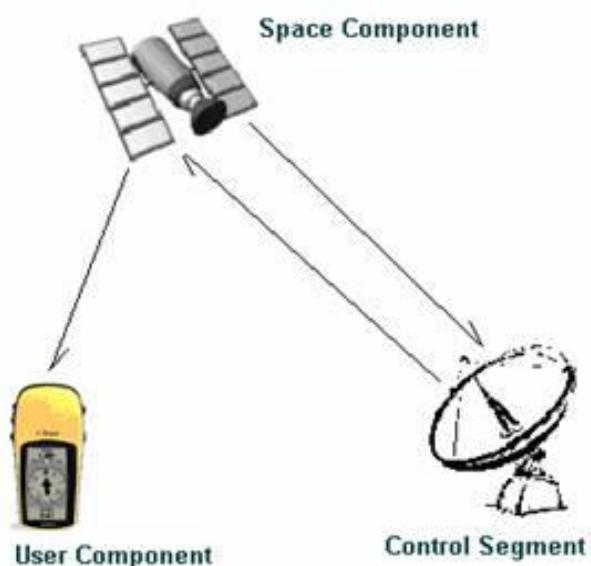


Figura 12: Elementos en el posicionamiento GPS

En la actualidad este sistema se encuentra muy extendido, debido al bajo coste de los aparatos receptores. Se pueden encontrar en todo tipo de entornos, desde vehículos hasta marcadores en animales salvajes.



Este sistema no está ausente de problemas:

- Al ser de uso primario militar puede que debido a conflictos quede restringido el uso de los satélites.
- La precisión es mayor en el rango de frecuencias de uso militar, quedando relegado el uso civil a unos 15 metros frente a menos de 2 metros en la frecuencia militar.
- Los edificios y demás obstáculos causan sombra a los satélites, por lo que puede que aumente el error al estar cerca de estos.

Hay otros sistemas en desarrollo similares al GPS:

- GLONASS: llevado a cabo por la extinta Unión Soviética y actualmente gestionado por la Federación Rusa. Comparte características con el GPS pero no ha llegado estar tan acabado.
- Galileo: desarrollado en la Unión Europea y de uso civil. Se espera que esté disponible a partir de 2015. Supondrá un considerable aumento de la precisión. Cabe destacar también que hay muchos otros países interesados entre los que se encuentran China, Brasil e Israel.

3.2 Triangulación de antenas de telefonía

Esta alternativa hace uso de cálculos básicos de geometría. Se apoya en el conocimiento de la posición y potencia de la señal de las antenas de telefonía que proporcionan servicio al terminal. Dicha información se adquiere mediante las antenas internas (potencia de la señal en decibelios) o mediante intercambio de información con cada antena dentro de rango (posición de emisión).

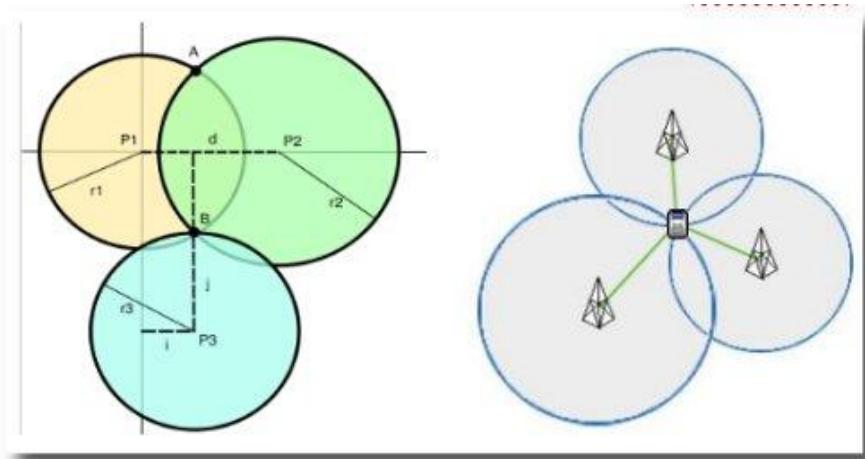


Figura 13: Esquema de los cálculos de triangulación



En la posición *B*, se pueden calcular las coordenadas usando los puntos conocidos *P1*, *P2* y *P3* en un plano horizontal. Medir la distancia *r₁* se pone en un círculo. Medir *r₂* se pone en dos puntos *A* o *B*. Medir la tercera distancia *r₃*, le da las coordenadas del punto *B*. Este cálculo se conoce como *resección* o *trilateración*.

4. Sistemas operativos para dispositivos móviles

A continuación se especifican brevemente los sistemas operativos para dispositivos móviles que tienen cuota de mercado apreciable. Algunos de estos sistemas operativos como Android o iOS funcionan tanto en smartphones como en los nuevos dispositivos móviles táctiles: las tabletas PC. Estos dispositivos se encuentran en medio de dos dispositivos: los ordenadores portátiles y los teléfonos móviles.

- **Android**



Android es un sistema operativo basado en Linux para dispositivos móviles como smartphones y tablets (computador portátil con la que se interactúa mediante una pantalla táctil o multitáctil). Inicialmente fue desarrollado por Android Inc, empresa que Google adquirió en 2005. Está compuesto por 12 millones de líneas de código, incluyendo 3 millones de líneas de XML, 2.8 millones de líneas de lenguaje C, 2.1 millones de líneas Java y 1.75 millones de líneas de C++.

Las características principales de este sistema son:

1. Framework de aplicaciones que permite reutilizar componentes.
2. Máquina virtual Dalvik optimizada para dispositivos móviles.
3. Navegador integrado basado en el motor de código abierto WebKit.
4. Gráficos optimizados impulsados por una librería gráfica 2D personalizada y la librería gráfica 3D basada en la especificación de OpenGL ES 1.0 (la aceleración hardware es opcional).
5. SQLite para el almacenamiento de datos estructurados.



6. Soporte multimedia para audio, vídeo e imágenes; todos ellos con varios formatos (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
7. Telefonía GSM (depende del hardware del dispositivo).
8. Bluetooth, EDGE, 3G and WiFi (depende del hardware del dispositivo).
9. Cámara, GPS, brújula y acelerómetro (depende del hardware del dispositivo).
10. Entorno de desarrollo que incluye un emulador de dispositivos, herramientas de depuración, memoria y perfiles de rendimiento y un plugin para Eclipse IDE.

Proporcionando una plataforma de desarrollo libre, Android ofrece a los desarrolladores la posibilidad de crear innovadoras aplicaciones. Los desarrolladores pueden utilizar las ventajas del hardware del dispositivo, acceder a la información de la posición, ejecutar servicios en ‘background’, establecer alarmas, añadir notificaciones a la barra de estado, etc.

Los desarrolladores tienen acceso íntegro a las mismas API's del framework que las aplicaciones del núcleo. La arquitectura de las aplicaciones está diseñada para simplificar la reutilización de los componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede hacer uso de dichas capacidades (sujeto a las restricciones de seguridad forzadas por el framework). Este mismo mecanismo permite que los componentes sean reemplazados por el usuario.

Principalmente, todas las aplicaciones son un conjunto de servicios y sistemas que incluyen:

- Un conjunto rico y extensible de “Views” que puede ser usado para crear una aplicación, incluyendo listas, cuadrículas, cajas de texto, botones e incluso un navegador web empotrado.
- “Content Providers” que permiten a las aplicaciones acceder a los datos de otras aplicaciones como los Contactos, o compartir sus propios datos.
- El gestor de recursos (“Resource Manager”), proporcionando acceso a los recursos estáticos como los Strings localizados, gráficos y archivos de diseño de interfaz.
- El gestor de notificaciones (“Notification Manager”) que permite a todas las aplicaciones mostrar sus propias alertas en la barra de estado.



- El gestor de actividades (“Activity Manager”) gestiona el ciclo de vida de las aplicaciones y proporciona un cambio entre estados común.

Android incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje Java. Cada aplicación Android ejecuta su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede correr múltiples máquinas virtuales de forma eficiente. Dalvik ejecuta archivos en el formato Dalvik Executable (.dex), el cual está optimizado para memoria mínima. La Máquina Virtual está basada en registros y corre clases compiladas por el compilador de Java que han sido transformadas al formato .dex por la herramienta incluida "dx".

Android depende de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila de software.



Figura 14: Arquitectura de Android OS



Este sistema operativo incluye un conjunto de librerías C/C++ que son usadas por varios componentes del sistema Android. Estas capacidades están disponibles para los desarrolladores a través de de Framework de Aplicaciones de Android. Algunas de las librerías del núcleo más importantes son las siguientes:

■ Librería C del sistema:

- [Symbian OS](#)
- [iOS](#)
- [RIM \(BlackBerry OS\)](#)
- [Windows Phone](#)
- [Otros](#)

Al comienzo de la realización de este proyecto estaba a la cabeza en número de terminales Android OS, que había multiplicado el número de dispositivos por diez, duplicando a iOS pese a la buena fama adquirida de este en sus anteriores terminales como iPhone3G e iPhone3GS.

• [iOS de Apple](#)



iOS (anteriormente denominado iPhone OS) es un sistema operativo móvil de Apple desarrollado originalmente para el iPhone, siendo después usado en el iPod Touch e iPad. Es un derivado de Mac OS X, que a su vez está basado en Darwin BSD. El iOS

tiene 4 capas de abstracción: la capa del núcleo del sistema operativo, la capa de "Servicios Principales", la capa de "Medios de comunicación" y la capa de "Cocoa Touch". Todo el sistema se encuentra en la partición "/root" del dispositivo, ocupa poco menos de 500 Megabytes. También llamado D-OS por sus fans.

Este sistema operativo es propio de un ordenador de sobremesa pero reinventado para dispositivos móviles. Como se basa en el sistema operativo para ordenadores más avanzado del mundo —Mac OS X—, iOS 4 es veloz y estable. También gestiona el consumo energético y ofrece la mayor autonomía posible. Además, su capacidad de multitarea permite usar varias aplicaciones al mismo tiempo sin renunciar al rendimiento de la aplicación en activo ni desperdiciar batería.



Dado que Apple fabrica tanto los componentes del iPhone como el sistema operativo, todo funciona en conjunto mejorando el rendimiento. Esta integración sin fisuras permite a las aplicaciones aprovechar al máximo las prestaciones del dispositivo, como la pantalla retina, la interfaz Multi-Touch, el acelerómetro, el giroscopio de tres ejes, el GPS y los gráficos. FaceTime es el ejemplo perfecto: usa las dos cámaras del iPhone 4, la pantalla, uno de los dos micrófonos y la conexión Wi-Fi.

Las características más importantes que ofrece este sistema operativo son:

FaceTime + iOS 4: FaceTime es otra prueba de que iOS 4 integra el hardware y el software de una manera revolucionaria, para que todas las prestaciones del iPhone sean muy fáciles de usar. Se puede ver a la persona con la que estás teniendo una conversación mientras hablas con ellos de iPhone 4 a iPhone 4 o iPod touch vía Wi-Fi de manera gratuita, lo que es toda una revolución en el mercado.

Game Center: iOS 4 presenta la aplicación Game Center, una red social para los juegos incluida de serie en el iPhone 4 e iPod touch. Se puede consultar en qué puesto has quedado, comparar tus logros con los de tus amigos, escoger la selección automática o desafiar a un grupo de amigos e incluso medirte con otros miembros de la red en un juego multijugador.

Multitarea: iOS 4 introduce un nuevo concepto de multitarea . Existe la posibilidad de ejecutar las aplicaciones de terceros y pasar de una a otra sin interrupciones , sin afectar a la velocidad de la aplicación en activo y sin consumir más batería de la necesaria.

Por otro lado, Antes de iOS 4, la multitarea estaba reservada para aplicaciones por defecto del sistema. A Apple le preocupaban los problemas de batería y rendimiento si se permitiese correr varias aplicaciones de terceros al mismo tiempo. A partir de iOS 4, dispositivos de tercera generación y posteriores soportan el uso de 7 API's para multitarea, específicamente:

- Audio en segundo plano
- Voz IP
- Localización en segundo plano
- Notificaciones push
- Notificaciones locales
- Completado de tareas
- Cambio rápido de aplicaciones



Sin embargo, no consiste en una verdadera multitarea, pues las aplicaciones ajenas al SO, quedan congeladas en segundo plano no recibiendo un solo ciclo de reloj del procesador.

- **Rim – BlackBerry OS**



El BlackBerry OS es un sistema operativo móvil desarrollado por Research In Motion para sus dispositivos BlackBerry. El sistema permite multitarea y tiene soporte para diferentes métodos de entrada adoptados por RIM para su uso en computadoras de mano, particularmente la *trackwheel*, *trackball*, touchpad y pantallas táctiles.

Su desarrollo se remonta la aparición de los primeros 'hanheld' en 1999. Estos dispositivos permiten el acceso al correo electrónico , navegación web y sincronización con programas como Microsoft Exchange o LotusNotes aparte de poder hacer las funciones usuales de un teléfono móvil convencional.

RIM estuvo en disputa con NTP Inc. la cual le acusaba de violar cinco patentes que pudo haber dejado sin servicio a sus usuarios en Estados Unidos (sobre tres millones). Las compañías llegaron a un acuerdo extrajudicial que soluciono la disputa en marzo de 2006 previo pago de 612 millones de dólares por parte de RIM.

Características principales de BlackBerry OS:

El SO BlackBerry esta claramente orientado a su uso profesional como gestor de correo electrónico y agenda. Desde la versión actual, la cuarta, se puede sincronizar el dispositivo con el correo electrónico, el calendario, tareas, notas y contactos de Microsoft Exchange Server además es compatible también con Lotus Notes y Novell GroupWise.

BlackBerry Enterprise Server (BES) proporciona el acceso y organización del email a grandes compañías identificando a cada usuario con un único BlackBerry PIN. Los usuarios más pequeños cuentan con el software BlackBerry Internet Service, programa más sencillo que proporciona acceso a Internet y a correo POP3 / IMAP / Outlook Web Access sin tener que usar BES.

Al igual que en el SO Symbian desarrolladores independientes también pueden crear programas para BlackBerry pero en el caso de querer tener acceso a ciertas funcionalidades restringidas necesitan ser firmados digitalmente para poder ser asociados a una cuenta de desarrollador de RIM.



Sistema operativo BlackBerry OS 6:

Blackberry 6 es un sistema desarrollado por Research In Motion la cual fue presentado en el WES 2010 junto con un video promocional donde se muestra algunas novedades. RIM apuesta que su BlackBerry 6 estará enfocado en el mercado corporativo y no-corporativo. La mejor experiencia de este sistema se encontrara en los equipos touchscreen (Pantalla Táctil), aunque RIM aseguro que en los equipos que cuenten con un TouchPad o TrackPad podrán ejecutarlo ya que ejerce casi la misma función. Así mismo todavía RIM no ha aclarado cuales son los equipos que se podrán actualizar a esta versión aunque hay muchos rumores al respecto.

RIM en el desarrollo de este OS se enfocó en la parte multimedia hacia el usuario, sin dejar a un lado la parte profesional, también se muestra la integración de las redes sociales y la mensajería instantánea en este. Sin duda RIM quiere dar al usuario una nueva experiencia en su equipo BlackBerry que nadie conocía.

Novedades de BlackBerry 6

- Un renovado diseño
- Nuevo Navegador con tecnología WebKit
- Nueva experiencia con las redes sociales (Facebook, Twitter, MySpace) y mensajería instantánea (BlackBerry Messenger, Windows Live Messenger)
- Posibilidad de ejecutar juegos 3D
- Item de lista numerada
- Menú de contexto gráfico en listado de tablas y pestañas.
- Múltiple lista de contactos.
- Mejoras en la aplicación de mensajes (Soporte push para aplicaciones de terceros incluido)
- Soporte para Wi-Fi LBS
- Reverse Geo-Coding
- Servicio de tiempo de viaje (Solo USA y Canada)
- Soporte para nuevas funciones con el TrackPad
- Soporte para escaneo de cogido de barras en 1D/2D
- Mejoras en el Auto-Focus (los lentes de la cámara y el Auto-Focus funcionan por separado)
- Reconocimiento de rostro en la cámara.



- **Windows Phone**



Windows Phone, anteriormente llamado Windows Mobile es un sistema operativo móvil compacto desarrollado por Microsoft, y diseñado para su uso en teléfonos inteligentes (*Smartphones*) y otros dispositivos móviles . Windows Phone hace parte de los sistemas operativos con interfaz natural de usuario.

Se basa en el núcleo del sistema operativo Windows CE y cuenta con un conjunto de aplicaciones básicas utilizando las API de Microsoft Windows. Está diseñado para ser similar a las versiones de escritorio de Windows estéticamente. Además, existe una gran oferta de software de terceros disponible para Windows Mobile, la cual se puede adquirir a través de Windows MarketPlace for Mobile.

Originalmente apareció bajo el nombre de *Pocket PC*, como una ramificación de desarrollo de Windows CE para equipos móviles con capacidades limitadas. En la actualidad, la mayoría de los teléfonos con Windows Mobile vienen con un estilete digital , que se utiliza para introducir comandos pulsando en la pantalla. Windows Mobile ha evolucionado y cambiado de nombre varias veces durante su desarrollo, siendo la última versión la llamada Windows Phone 7, anunciada el 15 de febrero del 2010 y sujetada a disponibilidad a finales de 2010.

Características principales de Windows Phone:

Tanto Windows Mobile para Pocket PC , como Windows Mobile para Smartphones, poseen bastantes aspectos parecidos.

- En la pantalla "Hoy" nos mostrará la fecha actual, la información del dueño, las citas próximas, los mensajes Email, y las tareas. En la parte inferior aparecerá, generalmente, una barra con dos botones. También incluye una barra que incluye iconos para notificar el estado del Bluetooth, batería, cobertura, etc. Este tema predeterminado puede ser cambiado añadiendo o eliminando complementos, como por ejemplo, alarma, temperatura, estado de la batería.
- En la barra de tareas muestra: la hora actual, el volumen y el estado de la conectividad. Cuando un programa o un mensaje están abiertos el espacio en blanco, en el que estaba el reloj se convierte en una "ok" o un ícono de cerrar (x). La característica principal de la barra de tareas es el botón de *Inicio*, que está diseñado para que sea parecido al botón de Inicio de las versiones de escritorio de Windows. El menú de Inicio ofrece programas abiertos recientemente, nueve entradas del menú personalizadas, y accesos directos a programas, ajustes, búsquedas, y ayuda.



- Las versiones Pocket PC incluyen en Windows Mobile aplicaciones de Microsoft Office. Éstos incluyen Pocket Word y Pocket Excel. En Windows Mobile 5.0 se incluye Pocket PowerPoint. Estas versiones incluyen muchas de las características que se utilizan en versiones de escritorio, pero algunas otras características como la inserción de las tablas e imágenes no se han incluido versiones anteriores a Windows 5.0. ActiveSync tiene la capacidad de convertir archivos de versiones de escritorio a archivos compatibles con Pocket PC.
- Outlook Mobile es también un programa que viene con Windows Mobile. Esto incluye tareas, calendario, contactos, y la bandeja de entrada. Microsoft Outlook para las versiones de escritorio se incluye a veces en los CD-ROM's del fabricante del Pocket PC
- Windows Media Player for Windows Mobile se añade con el software. Actualmente, todas las Pocket PC incluyen la versión 9 del reproductor, pero la versión 10 se ha incluido con un hardware más nuevo y con las nuevas versiones de Windows Mobile. Para algunos dispositivos, la versión 10 está disponible para su descarga solo para determinados dispositivos - éstos incluyen los dispositivos de la gama de Dell Axim. Windows Media Player reproduce: WMA, WMV, MP3 y AVI. Los archivos MPEG actualmente no están soportados, y se debe descargar un programa de terceros para reproducirlos, y los archivos de WAV se reproducen en un reproductor por separado. Algunas versiones son también capaces de reproducir MP4.
- Cliente para RPV's PPTP.



- **Symbian OS**



Symbian es un sistema operativo que fue producto de la alianza de varias empresas de telefonía móvil, entre las que se encuentran Nokia, Sony Ericsson, Psion Samsung, Siemens, Arima, Benq, Fujitsu, Lenovo, LG, Motorola, Mitsubishi Electric, Panasonic, Sharp, etc. Sus orígenes provienen de su antepasado, EPOC32, utilizado en PDA's y Handhelds de PSION.

El objetivo de Symbian fue crear un sistema operativo para terminales móviles que pudiera competir con el de Palm o el Windows Mobile de Microsoft y ahora Android de Google Inc., iOS de Apple Inc. y Blackberry OS.

Actualmente se encuentra en declive debido a que la empresa que más apostó por este sistema operativo, recientemente ha firmado un contrato con Windows Phone.



5. Productos similares

- **Wikitude**



Es la primera aplicación de realidad aumentada que salió al mercado. Wikitude es una guía para viajes y turismo basada en sistemas de geoposicionamiento, brújula digital, sensores de orientación y acelerómetro, mapas, video y contenidos informativos de la Wikipedia, desarrollada para la plataforma Android. Se trata de una aplicación cuya propuesta de valor es ayudar a un

viajero a planificar su viaje o encontrar información relevante sobre un sitio concreto o zona geográfica. Actualmente, Wikitude presenta información de 350.000 POI de todo el mundo que pueden ser buscados mediante GPS o mediante su dirección. Estos puntos de interés geográficos son mostrados en el móvil con Android.

Además, el navegador Wikitude 3D permite una mejor interacción con las localizaciones en las que se encuentra el usuario en tiempo real.

Mientras que la realidad aumentada 2D sólo muestra objetos superpuestos, la 3D presenta lugares, paisajes y objetos en 3 dimensiones, haciendo que sea muy fácil diferenciarlos y apreciarlos.

Por otra parte, el navegador ofrece a los usuarios descubrir detalles sobre las cercanías y acceder a artículos relacionados de la Wikipedia o información relevante publicada en Twitter.



Figura 16: Wikitude



- **Layar**



Layar es la aplicación de realidad aumentada más utilizada actualmente en dispositivos móviles. Es un navegador de realidad aumentada para Android, su funcionamiento se basa en usar la información que proporciona el GPS y la brújula que posee el terminal, mientras la pantalla nos muestra lo que la cámara capta y sobre ella información relativa en tiempo real de lo que se encuentra delante del usuario. Muestra en una interfaz la distancia hasta los puntos en cuestión, y está integrada con Google Maps si la ruta hasta el elemento es complicada. Además se caracteriza por disponer de un sistema que permite utilizar capas con las que se podrá decidir qué tipo de información se quiere desplegar.



Figura 15: Layar Reality Browser



III

Tecnologías utilizadas



1. Definición de las tecnologías a emplear.

En esta sección se muestran las razones por las que, finalmente, decidimos emplear unas tecnologías frente a otras para el desarrollo del proyecto.

La decisión de qué plataforma de desarrollo utilizar como base para la creación de nuestra aplicación, ha sido sin duda la que más impacto ha tenido en el proyecto.

Como hemos visto en apartados anteriores, cuando hablamos de terminales smartphone, los dos principales sistemas operativos que encontramos son Android, por parte de Google, e iOS, por parte de Apple.



Figura 1: iOS vs Android OS

Uno de los puntos que se han usado para catalogar a iOS o Android como mejor sistema operativo ha sido la fragmentación. Este término se utiliza para definir las variaciones que sufre un sistema operativo, debido a la adopción y personalización de fabricantes y/o operadores.

En el caso de IOS, la única fragmentación que existe se da en los propios dispositivos de Apple. Es decir, iPod Touch, iPhone (con sus respectivas versiones) y iPad. En Android, la fragmentación es mayor, pues cada fabricante incorpora o limita las aplicaciones de serie, como también personaliza la interfaz propia del OS, lo que hace que cada modelo de Android sea distinto, creando rivalidad entre dispositivos con el mismo sistema operativo.



La fragmentación puede ser positiva o negativa, según se mire. En el mundo Android, vemos como cada fabricante lanza su propio teléfono, con hardware distinto y versiones de Android diferentes. Este punto es negativo porque cada usuario dispone de una versión que, dependiendo de su antigüedad, soportará ciertas aplicaciones u ofrecerá más o menos servicios. Apple, sin embargo, ofrece cada nueva versión a todos sus dispositivos, lo que supone tener el terminal siempre a la última. Algunos iPhone o iPod se han quedado sin poder recibir dichas actualizaciones, pero para ser sinceros, eso ha sido a causa de limitaciones técnicas, ya que los componentes se han quedado anticuados para ciertas funciones.

Esto lleva a que Android venda más teléfonos que IOS, al existir más modelos y más marcas, pero la calidad o prestaciones dependerán mucho del teléfono que posea el usuario.

Aunque no entra ni puede entrar dentro de los objetivos de este proyecto, para ser justos hay que tener en cuenta también el aspecto económico. Apple ha sido toda una visionaria en cuanto al modelo de negocio de los teléfonos inteligentes. Android se ha hecho muy famoso por ser una plataforma en la que la mayoría de aplicaciones son gratuitas. Por supuesto, este es un punto muy positivo para los usuarios, pero no tanto para los desarrolladores.

Otro punto clave al comparar los dos sistemas operativos como plataforma de desarrollo es el hecho de que Android es opensource. Al tratarse de software libre, implica que podemos crear y ejecutar cualquier aplicación en un terminal. Esto no pasa, con iOS, puesto que Apple limita las aplicaciones en su App Store según sus criterios. Además las limitaciones no se quedan ahí, si no que el SDK para iOS solo puede instalarse en un Mac, programas sobre Objetive C y, además, para mandar tu aplicación a que sea revisada para ver si puede entrar en la App Store, es imprescindible hacerlo desde un ordenador Mac.



Dejando de lado la rentabilidad del proyecto, finalmente la balanza se inclinó a favor de Android principalmente por dos razones:

- Las limitaciones que tuvimos con las herramientas de trabajo, puesto que ningún miembro del equipo de trabajo poseía ordenadores de Apple. Además de disponer de tres terminales con Android.
- Las principales y más importantes aplicaciones que emplean realidad aumentada junto a la localización utilizan como plataforma de desarrollo Android frente a iOS.

2. Android OS

En este apartado intentaremos resumir los conocimientos mínimos que se deben adquirir para tener una noción de lo que es Android y sus ventajas. También introduciremos conceptos clave a muy alto nivel importantes para el desarrollo de una aplicación cualquiera.

2.1 Desarrollando una aplicación Android

La plataforma de Android proporciona diferentes componentes a la hora de programar en función del objetivo de tu aplicación. Android provee cuatro tipos diferentes de componentes:

- **Activity:** Una actividad es el componente más usado en las aplicaciones Android. Típicamente una actividad representa una pantalla individual en el terminal y presenta una interfaz gráfica al usuario. Por ejemplo, en una aplicación de listado de teléfonos utilizaríamos dos actividades. Una para mostrar el listado de nombres y teléfonos y la segunda, para mostrar la información detallada del contacto seleccionado. La navegación entre las pantallas se realiza iniciando nuevas actividades. Cuando una actividad es abierta, la actividad previa es puesta en pausa y agregada el '\history stack' y no volverá al estado de ejecución hasta que vuelva a ser invocada.
- **Services:** Un servicio no tiene interfaz gráfica, pero puede ejecutarse en '\background' por un tiempo indefinido (se asemeja mucho al demonio de los sistemas Linux). Por ejemplo, podemos utilizar un servicio para que vaya capturando cada cierto tiempo la posición GPS y nos avise cuando estemos cerca de algún amigo. Mientras tanto el usuario puede seguir realizando otras tareas.



- **Broadcast receivers:** Este tipo de componentes se utilizan para recibir y reaccionar ante ciertas notificaciones broadcast. Este tipo de componentes no tienen interfaz gráfica y pueden reaccionar ante eventos como cambio de zona horarias, llamadas, nivel de batería ... Todos los receivers heredan de la clase base BroadcastReceiver.
- **Intent:** Este tipo de componentes es una clase especial que usa Android para moverse de una pantalla a otra. Un Intent describe lo que una aplicación desea hacer. Cualquiera activity puede reutilizar funcionalidades de otros componentes con solo hacer una solicitud en la forma de Intent.

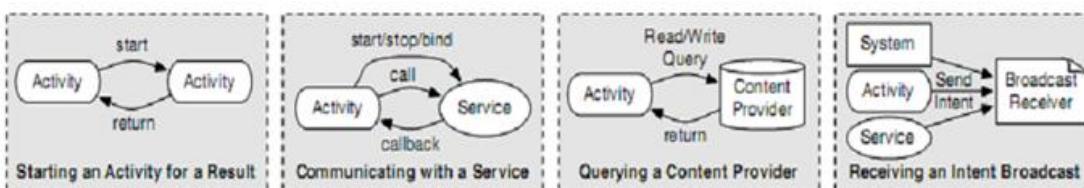


Figura 1: Flujo de componentes de Android OS

2.1.1 Ciclo de vida de una aplicación Android

En la mayoría de los casos, una aplicación Android ejecuta dentro de su propio proceso Linux. El proceso es creado para ejecutar el código de la aplicación y es el sistema quien pedirá y reclamará su memoria para reasignarla a otra aplicación.

Una característica peculiar en Android es que el tiempo de vida de un proceso no es controlado directamente por la aplicación. Es el sistema quien decide y determina el tiempo de vida basándose en el uso y capacidades del sistema. Para determinar que procesos deberían ser eliminados ante una condición baja de memoria, Android prioriza los procesos bajo una jerarquía para asignar a cada proceso una importancia en el sistema.

Existen diferentes procesos de acuerdo a esta jerarquía:

- **Foreground Process:** Es un proceso de primer plano que aloja una activity en la pantalla y con la que el usuario está interactuando (su método onResume() ha sido llamado) ó que un IntentReceiver está ejecutándose. Este tipo de procesos serán eliminados como último recurso si el sistema necesitase memoria.



- **Visible Process:** Es un proceso que aloja una activity pero no está en primera plano (su método onPause() ha sido llamado). Esto ocurre en situaciones donde la aplicación muestra una cuadro de dialogo para interactuar con el usuario. Este tipo de procesos no será eliminado a caso que sea necesaria la memoria para mantener a todos los procesos del primer plano corriendo.
- **Service Process:** Es un proceso que aloja un service que ha sido iniciado con el método startService(). Este tipo de procesos no son visibles y suelen ser importantes para el usuario (conexión con servidores, reproducción de música).
- **Background Process:** Es un proceso que aloja una activity que no es actualmente visible para el usuario (su método onStop() ha sido llamado). Normalmente la eliminación de estos procesos no suponen un gran impacto para la actividad del usuario. Es muy usual que existan numerosos procesos de este tipo en el sistema, por lo que el sistema mantiene una lista para asegurar que el último proceso visto por el usuario sea el último en eliminarse en caso de necesitar memoria.
- **Empty Process:** Es un proceso que no aloja ningún componente. La razón de existir de este proceso es tener una caché disponible de la aplicación para su próxima activación. Es común, que el sistema elimine este tipo de procesos con frecuencia para obtener memoria disponible

El ciclo de vida de una Activity contiene métodos relacionados con eventos para manejar los cambios de estado de cada Actividad de la aplicación. Cuando creamos una Actividad en nuestra aplicación debemos redefinir el método heredado onCreate(), ya que es el punto de partida de cada Actividad.

Los métodos heredados, de la clase Activity, para controlar los estados del ciclo de vida de una Actividad tienen una función concreta y se presentan en una fase determinada del ciclo de vida.

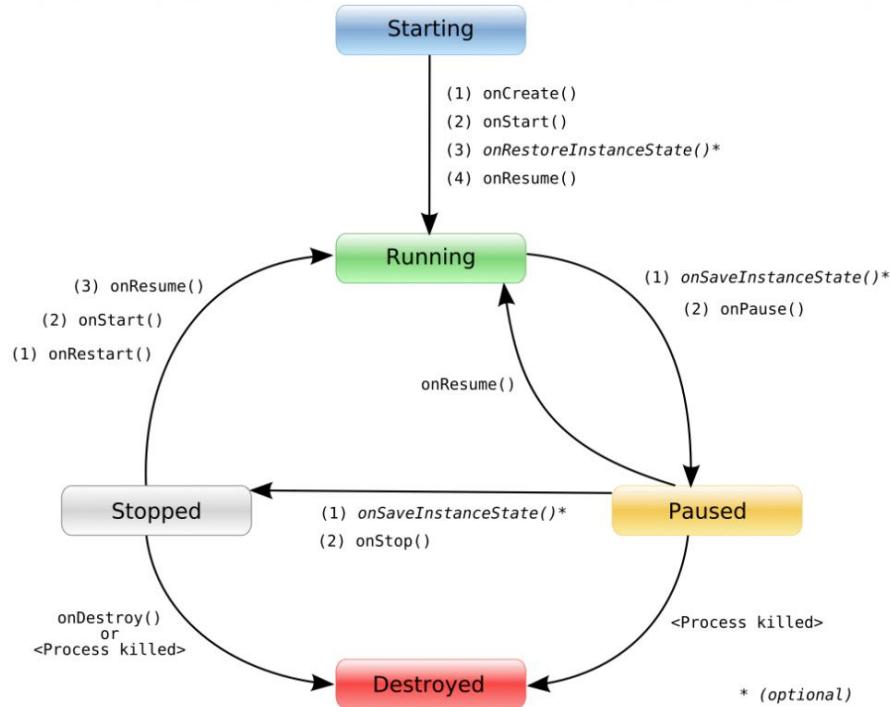


Figura 2: Estados de una Actividad de Android

A continuación se detallan los métodos del ciclo de vida que heredamos de la clase Activity:

- **onCreate(Bundle)**: se invoca cuando la Actividad se arranca por primera vez. Se utiliza para tareas de inicialización a realizar una sola vez, como crear la interfaz de usuario de la Actividad. Su parámetro es null o información de estado guardada previamente por onSaveInstanceState().
- **onStart()**: se invoca cuando la Actividad va a ser mostrada al usuario.
- **onResume()**: se invoca cuando la Actividad va a empezar a interactuar con el usuario.
- **onPause()**: se invoca cuando la Actividad va a pasar al fondo porque otra Actividad ha sido lanzada para ponerse delante. Se utiliza para guardar el estado persistente de la Actividad.
- **onStop()**: se invoca cuando la Actividad va a dejar de ser visible y no se necesitará durante un tiempo. Si hay escasez de recursos en el sistema este método puede llegar a no ser invocado. El ciclo de vida de una actividad se destruye directamente.



- **onRestart():** se invoca cuando la Actividad va a salir del estado de parada para volver a estar activa.
- **onDestroy():** se invoca cuando la Actividad va a ser destruida. Si hay escasez de recursos en el sistema este método puede llegar a nula implementación de la clase Activity ya guarda todo el estado de todos los componentes de la interfaz de usuario.
- **onRestoreInstanceState(Bundle):** se invoca para recuperar el estado guardado por onSaveInstanceState(Bundle). Normalmente no necesita ser redefinido porque la implementación de la clase Activity ya recupera todo el estado de todos los componentes de la interfaz de usuario.

2.1.2 El archivo Manifest

Cuando uno se encuentra con su primer proyecto Android o con los posteriores, va a encontrar un archivo suelto entre todo el árbol de paquetes, carpetas y otras hojas. Ese archivo tan llamativo y que se denomina `AndroidManifest.xml`, es uno de los archivos más importantes de nuestra aplicación.

Este XML se genera automáticamente al crear un proyecto y en él se declaran todas las especificaciones de nuestra aplicación. Cuando hablamos de especificaciones hacemos mención a las Activities utilizadas, los Intents, bibliotecas, el nombre de la aplicación, el hardware que se necesitará, los permisos de la aplicación, etcétera.



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

    <application >

        <activity>
            <intent-filter>
                <action />
                <category />
            </intent-filter>
        </activity>

        <activity>
        </activity>

        <meta-data>
        </meta-data>

        <supports-screens
        </supports-screens>

    </application>

    <uses-sdk
    </uses-sdk>

    <uses-library >
    </uses-library>

    <uses-permission >
    </uses-permission>

    <uses-feature >
    </uses-feature>

</manifest >
```

Figura 3: Estructura del Manifest

Vemos la estructura por partes:

- **<manifest>**: Dentro de este tag encontramos la referencia al número de versión de desarrollo de nuestro programa, su número de versión, y el paquete con el cual se referencia nuestra aplicación en el Android Market y el teléfono.



- **<application>**: Aquí adentro van todas las Activities, Services, Providers, Receivers y las bibliotecas que se usan en nuestra aplicación.
- **<activity>** Se tendrá un activity por cada pantalla de nuestra aplicación y dentro de él van sus atributos y los distintos Intents para comunicarse.
- **<supports-screens>**: Este tag es utilizado para describir las pantallas soportadas por nuestra aplicación.
- **<uses-permissions>**: Mediante este Tag especificamos los permisos que va a necesitar nuestra aplicación para poder ejecutarse, además son los que deberá aceptar el usuario antes de instalarla. Por ejemplo, si se desea utilizar funcionalidades con Internet o el vibrador del teléfono, hay que indicar que nuestra aplicación requiere esos permisos.
- **<uses-sdk>**: En este tag determinamos las distintas versiones Android que va a utilizar nuestra aplicación, tanto sobre qué versiones va a correr como qué versión fue utilizada para realizar nuestras pruebas. Mediante el atributo `android:minSdkVersion` establecemos a partir de qué versión de Android nuestra aplicación podrá correr.

Tener la información del Manifest actualizada, correcta y ordenada ayudará mucho a la hora de mantener nuestra aplicación y su correcto funcionamiento, pero por sobre todas las cosas ayudará a que sea visible en el Android Market el cual, muestra a los teléfonos las aplicaciones sólo si estos cumplen con los requisitos de hardware y software especificados por la aplicación en su *manifest*. Es decir, si tenemos mal la información del *manifest* nuestra aplicación en el mejor de los casos podrá ser descargada por dispositivos que no la pueden ejecutar correctamente dando como resultado puntuaciones y comentarios negativos o directamente no aparecerá para ser descargada.



2.1.3 Interfaz de usuario

En una aplicación de Android, la interfaz de usuario se crea utilizando los objetos View y ViewGroup. Hay muchos tipos de View y ViewGroup, cada uno de los cuales es descendiente de la clase View.

Las View son las unidades básicas de expresión de interfaz de usuario en la plataforma Android. La clase View sirve como base para las subclases llamados "widgets", que ofrecen plena aplicación de objetos de interfaz de usuario, como campos de texto y botones. La clase ViewGroup sirve como base para las subclases llamadas "diseños", que ofrecen diferentes tipos de arquitectura de diseño, como la relación lineal, tabular y.

Un objeto View es una estructura de datos cuyas propiedades almacenar los parámetros de diseño y el contenido de una zona específica rectangular de la pantalla. Un objeto View se encarga de su propia medición, el diseño, el dibujo, el enfoque cambia, el desplazamiento, y la clave de las interacciones entre gesto para el área rectangular de la pantalla en la que reside. Como un objeto en la interfaz de usuario, una vista es también un punto de interacción para el usuario y el receptor de los eventos de interacción.

- **Jerarquía de Views**

En la plataforma Android, se define la interfaz de usuario de una actividad, usando una jerarquía de View y ViewGroup, como se muestra en el diagrama. Este árbol de jerarquía puede ser tan simple o tan complejo como es necesario que sea, y se puede construir utilizando el conjunto de widgets de Android y diseños predefinidos, o con Views personalizadas que se crean.



Smartphone Mobile UI elements

v.0.3 - August 11, 2010

filament group inc.

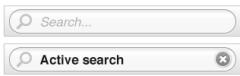
Buttons



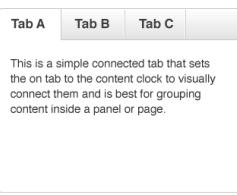
Toolbar



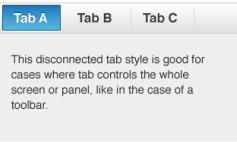
Search bar



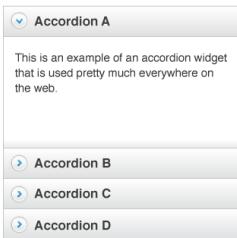
Connected tabs



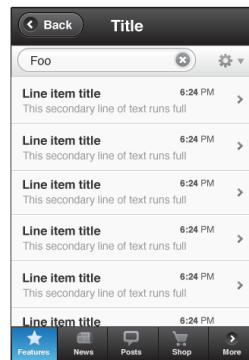
Disconnected tab bar



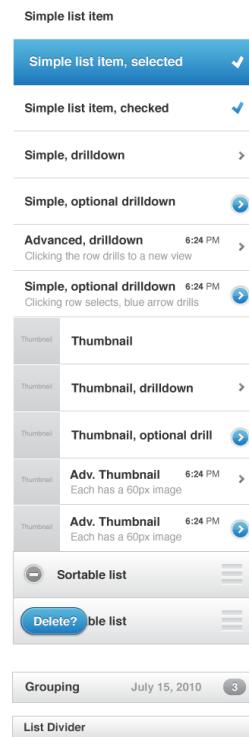
Accordion



List view



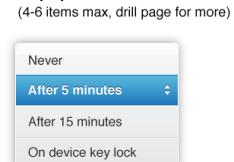
List item variations



Edit panel & form elements



Popup menu



Dialog



Figura 4: Interfaz de usuario - elementos UI

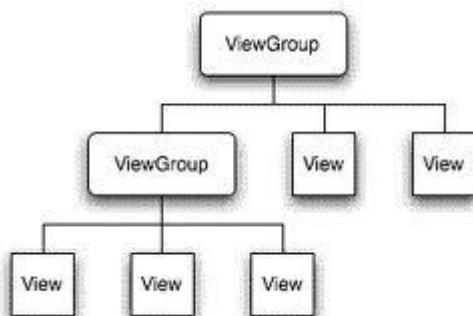


Figura 5: Jerarquía de Views

Con el fin de fijar el árbol de jerarquía de la View a la pantalla para la representación, su actividad debe llamar a la `setContentView ()` y pasar una referencia al objeto nodo raíz. El sistema Android recibe esta referencia y lo utiliza para invalidar, medir y dibujar el árbol. El nodo raíz de la jerarquía solicita a sus nodos hijo que se dibujen ellos mismos, a su vez, cada nodo de ViewGroup es responsable de llamar a cada uno de sus View hijos para que se dibujen ellos mismos. Los hijos pueden solicitar un tamaño y ubicación dentro de la matriz, pero el objeto principal tiene la decisión final sobre el tamaño de cada hijo. Android analiza los elementos de la jerarquía en orden (desde la raíz del árbol), instancias las Views y agregándolas a su parent (s). Debido a que se dibujan en orden, si hay elementos que se superponen las posiciones, el último que se instancia es el que se encuentre en la capa superior a los demás.

• Layouts

La manera más común para definir el diseño de la interfaz de usuario y expresar la jerarquía de Views es con un archivo de formato XML. XML ofrece una estructura legible para el diseño, al igual que HTML. Cada elemento de XML es un objeto View o ViewGroup (o descendientes de ellos). Las Views son las hojas en el árbol, los ViewGroup son ramas del árbol (ver la figura Ver Jerarquía de arriba).

El nombre de un elemento XML es relativa a la clase Java que representa. Por lo tanto un elemento `<TextView>` crea una TextView en la interfaz de usuario, y un elemento `<LinearLayout>` crea un grupo de vistas LinearLayout. Cuando se carga un recurso de diseño, el sistema Android inicializa estos objetos en tiempo de ejecución, correspondientes a los elementos de su proyecto.



```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >
    <TextView android:id="@+id/text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a TextView" />
    <Button android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello, I am a Button" />
</LinearLayout>
```

Figura 6: Layout en XML

Aunque un layout dentro de una aplicación será customizable por el usuario, por lo general se usan ciertos tipos predefinidos que cubren la mayoría de las necesidades comunes de cualquier aplicación Android destinada a un smartphone. Los principales tipos de layouts que se usan en las aplicaciones son los siguientes:

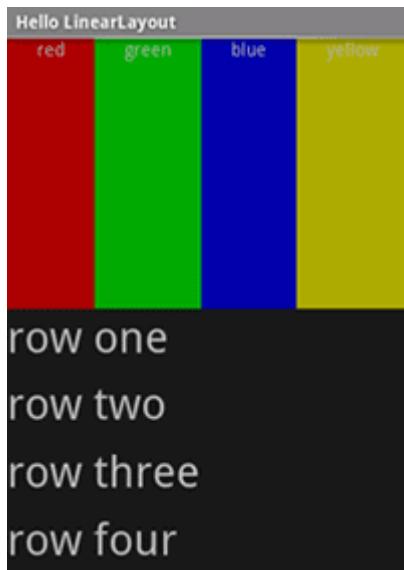


Figura 7(a): Linear Layout

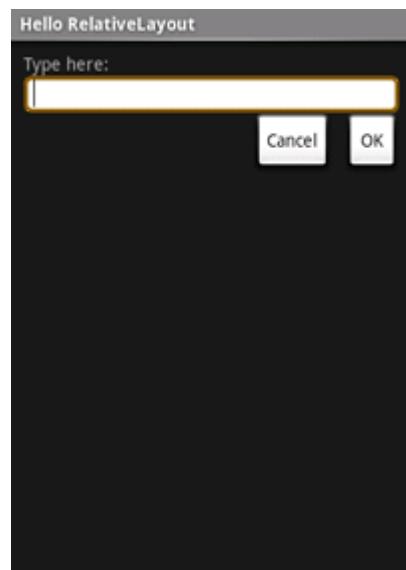


Figura 7(b): Relative Layout

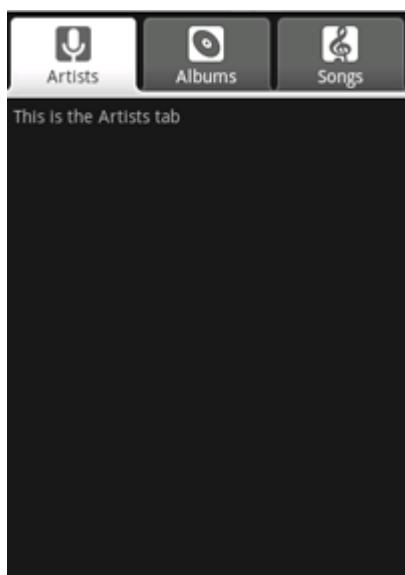


Figura 7(c): Tab Layout



Figura 7(d): List View

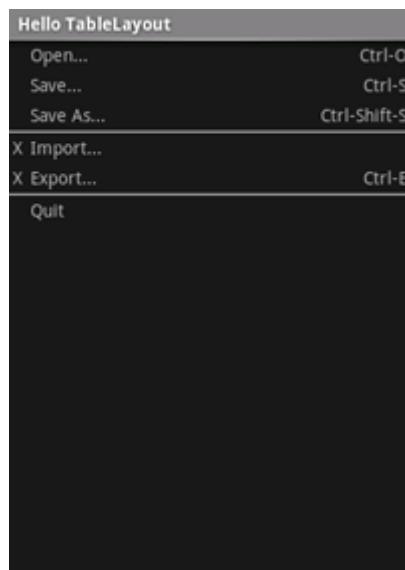


Figura 7(e): Table Layout

- **Widget**

Un widget es un objeto View que sirve como interfaz para la interacción con el usuario. Android proporciona un conjunto de widgets totalmente implementado, como botones, casillas de verificación, y campos de entrada de texto, así usted puede construir su interfaz de usuario. Algunos widgets proporcionados por Android son más complejos, como un selector de fechas, un reloj, y controles de zoom. Pero usted no está limitado a los tipos de widgets que ofrece la plataforma Android. Si desea hacer algo más personalizado y crear su propia elementos recurribles,



puede, mediante la definición de su propio objeto View o mediante la ampliación y la combinación de widgets existentes.



Figura 8: Widgets en Android OS

- **Eventos de usuario**

En la interfaz de cualquier aplicación, se necesitará poder captar la interacción del usuario con cada View o widget que contenga, y realizar acciones que se consideren. Para ser informado de los acontecimientos sobre interfaz de usuario, existen dos opciones: definir un detector de eventos y registrar con la View o bien reemplazar a un método de devolución de llamada existente para la View.



Figura 9: Eventos de usuario - TouchEvent



La primera opción es la más habitual y está basada en que la clase View contiene una colección de interfaces anidadas llamadas On<nombre_cualquiera>Listener, cada una con un callback llamado On<nombre_cualquiera>(). Por ejemplo, View.OnClickListener (para controlar los "clicks" en una View), View.OnTouchListener (para el control de los eventos sobre la pantalla táctil), y View.OnKeyListener (para la gestión de prensas dispositivo dentro de una vista). Así si se quisiera capturar un "clic" sobre una View (un botón por ejemplo) usaríamos OnClickListener, definiríamos su acción dentro de su método onClick () y lo registraríamos en View con setOnClickListener () .

La segunda forma es la mejor elección cuando la View con la que trabajamos la hemos implementado nosotros mismos y queremos capturar eventos específicos que ocurran sobre ella. Como por ejemplo eventos cuando se toca la pantalla (onTouchEvent ()), cuando la rueda de desplazamiento del dispositivo se mueve (onTrackballEvent ()), o cuando una tecla se presiona (onKeyDown ()). Esto permite definir el comportamiento por defecto para cada evento dentro de una View personalizada y más interesante aún determinar si delegarlo a otra View hija.

- **Menurización**

Los menús de aplicaciones son otra parte importante de la interfaz de usuario de una aplicación. Los menús ofrecen una interfaz fiable que muestran las funciones de la aplicación y su configuración. El menú de aplicación más común se desplegará con la tecla MENU del dispositivo. Sin embargo, también se pueden agregar menús contextuales, que pueden desplegarse cuando el usuario presiona y sostiene sobre un elemento.

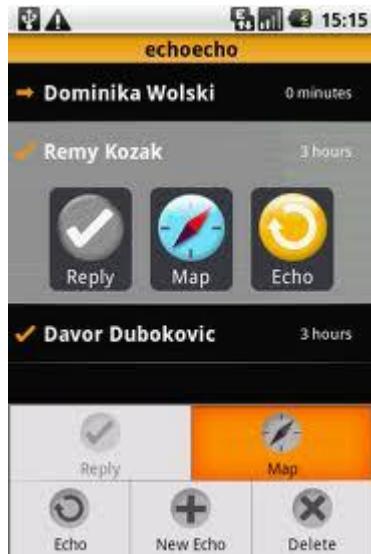


Figura 10(a): Menú estándar



Figura 10(b): Menú contextual

Los menús también están estructurados mediante una jerarquía de Views, pero en este caso no es posible definir la estructura a gusto del programador. En su lugar, se define el `onCreateOptionsMenu ()` u `onCreateContextMenu ()`, que son los métodos de devolución de llamada para la correspondiente actividad y para declarar los elementos que se deseen incluir en el menú. En el momento en el que se produzca el evento de usuario adecuado, Android creará automáticamente la jerarquía necesaria para el menú y dibujar cada uno de sus opciones de menú en el mismo.

Los menús también manejan sus propios eventos, así que no hay necesidad de registrar detectores de eventos en los elementos de su menú. Cuando un elemento se encuentra seleccionado, el método `onOptionsItemSelected ()` u `onContextItemSelected ()` serán llamados por el menú.

Y al igual que el diseño de la aplicación, existe la opción de declarar las opciones que se mostrarán al desplegarse un menú en un archivo XML.



2.1.4 Recursos de una aplicación

Recursos como imágenes y Strings de la aplicación, han de externalizarse de modo que usted puede mantener de forma independiente. Esto además permite ofrecer recursos alternativos que admiten configuraciones de dispositivos específicos, como idiomas o tamaños de pantalla, que se convierte en algo indispensable debido a la diversidad de dispositivos Android disponibles con diferentes configuraciones. Con el fin de proporcionar compatibilidad con diferentes configuraciones.

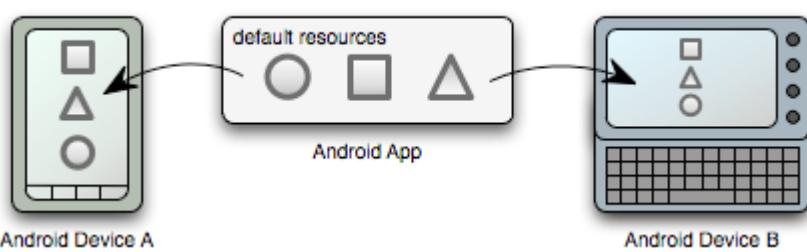


Figura 11: Dos dispositivos diferentes, tanto de los recursos por defecto de usar.

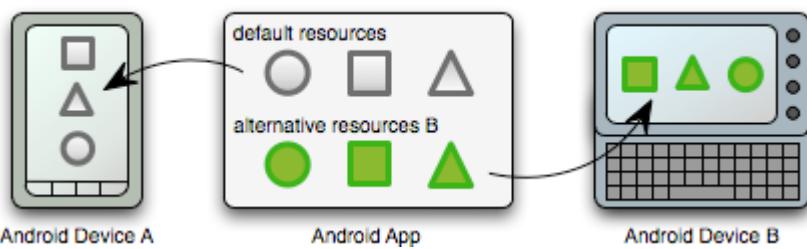


Figura 12: Dos dispositivos diferentes, uno con recursos alternativos.

Para cualquier tipo de recurso, puede especificar por defecto y múltiples recursos alternativos para su aplicación:

Recursos por defecto son los que deben ser utilizados con independencia de la configuración del dispositivo o cuando no hay recursos alternativos que coinciden con la configuración actual.

Recursos alternativos son aquellos que usted ha diseñado para utilizarse con una configuración específica. Para especificar que un grupo de recursos son para una configuración específica, anexar un calificador de la configuración apropiada para el nombre del directorio.

Por ejemplo, mientras que su diseño predeterminado de la interfaz de usuario se guarda en la res / layout / directorio, puede especificar un diferente diseño de interfaz de usuario que se utiliza cuando la pantalla está en orientación horizontal, guardándolo en la resolución / esquema de la tierra / de la guía.



Android se aplica automáticamente los recursos adecuados, haciendo coincidir la configuración actual del dispositivo a los nombres de los recursos de la guía.

2.2 Infografía de Android OS



Figura 13: Android – Infografía



3. OpenGL

Es una API multiplataforma y multilenguaje para la producción de software de gráficos 2D y 3D. Cuenta con una serie de funciones que permiten definir escenas complejas a partir de elementos sencillos. Tiene sus orígenes en el año 1992 cuando fue desarrollada por Silicon Graphics Inc.

El funcionamiento consiste en aceptar primitivas, como puntos y líneas, y convertirlas en píxeles. Este proceso se realiza en el pipeline del sistema, es ahí donde actúan gran parte de los comandos OpenGL, ya sea configurando su comportamiento o emitiendo primitivas.

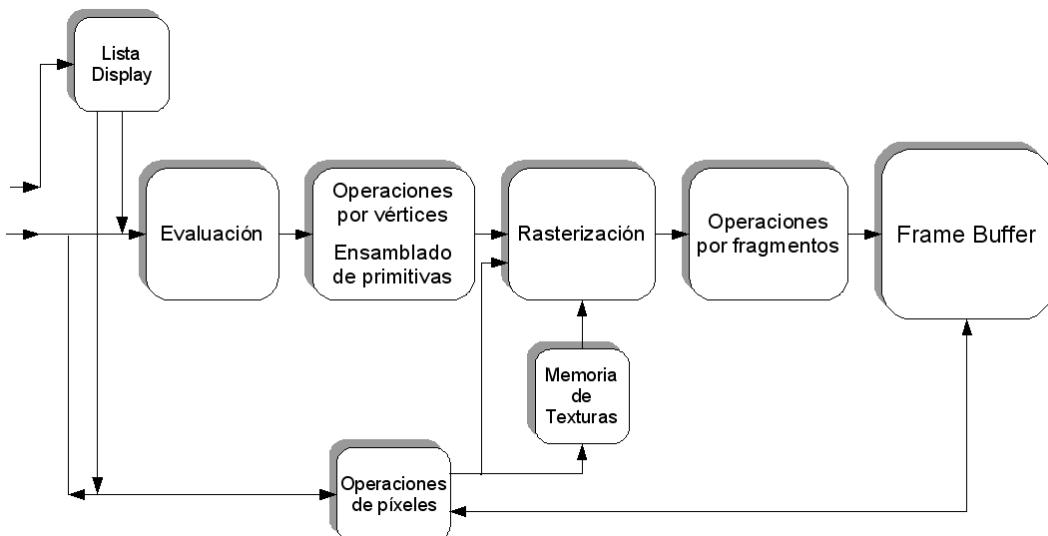


Figura 14: Pipeline de OpenGL

En dispositivos móviles se usa una versión especial llamada OpenGL ES (Embedded System). Tiene ciertas diferencias con la versión estándar, principalmente que restringe el número de funciones para hacer un uso más eficiente del sistema y también fija ciertos parámetros del hardware por el mismo motivo.

La mayor restricción se puede observar en la manera de volcar los vértices en el pipeline gráfico. Dónde en la versión estándar se puede ir mandando uno a uno, aquí es necesario enviar un buffer que contenga todos los vértices posibles y así limitar el número de operaciones de transferencia entre la CPU y la GPU.



Figura 15: Ejemplo que demuestra la potencia de OpenGL ES en móviles

Pese a estas restricciones se pueden conseguir grandes resultados debido al gran avance de los últimos terminales. Al contrario que las anteriores generaciones, estos incluyen hardware específico para la aceleración gráfica. Este hecho permite liberar a la CPU de una carga de trabajo realmente importante, permitiendo realizar otros cálculos como la lógica del juego, y también llevar a cabo las operaciones gráficas de una manera más eficiente.

Más en detalle

OpenGL es el entorno de desarrollo principal para aplicaciones portátiles, 2D interactivo y gráficos 3D. Desde su introducción en 1992, OpenGL se ha convertido en el API de gráficos 2D y 3D más utilizado y soportado por la industria, proporcionando miles de aplicaciones para una gran variedad de plataformas. OpenGL favorece la innovación y acelera el desarrollo de aplicaciones incorporando un amplio conjunto de funciones de renderizado, mapeado de texturas, efectos especiales, etc. Los desarrolladores pueden exprimir el potencial de OpenGL en todas las plataformas más utilizadas, asegurándose una gran compatibilidad.

Alta calidad visual y rendimiento

Cualquier aplicación visual que requiera un alto rendimiento, desde animación 3D a diseño CAD, puede sacar provecho de las capacidades de alta calidad y rendimiento de OpenGL. Esto permite a los desarrolladores de distintos mercados como televisión, CAD/CAM/CAE, entretenimiento, imágenes médicas, y realidad virtual producir y mostrar gráficos 2D y 3D increíblemente competentes.

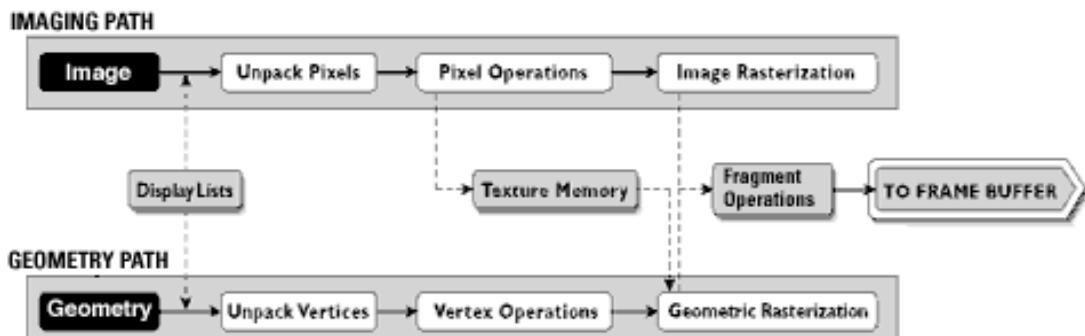


Ventajas para los desarrolladores

- Estándar para la industria: Un consorcio independiente, la mesa de revisión de arquitectura OpenGL, guía la especificación de OpenGL. Con un amplio soporte por la industria, OpenGL es el único estándar realmente abierto, independiente de fabricante y multiplataforma.
- Estable: Las implementaciones de OpenGL llevan disponibles desde hace más de siete años para una gran variedad de plataformas. Los añadidos a la especificación están bien controlados y las actualizaciones se anuncian con tiempo para que los desarrolladores puedan adoptar los cambios. La retro compatibilidad asegura que las aplicaciones existentes no se vuelven obsoletas.
- Seguro y portable: Todas las aplicaciones OpenGL producen resultados visuales consistentes en todas las plataformas que cumplan los requisitos de la API, independientemente del sistema operativo.
- Evolución: Por su diseño mirando al futuro, OpenGL permite que las nuevas innovaciones de hardware sean accesibles a través de la API mediante un mecanismo de extensión. De este modo se permite que los desarrolladores de aplicaciones y hardware incluyan dichas innovaciones dentro de su ciclo de producción normal.
- Escalable: Las aplicaciones basadas en el API de OpenGL pueden funcionar en sistemas que abarcan desde dispositivos portátiles hasta supercomputadores. Como resultado, las aplicaciones se pueden escalar a cualquier máquina que el desarrollador escoja como objetivo.
- Fácil de usar: OpenGL está bien estructurado con un diseño intuitivo y comandos lógicos. Las rutinas de OpenGL suelen dar como resultado aplicaciones con menos líneas de código que usando otras librerías de gráficos. Como añadido, los controladores de OpenGL encapsulan la información del hardware subyacente, liberando al desarrollador de tener que desarrollar características específicas para dicho hardware.
- Bien documentado: Se han publicado numerosos libros sobre OpenGL, y una gran cantidad de código de ejemplos está fácilmente disponible.



Pipeline de programación para OpenGL



OpenGL trabaja con datos de imágenes y con primitivas geométricas

Simplifica el desarrollo y acelera la producción

Las rutinas de OpenGL simplifican el desarrollo de software gráfico (desde renderizar un simple punto geométrico, una línea o un polígono relleno con la iluminación y texturizado más complejos). OpenGL provee a los desarrolladores acceso a primitivas geométricas y de imágenes, listas de muestreo, transformaciones, iluminación y texturizado, anti-aliasing, blending, y otras muchas características.

Cada implementación de OpenGL incluye la lista completa de funciones. El estándar revisado permite enlaces para C, C++, Fortran, Ada, y Java. Todas las implementaciones vienen con una sola especificación y un documento de enlace para lenguajes. Las aplicaciones que usan funciones OpenGL son fácilmente portables en muchas plataformas para aumentar la productividad.

Todos los elementos del estado de OpenGL (incluso contenidos de la memoria de texturas o el buffer de frames) pueden ser consultados por la aplicación. OpenGL también soporta la visualización de imágenes 2D tratadas como tipos primitivos que pueden ser manipulados como objetos geométricos 3D.

Disponible en todas partes

Soportado en todas las estaciones de trabajo UNIX®, y distribuido como estándar en cada sistema Windows y MacOS, no hay ninguna otra API de gráficos que trabaje en un conjunto tan grande de plataformas hardware y entornos software, OpenGL funciona en los principales sistemas operativos incluyendo Mac OS, OS/2, UNIX, Windows 95/98, Windows 2000, Windows NT, Linux, OPENStep, y BeOS; así como con todos los sistemas de ventanas



Win32, MacOS, Presentation Manager, y X-Window System. OpenGL es utilizable desde Ada, C, C++, Fortran, Python, Perl y Java y ofrece completa independencia de protocolos y topologías.

Pensado para flexibilidad y especialización: Extensiones

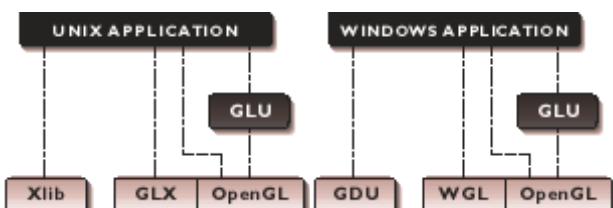
Aunque la especificación define un pipeline de procesamiento particular, los fabricantes de plataformas tienen la libertad de personalizar la implementación para satisfacer unos objetivos particulares. Algunas llamadas pueden realizarse en hardware dedicado, ejecutarse como rutinas normales en la CPU del sistema, o como una combinación de ambas. Esta flexibilidad de implementación permite una aceleración en cualquier sistema. Los desarrolladores tienen a su vez la seguridad de que obtendrán los mismos resultados independientemente de la plataforma que implemente el soporte OpenGL.

Usando el mecanismo de extensiones, los desarrolladores de hardware pueden diferenciar sus productos creando extensiones que permitan acceder a rendimiento adicional e innovaciones tecnológicas.

Muchas extensiones OpenGL, como las relacionadas con la API como GLU, GLX, y WGL, han sido definidas por grupos de fabricantes. El registro de extensiones de OpenGL está mantenido por SGI y contiene la especificaciones para todas las extensiones conocidas, modificaciones escritas y demás documentos. También define los convenios de nombres, guías para crear extensiones y demás.

Jerarquía de la API

Las aplicaciones OpenGL pueden usar el sistema de ventanas, entrada y eventos GLU que soporta quads, NURBS, polígonos complejos, utilidades para matrices y aún más ventajas



Este diagrama muestra la relación entre OpenGL GLU y las APIs de ventanas.



Gobierno

El OpenGL Architecture Review Board (ARB), era un consorcio independiente formado en 1992, que gobernaba el futuro de OpenGL, proponiendo y aprobando cambios en la especificación, nuevos lanzamientos y creando pruebas. En septiembre de 2006, el ARB se convirtió en el OpenGL Working Group bajo el consorcio del grupo Khronos para APIs estándar.

El OpenGL Performance Characterization Committee, otra organización independiente, crea y mantiene pruebas de OpenGL y publica los resultados en su web: www.specbench.org/gpc/opc.static/index.html.

Innovación continua

El estándar OpenGL está continuamente evolucionando. Las revisiones formales ocurren en intervalos periódicos, y las extensiones permiten hacer uso de los avances hardware. Cuando una extensión se acepta generalmente, son consideradas para su inclusión dentro del estándar OpenGL. Este proceso permite evolucionar de una manera controlada e innovadora.



IV

Características de FRAGUEL



1. Introducción

En anteriores apartados hemos tratado las diferentes aspectos que rodean a este proyecto. Hemos hablado de conceptos como la realidad aumentada y el posicionamiento e introducido Android OS como a varios niveles.

En este apartado presentamos las diferentes características de FRAGUEL y profundizamos en cada una de ellas.

Describimos la arquitectura del sistema a nivel de módulos y explicamos las líneas de comunicación que se establecen entre la parte cliente y la parte del servidor y cuando se producen.

Detallamos cómo y porqué se hace uso de los mapas en el sistema así como la localización y posicionamiento del usuario y los diferentes elementos que se desplegarán por el mapa. Igualmente exponemos el sistema creado para la gestión de rutas, su diseño y su funcionamiento.

Por último hablamos de todo el diferente contenido multimedia que se pondrá al alcance del usuario de la aplicación, a saber: *texto hablado*, *imágenes*, *vídeos* y *realidad aumentada*. Para cada uno de los contenidos se trata la forma en la que se integró en el sistema y qué funcionalidad se facilita.



2. Arquitectura

2.1 Servidor

Para el servidor preparamos una máquina personal en continuo funcionamiento y dedicada de forma exclusiva a esta labor.

El objeto del servidor es la de almacenar las rutas creadas en el sistema por los usuarios y hacerlas accesibles a otros usuarios que quieran descargarlas en su sistema.

Como interfaz para el usuario se ha creado una página web oficial del proyecto FRAGUEL que proporcionará este servicio.

De esta forma un usuario que quiera compartir una ruta personal podrá añadirla, a través de la web desde el mismo terminal, a la lista de rutas compartidas en la aplicación, y su vez consultar otras rutas disponibles por otros usuarios y descargarla en su sistema.

2.2 Cliente

Cada terminal Android que tenga instalada la aplicación FRAGUEL será considerado cliente. Los clientes podrán descargar rutas nuevas y los archivos de realidad aumentada asociados a las mismas. Además el cliente descargará del servidor las imágenes de las rutas, guardando una copia local en la tarjeta de memoria externa de las mismas.

Otra funcionalidad añadida en el cliente es la posibilidad de subir al servidor las propias rutas que cree, lo que hace que la aplicación sea dinámica y esté continuamente en desarrollo.

3. Mapas y posicionamiento

Sin duda, la principal característica de FRAGUEL, que sienta las bases a la hora de definir la funcionalidad de la aplicación, es el uso de la localización de elementos sobre mapas reales.

A través de la aplicación, el usuario podrá identificar sus puntos de interés preferidos sobre el mapa, realizar rutas guiadas a través de estos y acceder su contenido multimedia de forma rápida, simple e intuitiva.



3.1 Mapas

Al basar la principal funcionalidad de la aplicación sobre la localización de diferentes puntos del mundo real, los mapas serán el “tablero” en el que jugar.

Hoy en día Google Maps es una de las aplicaciones de posicionamiento más usadas ya sea a través de la web como en aplicaciones nativas en los terminales móviles. Y hemos querido aprovechar todo el potencial que ofrece adaptándolo a nuestra aplicación.

Pero los problemas surgieron al preguntarnos cómo desarrollamos la integración de Google Maps y cómo llevar a cabo mediante programación lo siguiente:

- Cambiar las vistas de Google Maps
- Obtener la latitud y longitud de los lugares en Google Maps
- Realizar geocodificación y geocodificación inversa
- Agregar marcadores de Google Maps

Solo integrar la API en nuestra aplicación y poder usar la tile de mapas resultó más complicado de lo esperado puesto que es necesario solicitar una clave. Para solicitarla necesitamos extraer la huella digital MD5 que identifica únicamente nuestra aplicación. La introdujimos en la página de soporte de Google Maps <http://code.google.com/android/maps-api-signup.html> y obtuvimos un código único que permite a la aplicación acceder a la tile de mapas de Google Maps.

Por último tuvimos que modificar una vez más el Manifest.xml añadiendo los siguientes:

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

Que habilita el acceso de nuestra aplicación a internet para recuperar las imágenes de los mapas que se necesiten.

Finalmente para desplegar los mapas modificamos el layout de nuestra View para los mapas introduciendo <com.google.android.maps.MapView> más la clave única obtenida.



Figura 1: Mapas en FRAGUEL

3.2 Puntos de interés

Desde el punto de vista funcional, la idea detrás de cada punto de interés es que sean lugares del mundo real definidos por los usuarios que tengan sentido dentro de una ruta. Esta idea deja libertad al usuario para que decida qué es para él un punto de interés y en qué consiste una ruta.

De esta forma en la aplicación existirán puntos que vayan desde un museo o monumento en una ciudad, hasta un bar dónde se celebró un cumpleaños.



Figura 2: Puntos de interés

Desde un punto de vista más técnico, un punto de interés consiste en coordenadas que marcan un punto en el cual está lo que queremos localizar y que enlazarán con una serie de contenido multimedia. Cada uno de ellos tiene la siguiente estructura:

- **Identificador:** es un número entero que, entre otras cosas, designará el orden del punto dentro de una ruta.
- **Coordenadas:** los valores de latitud y longitud en los que está localizado. Es el único campo imprescindible.
- **Título:** nombre que tiene el punto de interés.
- **Descripción:** un texto que puede ser desde una breve introducción histórica hasta una lista de precios de un restaurante por ejemplo.



- **Icono:** imagen con la que aparece representado en el mapa el punto de interés. Es un campo opcional y si no se informa se le asignará uno por defecto.
- **Imágenes:** una lista url's que enlacen con imágenes asociadas al punto y visibles mediante una galería y descargables por el usuario. Cada imagen podrá llevar asociado un texto (pie de imagen).
- **Vídeo:** enlace a un vídeo subido en YouTube relacionado con el punto en cuestión. Además, al integrarnos con el popular portal de videos, facilitamos al usuario su mantenimiento.
- **Realidad aumentada:** un punto de interés tendrá la capacidad aumentar aún más la experiencia del usuario mediante la superposición de objetos tridimensionales en el mundo real a través de la cámara del teléfono. Por ejemplo, en un punto de interés que sea la “Puerta del Sol”, sería posible que un usuario visualizara el “Oso y el Madroño” en su posición original. Se deberá indicar las coordenadas GPS (latitud y longitud) así como la altura los ficheros que debe cargar separados por ‘,’ (coma) y el texto que se reproducirá.

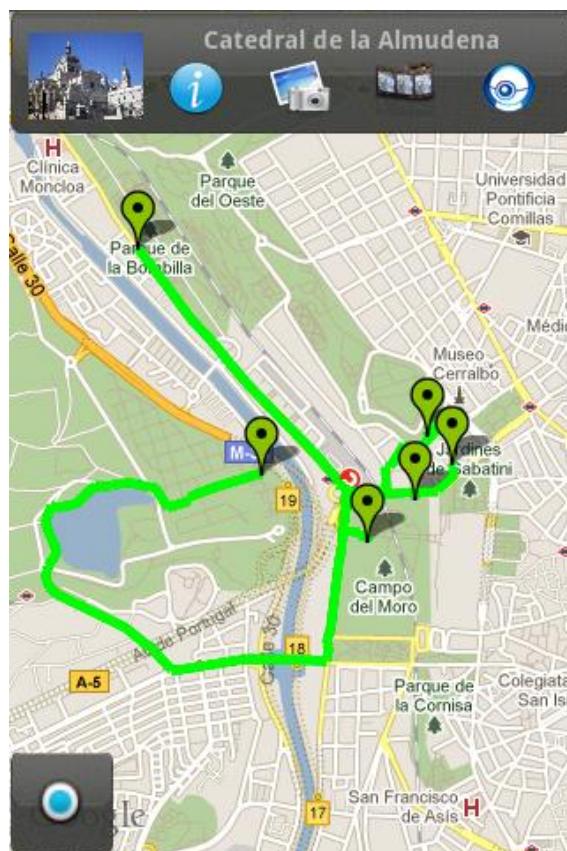


Figura 3: Puntos de Interés – Opciones del punto



Figura 4: Realidad Aumentada en la Puerta del Sol

Con el fin de facilitar la parametrización un punto de interés que quiera integrar en la aplicación, hacemos uso de una plantilla XML para cada ruta y cada punto que la forma. De esta forma con solo añadir dicho fichero informado con los campos que se considere y haciendo uso del parser SAX2 para XML, podremos alimentar la aplicación con nuestras propias rutas y puntos de interés haciendo uso de toda la funcionalidad de la aplicación sin requerir conocimientos técnicos y al alcance de cualquier tipo de usuario.

```
<point id="5">
    <coords x="40.416921" y="-3.703642"/>
    <title>Puerta del Sol</title>
    <pointdescription>La Puerta del Sol es una plaza de Madrid (España). Aquí se encuentra desde 1950 el denominado Kilómetro Cero de las carreteras nacionales. En su centro se encuentra la emblemática estatua del Oso y el Madroño, símbolo de la ciudad de Madrid. La plaza es un importante centro neurálgico de la capital, con numerosos edificios históricos y modernos alrededor, así como numerosos comercios y servicios. Es un lugar muy concurrido tanto por turistas como por residentes de la ciudad.</pointdescription>
    <pointicon>http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_icono.jpg</pointicon>
    <images>
        <image url="http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_imagen0.jpg">Fachada principal de la Casa de Correos, en la Puerta del Sol.</image>
        <image url="http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_imagen1.jpg">Campanadas en la Puerta del Sol el día de Nochevieja.</image>
        <image url="http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_imagen2.jpg">Estatua del Oso y el Madroño (Símbolo de la ciudad Madrid)</image>
        <image url="http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_imagen3.jpg">Anochecer en la Puerta del Sol.</image>
        <image url="http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_imagen4.jpg">Estampa antigua de la Puerta del Sol de Madrid.</image>
        <image url="http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_imagen5.jpg">Fuente y estatua ecuestre de Carlos III.</image>
        <image url="http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_imagen6.jpg">Antigua placa del kilómetro cero (1950-2009).</image>
        <image url="http://www.blackmesa.es/fraguel/imagenes/ruta2_punto5_imagen7.jpg">Vista aérea de la plaza.</image>
    </images>
    <video>http://www.youtube.com/watch?v=52UlyDht_9w</video>
    <ar lat="0.0" long="0.0" alt="0.0" file="oso.obj,oso2.obj">La escultura siempre ha estado en la Puerta del Sol, aunque con dos ubicaciones diferentes.</ar>
</point>
```

Figura 5: estructura XML de punto de interés



3.3 Rutas

Como hemos dicho en el punto anterior, si la idea detrás de cada punto de interés es que sean lugares definidos por los usuarios en cualquier ámbito, una ruta no será más que la unión de un grupo de estos puntos ordenados como se quiera.

Desde el punto de vista del usuario final de la aplicación, una ruta se puede ver como una pequeña historia real que se quiere transmitir a otros usuarios. Los límites de estas historias estarán acotados por la imaginación de los propios usuarios que creen la ruta. Para expresar esta diversidad que soporta la aplicación FRAGUEL, se incluyen dos ejemplos de rutas: "Ruta Cultural" y "Ruta de Ocio".

- **Ruta Cultural:** esta ruta tiene el objetivo de exemplificar el uso de la aplicación como un medio a través del cual definir una ruta turística totalmente guiada al usuario.
- **Ruta de Ocio:** muestra el potencial de la aplicación como herramienta para contar las experiencias personales de un usuario un día de saliendo por Moncloa.



Figura 6: Galería de imágenes de Ruta Cultural



Rutas descargadas en su terminal 'Monumentos'	
	Teatro Real de Madrid
	Palacio Real de Madrid
	Catedral de la Almudena
	Plaza de la Villa
	Plaza Mayor de Madrid
	Puerta del Sol

Figura 7: Lista de puntos de Ruta Cultural

Cada ruta que se defina para su uso estará formada por los siguientes elementos:

- **Identificador:** es un número entero que identifica únicamente a la ruta.
- **Nombre:** nombre que tiene la ruta.
- **Descripción:** texto que explica de manera breve el contenido de la ruta.
- **Icono:** imagen con la que aparecerán representados en el mapa cada uno de los punto de interés que formen parte de la ruta. Es un campo opcional y si no se informa se le asignará el icono que esté definido para cada punto o en su ausencia uno por defecto.
- **Lista de puntos de interés:** cada uno de los puntos de interés que formen parte de la ruta en cuestión. Así mismo, características propias de la ruta, como el orden de los puntos por ejemplo,

Como ya se anticipó en el punto anterior, una de las claves del éxito de una aplicación como FRAGUEL radica en lo bien alimentada que se encuentre de contenido, en este caso rutas. Por ello, para facilitar al usuario definir sus



propias rutas con rapidez y sin necesidad de conocimientos técnicos, únicamente es necesario informar una plantilla XML e introducirla en la aplicación.

```
<?xml version="1.0" encoding="UTF-8"?>
<route id="1" version="1.0">
    <name>Un viernes por Moncloa</name>
    <description>Un viernes, tras toda la semana sólo queda divertirse por Madrid. Este es el recorrido de aquel día, para aquel que aún no tenga j
    <icon>http://www.blackmesa.es/fraguel/imagenes/ruta1.jpg</icon>
    <points>
        <point id="0">
            <coords x="40.4327106" y="-3.7178161" />
            <title>Manolo Bar</title>
            <pointdescription>El emblemático bar Manolo, un sitio perfecto donde tomar unas cañas con los amigos. \n \n Restaurante tradicional y
            <pointicon>http://www.blackmesa.es/fraguel/imagenes/ruta1_punto0_icono.jpg</pointicon>
            <images>
                <image url="http://www.blackmesa.es/fraguel/imagenes/ruta1_punto0_imagen0.jpg">Fotografía del logotipo de Manolo Bar</image>
                <image url="http://www.blackmesa.es/fraguel/imagenes/ruta1_punto0_imagen1.jpg">Tapa de jamón ibérico típica a un atractivo precio:
            </images>
            <video>http://www.youtube.com/watch?v=3CZ-k3qyk2A</video>
            <ar lat="0.0" long="0.0" alt="0.0" file=""></ar>
        </point>
        <point id="1">
            <coords x="40.4352261" y="-3.7183039" />
            <title>Van Gogh Cafe</title>
            <pointdescription>Van Gogh Café se inauguró en Marzo de 2.005 para convertirse en un "lugar de encuentro en el corazón de Moncloa". Ofre
            <pointicon>http://www.blackmesa.es/fraguel/imagenes/ruta1_punto1_icono.jpeg</pointicon>
            <images>
                <image url="http://www.blackmesa.es/fraguel/imagenes/ruta1_punto1_imagen0.jpg">Desayuno típico andaluz en Van Gogh Café, pan tumac
                <image url="http://www.blackmesa.es/fraguel/imagenes/ruta1_punto1_imagen1.jpg">Fachada exterior del bar-café Van Gogh.</image>
                <image url="http://www.blackmesa.es/fraguel/imagenes/ruta1_punto1_imagen2.jpg">Zona interior del café, con réplicas de los cuadros
            </images>
            <video>http://www.youtube.com/watch?v=cxLG2wtE7TM</video>
            <ar lat="0.0" long="0.0" alt="0.0" file=""></ar>
        </point>
    </points>
</route>
```

Figura 8: Estructura XML de una ruta



3.4 Gestión de rutas

Ya hemos visto que FRAGUEL ofrecerá historias o rutas en un entorno real, pero ¿cómo guiamos a un usuario a través de ellas? ¿cómo conseguimos que seguir una ruta sea fácil, cómodo y atractivo? Esta es una pregunta que nos hemos hecho y hemos trabajado para hacerlo posible. Sin duda la gestión de las rutas incide directamente en lo usable o no que es nuestra aplicación.

Cuando se piensa cual es la mejor forma de guiar a un usuario a un lugar mediante un dispositivo móvil, aparecen las siguientes directrices clave:

- La transición de un punto a otro tiene que estar guiada e informada en todo momento.
- El usuario debe tener libertad de decisión y de movimiento.
- Debe existir la posibilidad de empezar una ruta desde cualquier punto desde cualquier lugar.

Todos estos puntos y además de otros se han tenido en cuenta a la hora de ofrecer al usuario una ruta guiada.

Cuando un usuario escoja una ruta que seguir se encontrará con las siguientes posibilidades:

1. Nada más empezar se le hará una introducción hablada sobre la ruta, y se le mostrará la ruta en el mapa, mostrando el título de la misma, una pequeña imagen y una descripción de lo que el usuario visitará.



Figura 9: Introducción de la ruta

2. Mediante el popup que aparece sobre el mapa, se guiará al usuario hasta el primer punto de interés. Se mostrará información de la distancia, la dirección que debe tomar y además se dibujará sobre el mapa la mejor forma de llegar yendo a pie.



Figura 10: Siguiendo una ruta



3. Puesto que controlamos la situación del terminal y cada punto de interés, haremos saltar un mensaje con vibración que alerte al usuario cuando se encuentre próximo a un punto. Así evitamos que tenga que estar con el dispositivo en la mano en todo momento.
4. Al llegar a cada punto de interés se introducirá al usuario con una presentación hablada las principales características y se le dará acceso a todo su contenido multimedia.



Figura 11: Introducción de punto de interés

5. Una vez se visiten todos los puntos o bien cuando el usuario decida abandonar la ruta, se volverá al modo libre haciéndose visibles de nuevo todos los puntos de todas las rutas que se encuentren disponibles en la aplicación.



3.5 Localización y geotagging

El software implementado hace uso de los sensores del terminal para establecer la localización del usuario. Los mecanismos utilizados para determinar dicha posición son el GPS y la triangulación por antenas de telefonía y WIFI. El propio sistema Android se encarga de decidir cuales utilizar en función de cuales están disponibles en cada momento.

Sin embargo, pese a la combinación de estos para minimizar el margen de error, hay algunos problemas al estar cerca de edificios o con algún tipo de estructura de gran tamaño sobre nosotros. Este es un problema general a todos los aparatos que utilizan localización por satélite y que es incluso más sensible en el caso de la triangulación de antenas. En este último caso hay grandes progresos ya que las últimas tecnologías móviles permiten distinguir las ondas directas de las rebotadas por el entorno y agruparlas para obtener una mayor señal.

Con esta posición obtenida podemos introducir toda la información necesaria en el entorno, en nuestro caso los puntos de interés. La vista principal de esta información es el mapa 2D que muestra el plano circundante. Se puede adaptar para ver una fotografía satelital. Si el usuario se encuentra dentro del área de influencia de un punto de interés podrá acceder a toda la información contenida en el.

También es posible crear nuevos puntos de interés y rutas haciendo uso de la localización actual. De esta forma el usuario no se limita a observar los contenidos, si no que puede crear nuevos para poder compartirlos con el resto de usuarios de la aplicación.

4. Multimedia

Hemos visto en apartados anteriores el esqueleto sobre el que se sustenta la funcionalidad de la aplicación: mapas y posicionamiento. Sin embargo, todo esto se quedaría a medias si no se alimentara todo ello contenido multimedia.



Figura 12: FRAGUEL - Opciones multimedia

4.1 Imágenes

Permitir que el usuario pudiera asociar imágenes a cada punto de interés ha sido un objetivo claro desde el principio. No solo había que hacer que fuera intuitivo para un usuario que cree una ruta, sino había que encontrar la mejor forma de mostrarlas ya que al trabajar con sistemas empotrados, la memoria y el espacio de la pantalla están muy limitados.



Figura 13: Imágenes - Galería



Un usuario final que se encuentre realizando una ruta podrá acceder a la galería de imágenes de un punto de interés cualquiera con tan solo seleccionarlo con el dedo.

En un primer momento se abrirá una *Grid View* con todas las imágenes disponibles del punto, si pulsáramos en una imagen cualquiera la visualizaríamos a pantalla completa mediante una *ImageView* customizada en la que hacemos uso de la tecnología multitouch para hacer zoom sobre la imagen. Además existe la posibilidad de añadir texto asociado y por supuesto hacer uso de la librería TTS y reproducirlo desde el altavoz del móvil o a través de unos auriculares.

4.2 Vídeos



En FRAGUEL tendremos la posibilidad de incorporar multitud de contenido multimedia como autores de nuestras propias rutas y explotarlo como usuarios. Y por supuesto el video no faltará en esta aplicación.

En punto de interés en el que nos encontremos, podremos visualizar su vídeo con la misma facilidad que el resto del contenido, con un solo dedo.

En Android existe un bastante contenido de apoyo para el soporte de contenido multimedia, y el vídeo no es una excepción. Sin embargo, tras un exhaustivo análisis encontramos los siguientes motivos por los que no parecía encajar:

1. Realizar streaming. Tendríamos que incluirlo como recurso de la aplicación con las desventajas que ello implica. Además, queríamos tener la flexibilidad necesaria controlar el vídeo tras la solicitud.
2. Soporte de contenido en alta definición. Queríamos limitar el vídeo a un tamaño estándar ajustado a las características de cada dispositivo. El ancho de banda en los móviles actuales está todavía muy limitado, y ver un vídeo demasiado pesado podría ser un proceso tedioso.



3. Conflicto con los formatos de vídeo. Queríamos usar un formato de vídeo lo más estándar posible, que facilitara a los usuarios la conversión de sus vídeos.
4. Necesidad de un servidor propio . Del mismo modo que el resto de contenido multimedia que soporta la aplicación, almacenar y mantener cada vídeo que suba un usuario supone tener dedicado un servidor propio con el coste en tiempo que conlleva. Además, requiere abrir servicios web a los usuarios para la subida de contenido. Por no hablar de futuros problemas que puedan surgir por el mantenimiento, como por ejemplo tener que hacer un redimensionamiento del espacio a medida que éste aumentara.

Tras valorar todo esto, llegamos a la conclusión de que YouTube cubría todas nuestras necesidades siendo la solución más obvia para la plataforma.

Al establecer una conexión entre la aplicación y YouTube nos evitamos tener que mantener el contenido ni desplegar un soporte web de cara al usuario, quien además ya se encuentra cómodo con el famoso portal de vídeos.

Así pues en la creación de una ruta, cuando se requiera anexar un vídeo a un punto de interés, únicamente habrá que añadir su enlace en YouTube. Cuando se desee acceder al vídeo de un punto, la aplicación cederá su enlace a la actividad que gestione la reproducción por defecto.

Las ventajas que se obtienen con este sistema son obvias:

- Desacoplamiento de la reproducción de vídeo.
- Fácil acceso y mantenimiento del contenido subido por un usuario.
- Reutilización de la diversidad de contenido audiovisual subido a la red.



Figura 14: Vídeos en FRAGUEL

4.3 Texto a voz

En una aplicación para dispositivos móviles, la cantidad de información que podemos mostrar está limitada al pequeño tamaño de la pantalla. Para evitar saturar la pantalla con elementos, explotamos toda las posibilidades de los móviles de hoy en día, y en concreto la capacidad de nuestra aplicación de narrar al usuario bien por altavoz o auriculares la descripción de cada imagen, punto de interés o ruta. Para ello ha sido fundamental la integración con la librería TTS.



Text-To-Speech (TTS) es una librería que permite a los desarrolladores añadir voz a sus aplicaciones. Gracias a TTS cualquier texto podrá ser leído en voz alta. Eyes-Free hace uso de esta librería para poder dotar de voz a las aplicaciones y permitir su uso sin necesidad de mirar la pantalla.



Catedral de la Almudena - Madrid

La catedral de Santa María la Real de la Almudena es la sede episcopal de la Archidiócesis de Madrid, (España). Se trata de un templo de 102 metros de longitud y 73 de altura, construido durante los siglos XIX y XX en una mezcla de diferentes estilos: neoclásico en el exterior, neogótico en el interior y neorrománico en la cripta.

Fue consagrada por el pontífice Juan Pablo II en su cuarto viaje a España, el 15 de junio de 1993, siendo de este modo la única catedral española dedicada por un papa.

Está ubicada en el centro de la ciudad. La fachada principal se encuentra frente al Palacio Real. La fachada del crucero mira hacia la calle de Bailén, y el acceso a la cripta se

Figura 15: Text To Speech

Así logramos que un usuario final de la aplicación que se encuentre siguiendo una ruta, no tenga la necesidad de ir mirando la pantalla en todo momento, pudiendo observar el mundo que le rodea y recibir escuchar

Por supuesto, el uso de esta funcionalidad se puede activar o desactivar a gusto del usuario con un simple clic en la opción correspondiente.

4.4 Realidad aumentada

La aplicación cuenta con un sistema de realidad aumentada básico pero funcional. Es posible posicionar elementos 3D en cualquier punto terrestre. Las vistas de la cámara y OpenGL[Ref. 2] se componen en uno de los estados implementados, con lo que es posible acceder en cualquier momento.

El sistema hace uso de la cámara principal incorporada en el terminal (actualmente todos los dispositivos incorporan al menos una cámara, siendo la principal la localizada en la parte trasera). También utiliza la brújula y el giroscopio para determinar la rotación en los tres ejes para aplicar dicha



transformación a la cámara de OpenGL. Por último hace uso del localizador GPS para establecer la posición del observador.

Por otra parte se encuentra el mecanismo gráfico de OpenGL. Para ahorrar carga de trabajo, y no tener que implementar una plataforma de manejo de la librería gráfica y carga de mallas, hemos hecho uso de una librería de código abierto llamada Min3D [Ref. 9]. Esta librería nos ha permitido las siguientes facilidades:

- Gestión de objetos virtuales: Posibilidad de transformar los objetos 3D, moverlos, rotarlos, etc.
- Carga de mallas: Permite cargar mallas establecidas como recursos de la aplicación, hay que tener cuidado con que no sean demasiado pesadas para el terminal. También es posible aplicar textura en dichas mallas.
- Creación de primitivas: Con una sola instrucción es posible crear primitivas sencillas (cubos, pirámides, esferas, etc.). No es necesario entrar en complicaciones con los mecanismos de OpenGL ES [Ref. 10].
- Movimiento de la cámara: la implementación inicial requería especificar una posición para la cámara así como la del punto hacia donde mira. Esto suponía un gran problema en nuestro caso ya que no es posible determinar un punto objetivo para la cámara, por tanto lo adaptamos para que utilizara en su lugar la orientación y rotación del terminal.

Como se ha mencionado, el sistema es capaz de crear formas básicas, como cubos, o cargar mallas 3D. En el primer caso se crean directamente a través del código ya que la librería Min3D dispone de métodos para ello.

En caso de querer cargar una malla creada desde un programa de edición 3D (ya sea sencillo como Google Sketchup [Ref. 10] o más profesional como 3D Studio Max [Ref. 11]) esta debe cumplir unos requisitos para poder ser utilizada en nuestra aplicación.

- El formato de exportación debe ser OBJ. El parser incluido por defecto es para este tipo de fichero, sin embargo es posible extenderlo a otros



formatos, como 3DS o MD2, creando nuevos parsers que implementen una interfaz codificada en el sistema de carga de recursos.

- La malla debe contener el menor número posible de polígonos, idealmente menos de dos mil. Esto hace que ocupen menos en memoria y se puedan mostrar más elementos al mismo tiempo.
- Todas las texturas que se quieran aplicar deben estar integradas en un solo fichero, esta limitación es debida a OpenGL ES por motivos de optimización.
- Si se desean mostrar varios elementos simultáneamente se puede hacer exportando una sola malla que los contenga (con el inconveniente de las limitaciones de texturas), o cargando varios OBJ en un mismo punto. En este último caso debe exportarse cada elemento desplazado según su posición en el conjunto ya que no se especifica un posicionamiento 3D para cada objeto dentro del punto de interés.
- Las normales de todas las caras deben estar correctamente alineadas. Esto evita posibles errores durante la ejecución.

4.5. Modelado del espacio

El posicionamiento, tanto de los elementos como del usuario, es un elemento clave para la aplicación. Para ello hacemos uso de todos los sensores que nos proporcionan los terminales.

El sistema usa las coordenadas GPS y la altitud para posicionar todos los elementos.

Para la representación 3D de la realidad aumentada no basta con usar directamente la información obtenida de los sensores. Esto se debe a los siguientes factores:

- Coordenadas GPS: estas coordenadas son polares (grados en una circunferencia) y no cartesianas como usa OpenGL. Por tanto es necesario realizar una conversión de los valores a una proyección plana, el sistema de partida es WGS84 (grados) y el de destino UTM (metros). Este cálculo se basa en operaciones de geometría, pero no por ello es trivial.

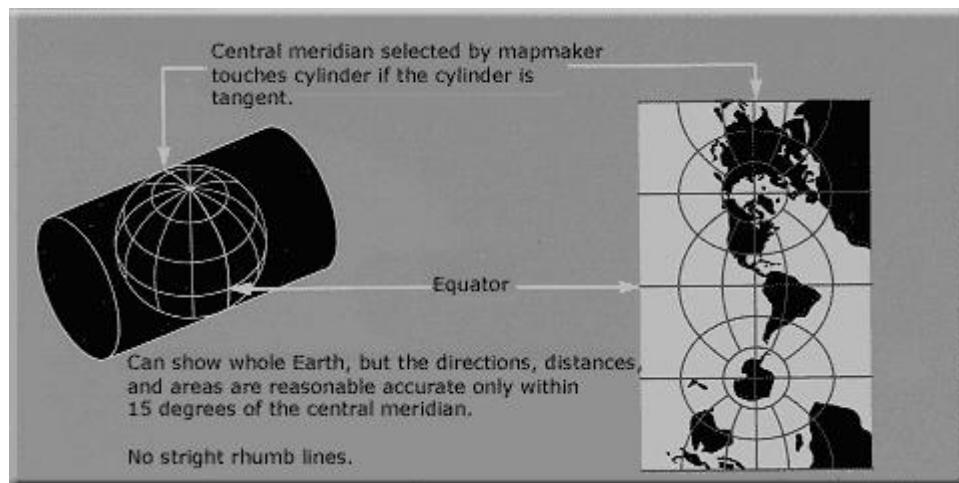


Figura 16: Modelado del espacio

El cálculo se hace en dos partes. Primero se calculan todas las variables necesarias para la conversión y por último se aplican en las fórmulas pertinentes.

Paso 1:

```
protected void setVariables(double latitude, double longitude)
{
    latitude = degreeToRadian(latitude);
    rho = equatorialRadius * (1 - e * e) / Math.pow(1 - Math.pow(e * Math.sin(latitude), 2), 3 / 2.0);

    nu = equatorialRadius / Math.pow(1 - Math.pow(e * Math.sin(latitude), 2), 1 / 2.0);

    double var1;
    if (longitude < 0.0)
    {
        var1 = ((int) ((180 + longitude) / 6.0)) + 1;
    }
    else
    {
        var1 = ((int) (longitude / 6)) + 31;
    }
    double var2 = (6 * var1) - 183;
    double var3 = longitude - var2;
    p = var3 * 3600 / 10000;

    S = A0 * latitude - B0 * Math.sin(2 * latitude) + C0 * Math.sin(4 * latitude) - D0
        * Math.sin(6 * latitude) + E0 * Math.sin(8 * latitude);

    K1 = S * k0;
    K2 = nu * Math.sin(latitude) * Math.cos(latitude) * Math.pow(Math.sin1, 2) * k0 * (100000000) / 2;
    K3 = ((Math.pow(Math.sin1, 4) * nu * Math.sin(latitude) * Math.pow(Math.cos(latitude), 3)) / 24)
        * (5 - Math.pow(Math.tan(latitude), 2) + 9 * e1sq * Math.pow(Math.cos(latitude), 2) + 4
        * Math.pow(e1sq, 2) * Math.pow(Math.cos(latitude), 4)) * k0 * (1000000000000000L);

    K4 = nu * Math.cos(latitude) * Math.sin1 * k0 * 10000;

    K5 = Math.pow(Math.sin1 * Math.cos(latitude), 3) * (nu / 6)
        * (1 - Math.pow(Math.tan(latitude), 2) + e1sq * Math.pow(Math.cos(latitude), 2)) * k0
        * 1000000000000L;
```



```
A6 = (POW(p * sin1, 6) * nu * SIN(latitude) * POW(COS(latitude), 5) / 720)
    * (61 - 58 * POW(TAN(latitude), 2) + POW(TAN(latitude), 4) + 270
    * e1sq * POW(COS(latitude), 2) - 330 * e1sq
    * POW(SIN(latitude), 2)) * k0 * (1E+24);
```

```
}
```

Paso 2:

```
protected double getNorthing(double latitude)
{
    double northing = K1 + K2 * p * p + K3 * POW(p, 4);
    if (latitude < 0.0)
    {
        northing = 10000000 + northing;
    }
    return northing;
}

protected double getEasting()
{
    return 500000 + (K4 * p + K5 * POW(p, 3));
}
```

Esta conversión no introduce más error en el posicionamiento y por tanto este se limita al que genere el sistema GPS.

Debido a la complejidad de estos cálculos hemos hecho uso de una implementación disponible en la página web de IBM.[Ref. 1]

- Ejes del terminal: los ejes sobre los que informa el teléfono no son los mismos con los que trabaja OpenGL. Es necesario aplicar una transformación de rotación para ajustar los valores. En las siguientes imágenes se puede apreciar:

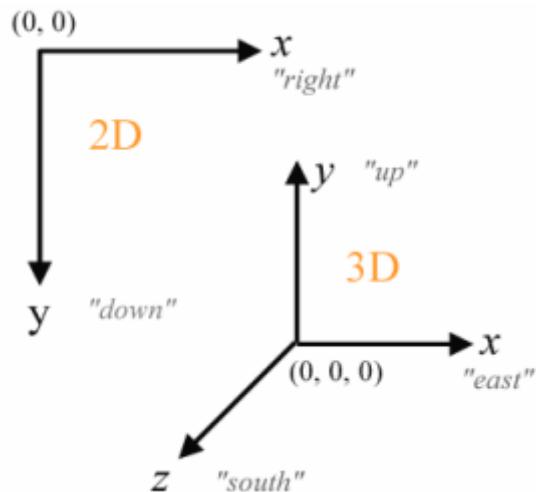


Figura 17: Figura Ejes de OpenGL

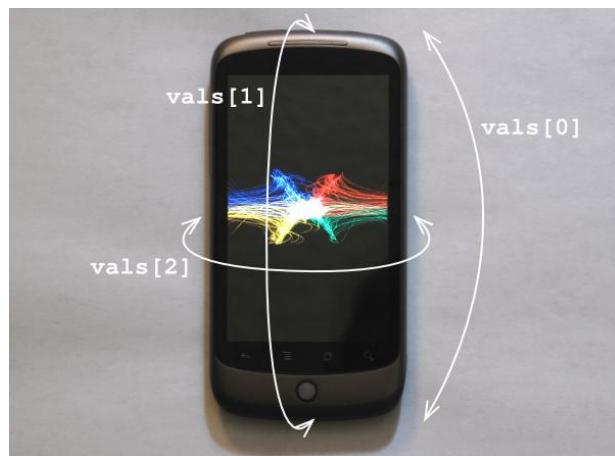


Figura 18: Ejes de Android

Hay que tener en cuenta que en Android la posición de reposo (valores (0,0,0)) es con el teléfono en horizontal, con la pantalla hacia arriba y la parte superior apuntando al norte. Con todo esto se obtiene la siguiente correspondencia:

OpenGL	Android
X	Z + 90
Y	X + 90
Z	Y

- Precisión: todos los sensores proporcionan demasiada precisión, esto provoca que se produzcan temblores en la cámara con cualquier mínimo cambio en la rotación o en la posición. Para ello es necesario establecer un delta (rango) en el cual no se apliquen los cambios.

Para la gestión del entorno 3D hemos hecho uso de una librería de código abierto, realizada para Android, llamada Min3D. Esta librería permite la carga de algunos tipos de mallas (modelos 3D) así como el posicionamiento de objetos y cámara a un nivel más alto. Esto simplifica el uso de OpenGL.



Fue necesario hacer unos retoques en dicha librería, principalmente por el tipo de funcionamiento de la cámara. El uso original requería especificar la posición de la cámara y la posición hacia dónde tiene que apuntar. En nuestro caso tenemos la posición de la cámara pero, en lugar de disponer el objetivo, tenemos información sobre la rotación de la cámara. Por tanto hubo que crear nuevos métodos accesores y mutadores para la cámara así como cambiar las operaciones de transformación que se hacían sobre la vista de OpenGL.



V

Gestión del proyecto



1. Planificación del proyecto

Debido al tiempo disponible (unos seis meses) y a la cantidad de miembros del grupo de desarrollo nos decantamos por una metodología **Scrum**. Este método se basa en un desarrollo iterativo e incremental típico del desarrollo ágil de software.

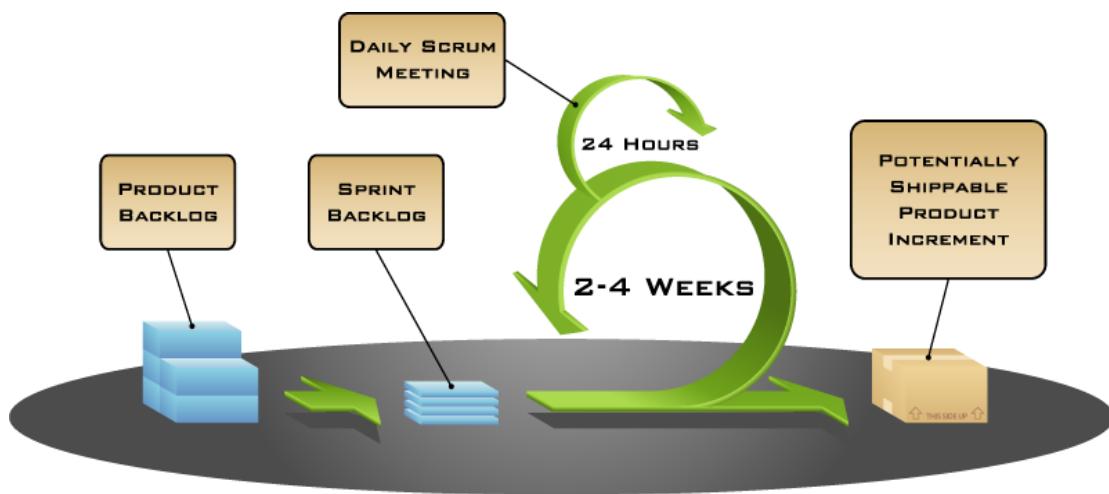


Figura 1:Modelo de planificación

Comúnmente hay cuatro roles posibles en un equipo de desarrollo Scrum:

- Scrum master: es el que se hace responsable de que se cumpla la metodología Scrum, sus prácticas y reglas. Se encarga de liderar y hacer que el grupo sea más productivo. A resumidas cuentas, es un asesor.
- Dueño del producto: su trabajo es gestionar las tareas y asegurarse del trabajo del equipo de desarrollo. Debe saber que tareas tienen más prioridad y hacer saber a cada uno su trabajo. Todos deben respetar sus decisiones.
- Equipo: es el conjunto de desarrolladores que llevan a cabo las tareas de cada iteración. Deben tener conocimientos suficientes para realizar el trabajo que se les asigna. Cabe destacar que se auto-organizan según sus habilidades y necesidades. El equipo ideal debería contar con siete personas. Más gente aumenta la complejidad y menos genera menor productividad.



Cabe destacar que, debido a la peculiaridad del proyecto, todos los integrantes del grupo hemos asumido varios papeles para poder realizar una autogestión.

El ciclo de desarrollo Scrum consiste en realizar varios hitos que proporcionen un producto utilizable, en nuestro caso nos decantamos por realizar cuatro hitos:

- 19 de Diciembre de 2010
- 20 de Febrero de 2011
- 30 de Abril de 2011
- 31 de Mayo de 2011

Para alcanzar cada hito se divide el tiempo en una serie de iteraciones programadas y de duración fija, en nuestro caso cada iteración tenía una duración de una semana. Al final de cada iteración se hace una reunión para observar el resultado obtenido y planificar la siguiente.

Cada iteración está a su vez compuesta de tareas. Cuando surge una nueva tarea, se añade al congelador (o icebox) para que se inicie en una iteración futura. Las tareas llevan asociada una complejidad estimada para ayudar a la planificación. Al cumplir un hito se decide que tareas se van a realizar para el siguiente y se pasan del congelador a la cola global (backlog). Por último, en cada reunión de iteración se hace una selección de tareas de la cola global y se ponen en la cola de la iteración. Las tareas pueden ser de varios tipos:

- Funcionalidad: es el tipo principal, añaden nuevas características a la aplicación y son la principal fuente de complejidad de las iteraciones.
- Bug: consiste en reparar algún error de la aplicación. Aportan complejidad a la iteración.
- Investigación: cómo su nombre indica, suponen realizar búsquedas y documentarse con el objetivo de poder realizar futuras tareas o mejorar el rendimiento. No aportan complejidad.
- Lanzamiento: indican tareas asociadas directamente con la versión que se publique. No aportan complejidad.

Gestionar la planificación Scrum supone un trabajo complicado si no se dispone de herramientas adecuadas. Nosotros nos apoyamos en un servicio web llamado **Pivotal Tracker**. Actualmente es gratuito para proyectos no



comerciales con un máximo de un proyecto por persona. Este sitio nos permite crear tareas, planificar y ver las estadísticas desde cualquier navegador web y con mucha comodidad. Incluso hay aplicaciones para terminales móviles. Esta captura de un proyecto de ejemplo ilustra perfectamente el sistema:

The screenshot shows the Pivotal Tracker interface for the 'My Demo Project'. The top navigation bar includes 'Projects' (selected), 'PIVOTAL TRACKER', 'Welcome, Alejandro Pérez García', and links for 'Dashboard', 'Reports', 'My Profile', 'My Accounts', 'Help', and 'Sign Out'. Below the navigation is a search bar and a 'Velocity: 8' indicator ('3 iteration average'). The main area is divided into three columns: 'Current', 'Backlog', and 'Icebox'.
Current Column: Shows tasks for the current iteration (28 Jun - Current). Tasks include:

- shopping Shopper should be able to click on a product, and see all product details, including photos
- cart, shopping Shopper should be able to add product to shopping cart
- cart, shopping Shopper should be able to view contents of shopping cart
- cart, shopping Shopper should be able to remove product from shopping cart
- cart, shopping Cart manipulation should be AJAXy
- shopping Some product photos not scaled properly when browsing products
- shopping Shopper should be able to recommend a product to a friend
- search configure solr for full text searching

Buttons for 'Accept' and 'Reject' are visible for some tasks.
Backlog Column: Shows tasks for the next iteration (5 Jul). Tasks include:

- search, shopping Shopper should be able to search for product
- Initial demo to investors
- checkout, shopping Shopper should be able to enter credit card information and shipping address
- checkout, shopping Integrate with payment gateway
- checkout, needs discussion, shopping When shopper submits order, authorize total product amount from payment gateway
- checkout, shopping If system fails to authorize payment amount, display error message to shopper
- checkout, shopping If authorization is successful, show order number and confirmation message to shopper

Icebox Column: Shows tasks that have been moved to the backlog:

- Product browsing pagination not working in IE6
- Integrate with automated order fulfillment system
- native iPhone app to allow product browsing and checkout
- Facebook app, allowing users to share favorite products

Figura 2: Pivotal Tracker

A la izquierda de la imagen se ve la iteración actual (current) con sus tareas terminadas en verde, las empezadas en naranja y las pendientes en blanco. En la columna central está la cola global de tareas y se puede apreciar un hito (en azul con una bandera a cuadros). Por último, a la derecha, vemos el congelador con las tareas que van surgiendo.



2. Gestión de riesgos

El rol o propósito esencial de la gestión de riesgos es la de mejorar el desempeño del proyecto mediante la sistemática identificación, valoración y administración de los riesgos que se puedan dar.

2.1 Identificación

Tras un análisis de los posibles riesgos de los que podía ser objeto el proyecto, los identificamos y agrupamos en riesgos relativos a la planificación del proyecto, riesgos que tienen que ver con la tecnología empleada en el desarrollo, riesgos asociados al personal que forma el equipo de trabajo y por último riesgos de fuerza mayor o externos. A continuación se muestran los riesgos identificados que tienen impacto en la realización del proyecto:

Riesgos de administración del proyecto:

- [RA-1] Incumplimiento de fechas de los hitos establecidos
- [RA-2] Alcance del proyecto demasiado ambicioso

Riesgos Tecnológicos:

- [RT-1] Problemas de compatibilidad entre diferentes dispositivos Android y sus versiones.
- [RT-2] Problemas relacionados con el comportamiento de la realidad aumentada.
- [RT-3] Funcionalidad no implementada en la API de OpenGL para Android.
- [RT-4] Problema asociados con la conexión de datos 3G del terminal.
- [RT-5] Problemas asociados con el GPS del terminal.
- [RT-6] Problemas de rendimiento del terminal.

Riesgos de personal:

- [RP-1] Ausencia conocimiento del SDK de Android.
- [RP-2] Los miembros del equipo no se implican en el proyecto y no se alcanza el nivel de rendimiento deseado.
- [RP-3] Baja permanente de un miembro del equipo de trabajo del proyecto.

**Riesgos Externos:**

- [RE-1] Baja de personal del proyecto por enfermedad o accidente.
- [RE-2] Cambio de prioridad de los tutores.
- [RE-3] Fallos en las herramientas de trabajo.

2.2 Análisis y planificación

Una vez identificados los posibles riesgos que pudieran surgir a lo largo de las diferentes etapas del proyecto pasamos a analizar y medir sus consecuencias y estudiar la forma de evitarlos o mitigarlos en su defecto.

Así pues, aunque los riesgos relativos a la administración se pueden evitar haciendo una evaluación y reajuste del proyecto cada cierto periodo de tiempo, y los relativos a los integrantes del grupo y a las tecnologías empleadas en el proyecto intentar minimizarlos, para otro tipo de riesgos como los externos se sugerirán diferentes planes de actuación para mitigarlos.

[RA-1] Incumplimiento de fechas de hitos establecidos	
Impacto	Crítico
Probabilidad	Media
Plan de actuación	Se deberá revisar la planificación del proyecto y actualizar en consecuencia el calendario de hitos seguido hasta el momento.

[RA-2] Alcance del proyecto demasiado ambicioso	
Impacto	Crítico
Probabilidad	Media
Plan de actuación	En caso de ocurrir deberá establecerse una reunión con urgencia dónde exponer los motivos y las causas y tras un análisis acotar el alcance del proyecto a uno más realista en función del trabajo desempeñado hasta el momento.

**[RT-1] Problemas de compatibilidad entre diferentes dispositivos Android y sus versiones**

Impacto	Crítico
Probabilidad	Baja
Plan de actuación	Se estudiará la migración del proyecto a una versión de Android más estable y en su caso se intentará obtener otros modelos de terminales para las pruebas.

[RT-2] Problemas relacionados con el comportamiento de la realidad aumentada

Impacto	Crítico
Probabilidad	Alta
Plan de actuación	Se analizarán las causas del problema y en función del hito en las que nos encontrremos, se dejará fuera del alcance del proyecto.

[RT-3] Funcionalidad no implementada en la API de OpenGL para Android

Impacto	Crítico
Probabilidad	Media
Plan de actuación	Este riesgo está estrechamente relacionado con el [RT-2]. En caso de suceder se estudiará el coste en tiempo del desarrollo de la funcionalidad y se estimarán revisarán las fechas de los hitos si el proyecto no se encontrara en su fase final. De no ser así, se dejará la funcionalidad de Realidad Aumentada fuera del alcance.



[RT-4] Problema asociados con la conexión de datos 3G del terminal	
Impacto	Crítico
Probabilidad	Media
Plan de actuación	Se probará a cambiar de terminal y si fuera necesario disminuir el peso de las descargas requeridas por la aplicación.

RT-5] Problemas asociados con el GPS del terminal	
Impacto	Crítico
Probabilidad	Media
Plan de actuación	De ocurrir se estudiará limitar su uso únicamente a la geolocalización.

[RT-6] Problemas de rendimiento del terminal	
Impacto	Crítico
Probabilidad	Media
Plan de actuación	Se realizarán los cambios pertinentes en la aplicación para reducir el consumo. De no ser posible se valorará la adquisición de un terminal más potente.

[RP-1] Ausencia conocimiento del SDK de Android	
Impacto	Marginal
Probabilidad	Baja
Plan de actuación	Aquellos miembros del equipo de desarrollo que tengan poca experiencia usando el SDK, deberán dedicar el tiempo suficiente para adquirir los conocimientos del resto de miembros.

**[RP-2] Los miembros del equipo no se implican en el proyecto y no se alcanza el nivel de rendimiento deseado**

Impacto	Crítico
Probabilidad	Media
Plan de actuación	Se establecerá una reunión para persuadir y motivar a aquellos miembros no implicados. De no lograrse se deberá reajustar el alcance del proyecto.

[RP-3] Baja permanente de un miembro del equipo de trabajo del proyecto.

Impacto	Crítico
Probabilidad	Baja
Plan de actuación	El resto de miembros deberán suplir esta baja y aumentar su dedicación al proyecto.

[RE-1] Baja de personal del proyecto por enfermedad o accidente

Impacto	Crítico
Probabilidad	Media
Plan de actuación	El resto de miembros deberán suplir esta baja y aumentar su dedicación al proyecto.



[RE-2] Cambio de prioridad de los tutores	
Impacto	Crítico
Probabilidad	Media
Plan de actuación	Se revisará la planificación del proyecto y se revisarán las fechas de los hitos en caso de ser necesario.

[RE-3] Fallos en las herramientas de trabajo	
Impacto	Catastrófico
Probabilidad	Baja
Plan de actuación	Si esto ocurriera será necesario reemplazar los equipos o herramientas necesarias para desarrollar lo antes posible.



3. Casos de uso

En este apartado se pasará a detallar los casos de uso del sistema a alto nivel con el objetivo mostrar toda la funcionalidad que ofrece la aplicación. Aunque usamos un diagrama de actores y los diferentes casos de uso para visualizar su relación, no hemos visto utilidad a incluir las tablas con el detalle de cada uno de ellos.

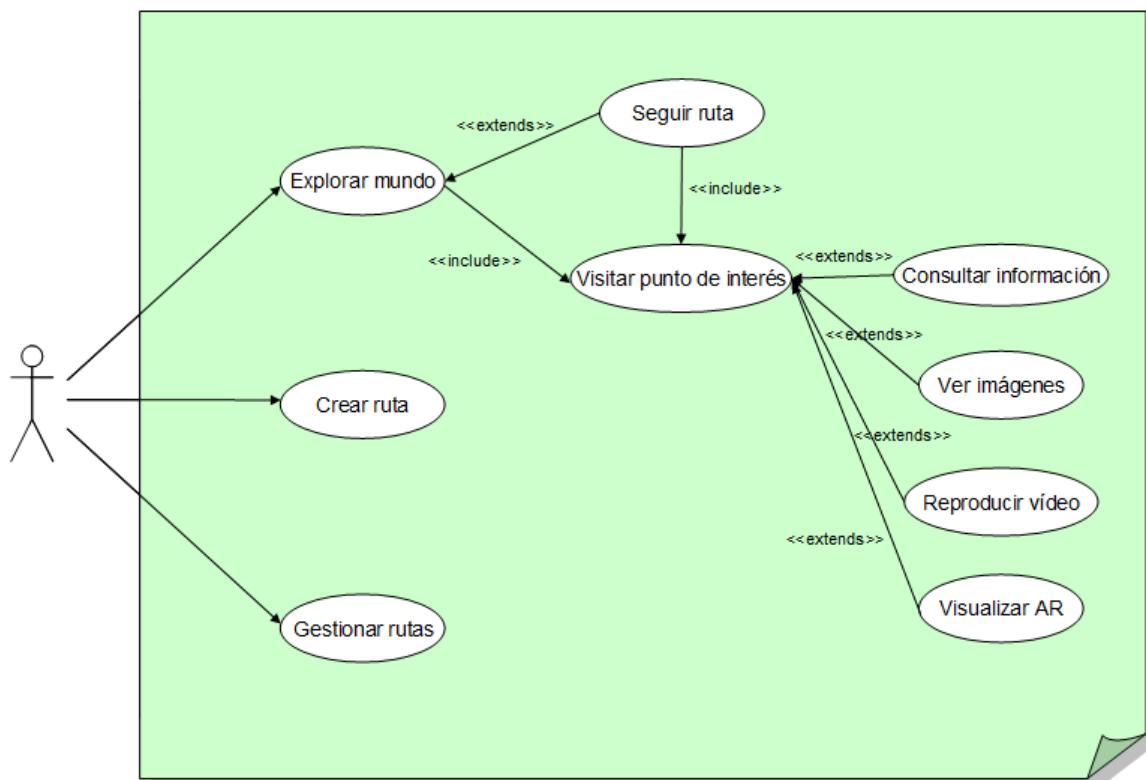


Figura 3: Diagrama de casos de uso de la aplicación FRAGUEL

A continuación se muestran los principales casos de uso que se pueden dar en la aplicación:

1. Explorar mundo

Este caso de uso se da cuando desde la pantalla de menú del sistema y pulsa en "Inicio". Cuando esto suceda se mostrará el mapa de la zona en la que esté localizado el usuario, se le situará en su posición sobre el mapa y se cargarán todos los puntos de interés de todas las rutas disponibles que se hayan descargado desde la aplicación.

Dentro de este caso de uso está incluido "Visitar punto de interés" y "Seguir ruta", de esta forma el usuario será libre de visitar cualquier punto



que localice en el mapa con solo seleccionarlo o seguir una ruta desplegándose el menú de rutas al dejar pulsado el botón sobre cualquier lugar dentro del mapa. La exploración libre del mundo finaliza precisamente seleccionar una ruta que seguir.

2. Crear ruta

En este caso de uso un usuario podrá crear e informar una nueva ruta que se podrá consumir desde la aplicación. Se accederá pulsando el botón “Nueva Ruta” desde el menú principal. Al pulsarlo se accederá al menú donde se dará la opción de crear una plantilla en blanco o ayudarse del Geotagging para situar los puntos de interés con solo pulsar sobre el mapa.

3. Gestionar rutas

Este caso de uso se dará al pulsar el botón “Gestión de rutas” en el menú principal del sistema. En este caso de uso el usuario podrá administrar las rutas disponibles en su aplicación.

Se dará la posibilidad de añadir una nueva ruta descargada al dispositivo o eliminar una ya existente. Además será posible ver de qué puntos de interés está formada cada ruta y ver su contenido multimedia.

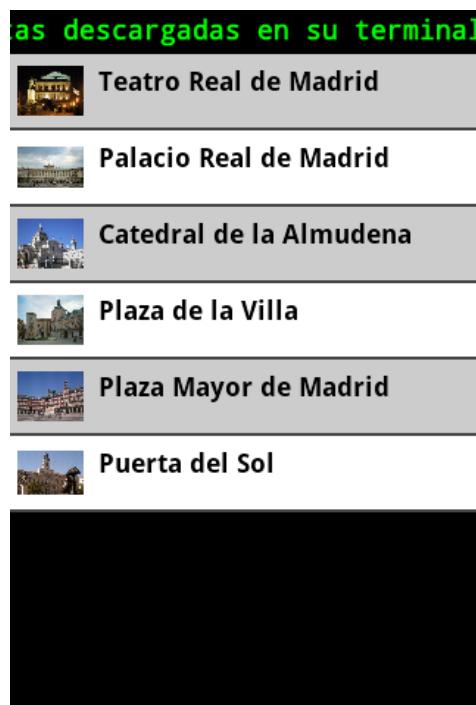


Figura 4: Gestión de rutas en FRAGUEL



4. Seguir ruta

Es una extensión del caso de uso “Explorar mundo”. Se dará cuando al explorar el mapa, tras dejar pulsado sobre cualquier punto durante un segundo, escojamos una opción del menú desplegado. En este menú se le dará al usuario las siguientes opciones:

- *Empezar desde el principio la ruta más cercana al lugar del mapa pulsado.*
- *Empezar desde el punto más próximo de la ruta más cercana al lugar del mapa pulsado.*
- *Elegir otra ruta de entre las disponibles.* Al seleccionar la opción se desplegará en el menú todas la rutas disponibles en la aplicación. Al pulsar sobre una ruta, se mostrarán los diferentes puntos de interés por los que está formada la ruta siendo posible seleccionar uno cualquiera desde donde comenzar.



Figura 5: Empezar ruta

Tras escoger una opción de las anteriores, se introducirá al usuario la ruta en cuestión, se marcará en el mapa y se indicará como llegar al primer punto de interés.



En este caso de uso está incluido “Visitar punto de interés” y finalizará en el momento en el que se seleccione la opción de para abandonar la ruta seguida.

5. Visitar punto de interés

Se entra en este caso de uso siempre que seleccionemos un punto de interés disponible sobre el mapa. Al seleccionar un punto, se mostrará un “pop up” en

la parte superior de la pantalla dónde se mostrará su descripción y todas sus posibles opciones: *consultar información, ver imágenes, reproducir vídeo y visualizar realidad aumentada*.

6. Consultar información

Este caso de uso será una extensión de “Visitar punto de interés”.

Se le mostrará al usuario toda la información, de forma escrita y hablada, relativa al punto de interés que se esté visitando.

7. Ver imágenes

Este caso de uso será una extensión de “Visitar punto de interés”. Se accederá a una galería con todas las imágenes asociadas al punto de interés que se esté visitando. Además existirá la opción de visualizar una presentación hablada mostrándonos una descripción de cada ilustración.

8. Reproducir vídeo

Este caso de uso será una extensión de “Visitar punto de interés”. El usuario tendrá la posibilidad de ver un vídeo en relación al punto de interés que se esté visitando. Podrá avanzar y retroceder por el vídeo, activar o desactivar el audio y descargarlo a su dispositivo móvil.

9. Visualizar realidad aumentada

Este caso de uso será una extensión de “Visitar punto de interés”. Tras seleccionar esta opción la pantalla del dispositivo que ejecute el sistema pasará a mostrar la vista de la cámara en posición horizontal, permitiendo al usuario observar elementos 3D superpuestos aparentemente sobre un punto del mundo real.



4. Análisis de requisitos

Esta sección trata los requisitos tomados para la realización del proyecto FRAGUEL.

Diferenciaremos entre requisitos funcionales y requisitos no funcionales. Los funcionales definen qué es lo que debe poder hacer nuestro sistema, por tanto englobarán qué es lo que un usuario debe poder hacer en nuestra aplicación final. Los requisitos no funcionales van orientados a definir bajo qué condiciones tanto de hardware como de software debe ejecutarse la aplicación.

4.1 Análisis de requisitos funcionales

- **El sistema debe ser escalable y modulable.** De forma que sea posible añadir o quitar funcionalidad sin que afecte a la arquitectura.
- **Acceder a la capa de comunicación del dispositivo.** El sistema tendrá que poder ejercer control sobre el Wifi y la conexión de datos activándola cuando se requiera.
- **Superposición de capas en OpenGL sobre una View de la cámara.** Con el objetivo de poder mostrar la Realidad Aumentada, el sistema deberá estar implementado con una API que lo soporte.
- **Acceder al sensor GPS del dispositivo.** Para localizar y situar sobre el mapa a un usuario se tendrá que tener control sobre este sensor del dispositivo.
- **Desplegar contenido multimedia streaming.** El sistema deberá ser capaz de mostrar texto, imágenes, y reproducir vídeos a partir de enlaces web haciendo uso de la capa de comunicación del dispositivo.
- **Localizar y posicionar el dispositivo en el mundo.** A partir de los datos recogidos por el GPS del dispositivo, el sistema situará al usuario sobre un mapa real y refrescará su posición en todo momento.
- **Debe ser capaz de guiar a un usuario a través de una ruta.** Haciendo uso de la capa de comunicación, el GPS y la pantalla del dispositivo, el sistema deberá ser capaz de llevar a un usuario a un punto cualquiera del mapa.



- **Añadir, eliminar y crear nuevas rutas en tiempo de ejecución.** Cualquier usuario final del sistema tendrá que poder añadir rutas creadas por otros usuarios, eliminar rutas de otros usuarios que se hubieran añadido anteriormente, y crear rutas propias que poder compartir con otros usuarios.

4.2 Análisis de requisitos no funcionales

4.2.1 Software

- **Compatible con cualquier versión de Android igual o superior a la v2.1.** Aunque en principio nos planteamos trabajar con versiones anteriores como la v1.6, lo desestimamos ya que no incorporaban algunas mejoras fundamentales que hacían mucho más estable el uso de Google Maps, captura de eventos multitouch o la superposición de capas para mostrar Realidad Aumentada.
- **GoogleMaps.** Deberá estar instalada en cualquier dispositivo que haga uso del sistema. En principio viene instalada por defecto en cualquier sistema empotrado Android, aunque de no ser así su descarga desde el Market es gratuita.
- **Aplicación YouTube móvil o similar.** El dispositivo que haga uso del sistema tendrá que disponer de una aplicación capaz de reproducir vídeos a partir de enlaces al portal YouTube. Esto es posible desde cualquier navegador web como de aplicaciones dedicadas a la reproducción y gestión de contenido del famoso portal de vídeos.



4.2.2 Hardware

- **Conexión 3G.** Para el correcto uso del sistema deberá ser imprescindible tener conexión 3G en el dispositivo y una tarifa de datos asociada. Puesto que el sistema está pensado para su uso en exteriores, el uso de Wifi no será viable por sus obvias limitaciones.
- **El dispositivo debe tener GPS.** La mayor parte de la funcionalidad del sistema está basada en la posición actual de un usuario en el mundo real, para ello será imprescindible el uso del GPS incluido en el dispositivo.
- **Correcto funcionamiento para cualquier dispositivo móvil con hardware superior o equivalente al HTC-Hero.** Puesto que el desarrollo del sistema y todas las pruebas realizadas se harán sobre el modelo HTC-Hero, no podemos garantizar el correcto funcionamiento para otros terminales que no sea éste. Sin embargo, Android OS en principio asegura la compatibilidad de sus aplicaciones para todos los terminales con una misma versión o superior.



5. Descripción de componentes

En este apartado se pasa a exponer los diferentes componentes de la aplicación a través de diagramas UML.

Para el diseño de la arquitectura de la aplicación FRAGUEL optamos por utilizar el patrón State. Puesto que vimos que era la mejor solución al problema de encontrarnos con una Activity que iba tener distintos comportamientos en función del estado en el que se encontrase. Además, una ventaja añadida de utilizar este patrón para el diseño principal del sistema, es que lo hace totalmente modular y escalable, de este modo si queremos añadir una nueva funcionalidad al sistema bastará con implementar un nuevo estado.

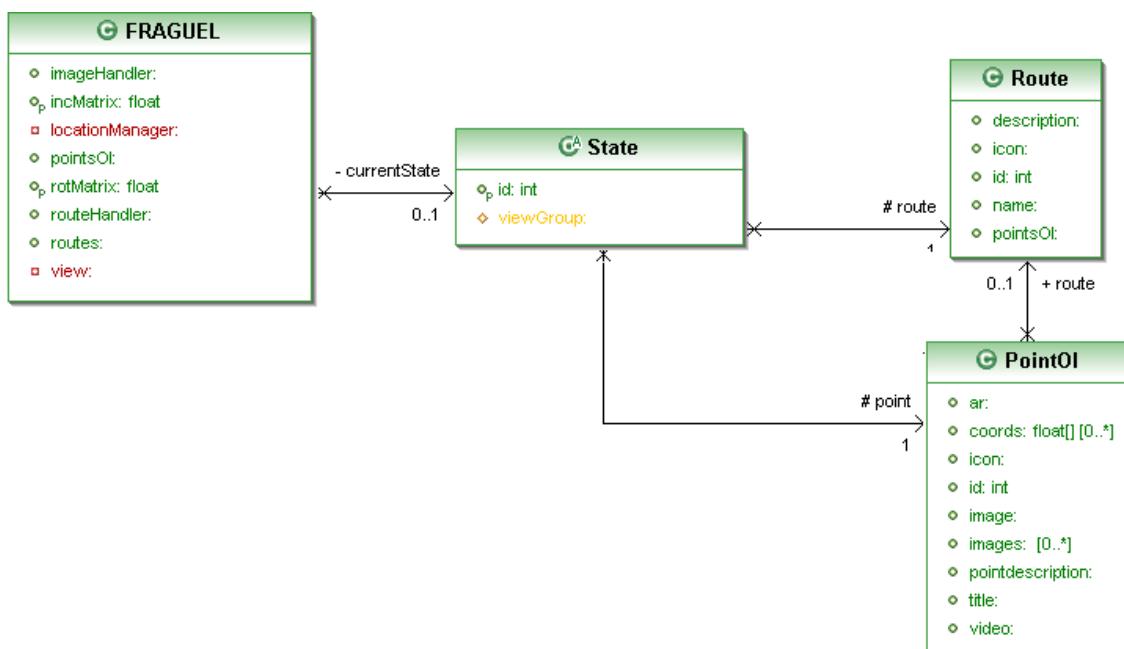


Figura 6: Diagrama UML con clase principal de FRAGUEL

• FRAGUEL

La clase FRAGUEL es la clase principal del proyecto. Es una clase Singleton que actuará como eje principal de la máquina de estados. Desde aquí se inicializarán todos objetos y estructuras necesarias para la correcta ejecución de la aplicación y será el nexo común para los diferentes estados del sistema.

• PointOl

Esta clase representará un punto de interés. Qedarán definidos todos los atributos de éste como URL's de imágenes, vídeo y malla de AR o los textos con las descripciones de cada elemento que se quiera.



- **Route**

Esta clase representará el objeto que será cada ruta que esté integrada en el sistema. Puesto que una ruta estará formada por varios puntos de interés, este objeto estará compuesto por dos o más objetos de la clase PointOI.

- **State**

Es la interfaz que implementa cada estado que tiene la aplicación. Tendrá un método load() que se invocará al cambiar de estado donde se crearán todos los objetos empleados y un método unload() donde se eliminan que se invocará al abandonarlo.

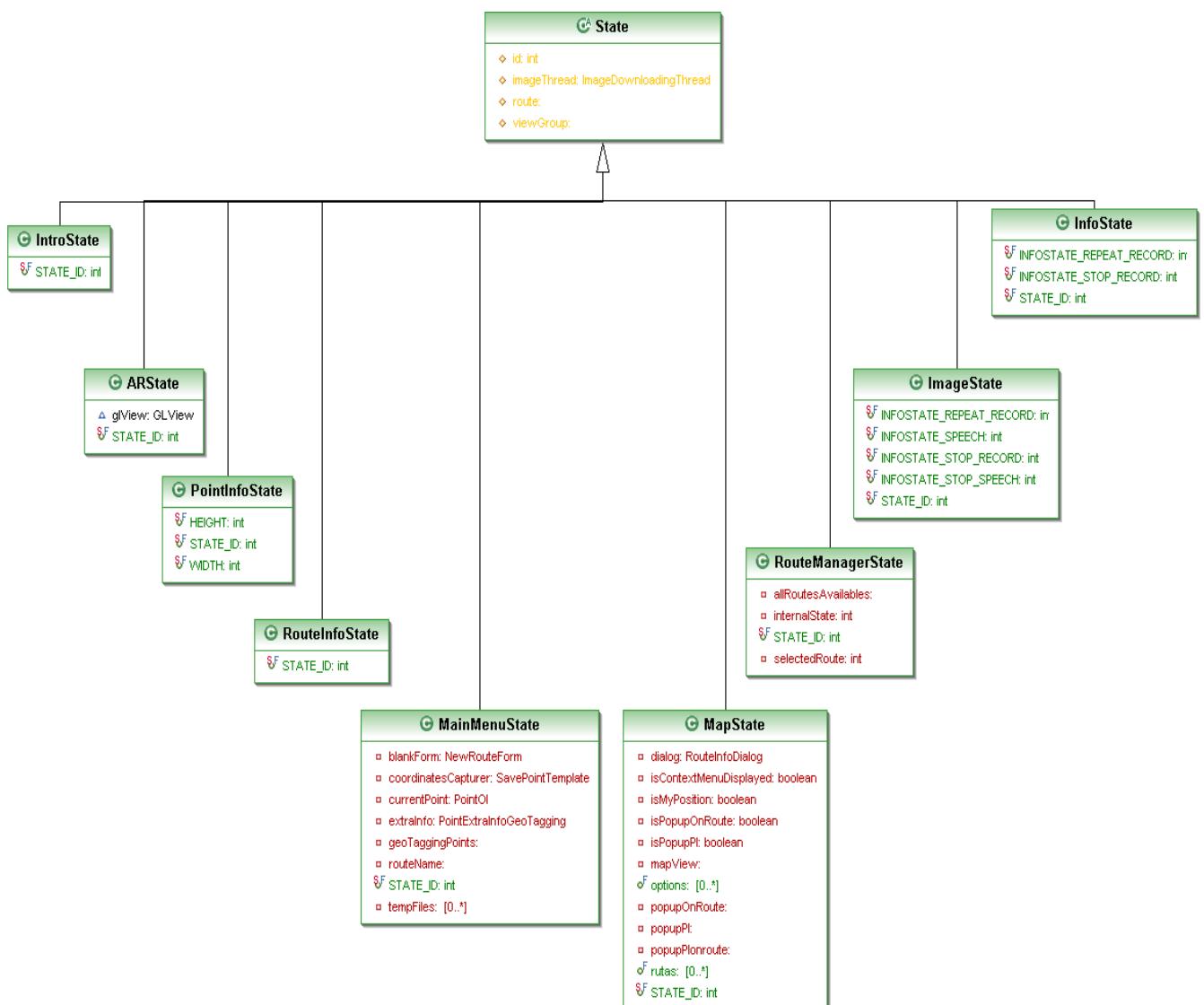


Figura 7: Diagrama UML con los estados de FRAGUEL



- **IntroState**

Esta clase usará la interfaz State y mostrará las pantallas de presentación de FRAGUEL al arrancar la aplicación.

- **InfoState**

Implementa la interfaz State y será el estado donde se muestre la información general sobre un punto de interés. La información podrá escucharse mediante el uso de la librería TTS. Además enlazará con otros estados como ImageState y ARState.

- **ConfigState**

Implementa la interfaz State. Formará un nuevo estado de la aplicación que se encargará de mostrar al usuario mediante un menú las diferentes opciones de configuración de la aplicación.

- **ImageState**

Esta clase implementa la interfaz State y será un estado más de la aplicación. Mostrará mediante una galería de imágenes las ilustraciones de un punto de interés que se esté visitando mediante enlaces URL.

- **MainMenuState**

Será un nuevo estado del sistema y por tanto implementa la clase State. Se encargará de desplegar el menú principal de la aplicación con las diferentes opciones de cara al usuario, a saber: *Explorar mundo, Gestión de Rutas, Borrar Caché y Crear ruta*.

- **ARState**

También implementará la interfaz State puesto que será un estado más de la aplicación. Se encargará de superponer una capa de OpenGL sobre la View de la cámara del dispositivo y mostrar en el lugar correcto una malla de un objeto tridimensional. Este objeto será el que tenga asociado el punto de interés que estemos mostrando. Al mismo tiempo mediante el uso de la librería TTS lanzará una descripción del objeto por voz.

- **RouteManagerState**

Puesto que forma un nuevo estado dentro de FRAGUEL, implementará la interfaz State. Se encargará de mostrar en una ListView las diferentes rutas que se encuentran descargas en la aplicación y permitiendo borrarlas mediante el método `deleteSelectedRoute(int id)` o visitarlas accediendo a la información de cada una de ellas.



- **MapState**

Esta clase es una de las más importantes del sistema, es una clase Singleton e implementa State.

Es la clase que soporta toda la interacción del usuario con el mapa. Utiliza la librería de Google Maps para cargar la tile de mapas, la TTS para el tratamiento de texto a diálogo y está estrechamente relacionada con la guía de ruta. Al iniciarse (load()) se cargarán todos los puntos de interés de todas las rutas disponibles en la aplicación y tratará todos los eventos de usuario que actúen sobre el mapa.

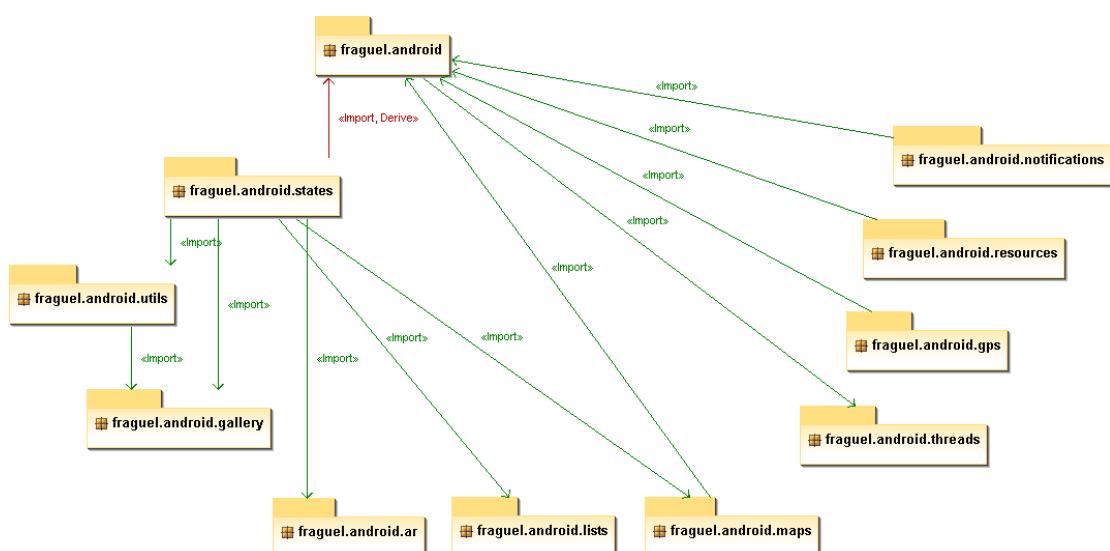


Figura 8: Diagrama UML de dependencias de paquetes de la aplicación FRAGUEL

Con el objetivo de tomar una idea general de los componentes del proyecto, pasamos a exponer a mayor nivel la agrupación, en concreto por los diferentes paquetes.

- **Paquete notifications**

Dentro de este paquete residen todas las clases asociadas a los avisos de eventos. Desde la alertas ante proximidad a un punto de interés hasta avisos del estado del GPS del terminal.

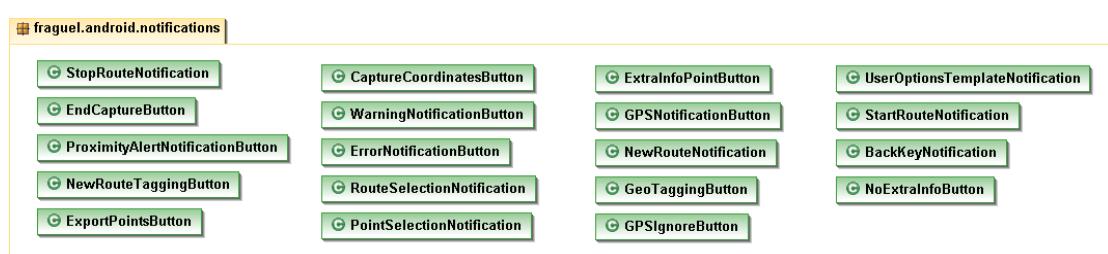


Figura 9: Esquema paquete notifications



- **Paquete resources**

Está compuesto principalmente por las clases empleadas para parsear los XML's de las rutas.



Figura 10: Esquema paquete resources

- **Paquete GPS**

Está compuesto por las clases relacionadas directamente con el dispositivo GPS del terminal. En principio hacen referencia a la alerta de proximidad a un punto de interés.

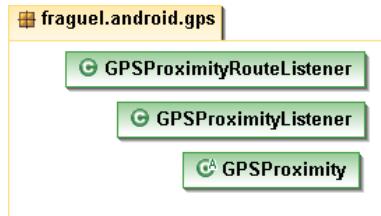


Figura 11: Esquema paquete GPS

- **Paquete threads**

Este paquete contiene las clases necesarias para la descarga simultánea mediante hilos de imágenes en la galería de un punto de interés.

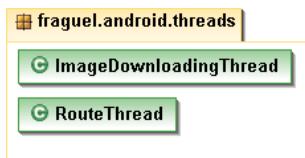


Figura 12: Esquema paquete threads

- **Paquete states**

Está formado por todos los diferentes estados de la aplicación, ya vistos en detalle anteriormente.



- **Paquete Maps**

Dentro de este paquete residirán todas las clases necesarias para el despliegue de las rutas sobre el mapa y los diferentes popup (como el de guía o el de información del punto de interés) que puedan aparecer sobre el MapView.

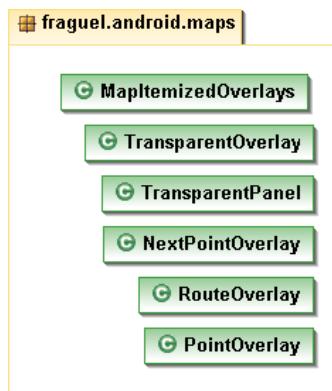


Figura 13: Esquema paquete Maps

- **Paquete utils**

Contiene una variedad de clases de diferente ámbito, como la plantilla del formulario para crear una nueva ruta o la clase para en texto informativo de los puntos de interés.



Figura 14: Esquema paquete utils

- **Paquete ar**

Será donde se encuentren todas la clases necesarias para el despliegue de la realidad aumentada en el terminal dentro del ARState.

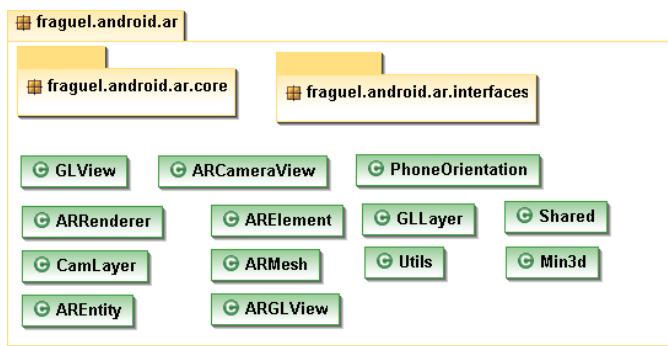


Figura 15: Esquema paquete ar

- **Paquete lists**

Contiene los `BaseAdapter` de los estados de información del punto de interés y el gestor de rutas de la aplicación.

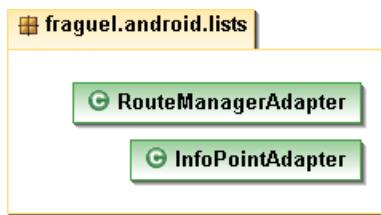


Figura 16: Esquema paquete lists

- **Paquete gallery**

En este paquete se incluyen todas clases auxiliares para mostrar una galería de objetos, en nuestro caso imágenes, aunque inicialmente estaba pensada también para una galería de vídeos pero finalmente se excluyó de los requisitos. Además contiene otras clases como `FullScreenGallery`, para la visualización a pantalla completa.

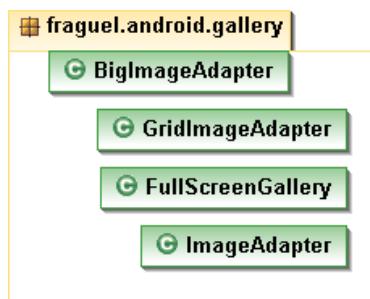


Figura 17: Esquema paquete gallery



VI

Desarrollo del proyecto



1. Etapas de desarrollo

1.1 Primer hito (19/12/2010)

En esta primera entrega nos propusimos realizar una demo tecnológica en la que mostrar las capacidades básicas del sistema. Debido al desconocimiento de la plataforma y a ciertas incompatibilidades no pudimos avanzar con el apartado de realidad aumentada con la cámara.

Pese a todo conseguimos utilizar los mapas, la localización y orientación del terminal, así como mostrar texto e imágenes y la conversión de texto a voz.

Los primeros pasos realizados fueron orientados a familiarizarse con el entorno de desarrollo Eclipse [Ref. 13], los requisitos técnicos y la manera de trabajar propias de Android. Estos pasos son los siguientes:

- Crear la aplicación base de Android [Ref. 14] que hace uso de las distintas APIs [Ref. 15] de Google [Ref. 16]. Esto implica que a partir de entonces podíamos utilizar gran cantidad de elementos ya implementados y que funcionan sin problemas, en nuestro caso sólo era necesario el servicio de mapas.
- Implementar la máquina de estados para la aplicación, siguiendo el patrón de programación State [Ref. 17]. Con esto podemos realizar distintos módulos intercomunicados a voluntad, cada uno con una funcionalidad de la aplicación. Dispone también de una pila para que sea posible retroceder en el flujo de los estados recorridos.
- Obtener la información de los sensores, aunque aún no se hacía uso de ellos. Se crearon los distintos observadores [Ref. 18] necesarios y almacenaba los resultados obtenidos.
- Primeros pasos con las vistas [Ref. 19], punto muy importante dado que es la base de toda interfaz gráfica en Android.
- También se realizaron las primeras pruebas con el sistema de “Texto a voz” [Ref. 20].



1.2 Segundo hito (20/02/2011)

Debido a la persistencia de los problemas con la combinación de cámara y OpenGL nos centramos en el resto de aspectos de la aplicación. En estos aspectos tenemos la carga de ficheros desde una URL o desde la tarjeta de memoria, mensajes de error. También creamos un sistema de ficheros XML para el almacenamiento de la información de las rutas y puntos de interés. Este hito supuso también la creación de una galería de imágenes y un reproductor de video.

Se integró toda la funcionalidad en un estado central (el Menú principal), desde el que se accedía al mapa y a la gestión de rutas.

Se implementaron todos los avisos y mensajes de error, así como los pop-ups de información que darían la posibilidad de acceder al contenido multimedia en cada punto. Al mismo tiempo se integró la posibilidad de parsear mediante SAX los ficheros XML de las rutas con los pop-ups y estados implementados, por lo que el acceso desde un punto a un contenido multimedia era dinámico.

Por otra parte se analizó toda la lógica de las rutas y su seguimiento, implementando el modo libre y realizando ya las pruebas unitarias y de integración oportunas.

En esta etapa también se investigó todo lo relacionado con la petición de rutas al servidor de Google Navigation, la recepción de los resultados y su tratamiento e integración con la aplicación.

1.3 Tercer hito (30/04/2011)

En esta entrega realizamos diversas mejoras en todos los sistemas. Principalmente en la gestión de los sensores y la estructura de los XML. Cabe destacar que después una ardua tarea de investigación y experimentación se consiguió hacer funcionar OpenGL junto con la cámara así como integrar un framework para trabajar con el sistema gráfico de una manera más sencilla, aunque hubo que realizarle ciertas modificaciones.

Se añadió la funcionalidad de añadir y eliminar rutas del terminal, así como la posibilidad de ver el contenido de cada punto sin necesidad de acceder al mapa y buscarlo.



En esta etapa se terminó de implementar la lógica del seguimiento de las rutas, desde su inicio hasta su fin, dibujando en el mapa la ruta con dos colores diferentes para que el usuario pudiese distinguir fácilmente entre los puntos visitados y los que le restan por visitar.

Se realizó por completo la funcionalidad de creación de rutas a medida por parte del usuario y la captura de los puntos. Esto produjo también añadir la funcionalidad extra de poder continuar editando una ruta propia ya empezada.

Se realizaron pruebas exhaustivas de integración y unitarias de cada módulo nuevo que se integrará para conservar el buen funcionamiento de la aplicación hasta el momento.

1.4 Cuarto hito (31/05/2011)

El objetivo de este último hito fue darle unas últimas pinceladas al sistema antes de su entrega. Se realizó una limpieza general del código así como mejoras en la interfaz. En el tema de la realidad aumentada se permite la carga de mallas 3D para posicionarlas en el mundo y la aplicación de texturas a dichos objetos.

Esta parte de realidad aumentada, absorbió gran parte del esfuerzo de esta etapa ya que existía gran margen de error en la geolocalización y posicionamiento de la cámara en la parte de realidad aumentada. Éste error surgía del propio error que devuelve el receptor GPS así como de la conversión de dichas coordenadas recibidas a metros para su integración con OpenGL.

También se empleó mucho esfuerzo en la memoria del proyecto así como en la documentación del código para su posible reutilización.

Finalmente se realizaron numerosas pruebas de verificación y de validación por parte de los directores del proyecto así como de compañeros de la misma facultad, obteniendo buenos resultados, estudiando la viabilidad de las sugerencias recibidas y su posible implementación y/o mejora.



2. Proceso de trabajo

2.1 Creación de tareas

Este primer proceso, esencial para un desarrollo Scrum [Ref. 8], ha surgido por dos maneras distintas:

- Durante la iteración: mientras tenía lugar el desarrollo, y antes de tener la siguiente reunión, surgen nuevas ideas, *bugs* [Ref. 21] o correcciones. Tras reflexionar sobre su composición las añadimos al *Icebox* [Ref. 5] para tratarlas en la siguiente reunión. Por lo general no se realiza una estimación de este tipo de ideas ya que las cargas de trabajo mejor predichas son aquellas meditadas por todo el grupo.
- Reuniones: En las reuniones que hemos llevado a cabo periódicamente (cada una o dos semanas) realizamos una fase de sugerencia de nuevas tareas a realizar. Estas sugerencias se basan en el trabajo que se ha llevado a cabo durante la última iteración, teniendo muy en cuenta qué se ha completado y qué no. También se le intenta asignar una estimación de la carga de trabajo que suponen.

2.2 Desarrollo de las tareas

En el periodo entre reuniones se procede a desarrollar las tareas asignadas. Esta asignación se realiza durante la reunión Scrum e, idealmente, solo involucra a un miembro del equipo; sin llegar nunca a ocupar a los tres integrantes. Esto ayuda a poder trabajar en distintos frentes al mismo tiempo.

Es muy importante realizar una correcta estimación de la carga de trabajo que suponen las tareas, antes de asignarlas se le asigna una o modifica la existente según se viera conveniente. Una vez realizada la tarea también se modifica si ha supuesto más o menos carga de la prevista. Esto hace que para futuras iteraciones se tenga una visión más ajustada a la realidad de lo que es capaz de realizar el equipo.

Cabe destacar que, pese a esta separación de trabajos, hay una constante comunicación entre los miembros para poder solventar las dependencias existentes, así como contribuir con ayuda o sugerencias para el trabajo de los demás.



2.3 Revisión de fallos

Este proceso tiene lugar como una etapa de la reunión Scrum. Se realizan unas pruebas para comprobar el correcto funcionamiento de las tareas realizadas en la última iteración. Si se observa algún fallo se intenta identificar el origen y solucionarlo durante la propia reunión. Pero si por cualquier motivo no se encuentra la razón del fallo o no es posible resolverlo en el momento, se crea una nueva tarea de tipo *bug* para que se lleve a cabo en una próxima iteración y con la mayor brevedad posible.

Una vez se ha realizado la tarea correspondiente a ese fallo ya se da por resuelto, pasando a un archivo de fallos resueltos que contenga una descripción del mismo, origen y solución. De esta forma se lo sigue teniendo en cuenta por si en un momento posterior pudo generar otro fallo y así encontrar con más facilidad su origen.



VII

Resultados



1. Discusión de los resultados obtenidos

Evaluando el conjunto de resultados obtenidos durante la realización de este proyecto hemos llegado a la conclusión de que ha sido muy satisfactorio.

1.1 Revisión de los objetivos iniciales

A continuación se revisan todos los objetivos planteados en el primer capítulo de este documento.

- **Implementación del sistema en dispositivos móviles**

Aunque ningún miembro del equipo poseía experiencia previa en el desarrollo de aplicaciones para sistemas empotrados, ayudándonos de la experiencia de otros usuarios en otros proyectos similares hemos logrado superar los problemas y particularidades de este tipo de plataformas. En especial las pruebas del software con los terminales y el control de sus sensores.

- **Posicionamiento en exteriores**

Gracias al control del dispositivo GPS y la conectividad 3G o por Wifi, hemos podido solventar con éxito este punto. A pesar de que para la representación la posición en mapas nos hemos ayudado de Google Maps, para las coordenadas de los puntos de interés así como la lógica del sistema de guía de rutas, hemos necesitado el receptor GPS integrado y/o en su defecto, la conexión a Internet.

- **Uso de la realidad aumentada**

Este punto se definió como uno de los objetivos principales a cumplir y sin duda ha sido el que más coste en tiempo y esfuerzo ha llevado. Bien por falta de documentación al respecto, por tratarse aún de una tecnología en auge y novedosa y por la contrastada inestabilidad de Android a la hora de superponer capas mediante el uso de OpenGL, hasta la última etapa del proyecto no logramos cumplirlo, llegando incluso a pensar en dejarlo definitivamente fuera del alcance. Sin embargo finalmente logramos solucionar los problemas de la representación de mallas de objetos tridimensionales sobre la capa de la cámara de los terminales así como situarlo correctamente en un plano tridimensional del mundo real.



- **Participación activa del usuario**

Para llevar a cabo este punto lo más fielmente posible, hemos ido mostrando la aplicación a diferentes usuarios y teniendo en cuenta su feedback tanto en la funcionalidad como en la navegación.

Como resultado, la aplicación mantiene siempre informado al usuario final de sus opciones y en lo que respecta al sistema de guía de rutas empleando diversas formas, desde mensajes y vibración hasta texto hablado.

Por último, un aspecto que creíamos también importante es dejar al usuario participar activamente en el contenido de la aplicación, pudiendo crear sus propias rutas de una forma ágil y haciendo uso de geotagging. Por otro lado, no hemos podido llegar hasta el punto de crear un portal para compartir el contenido creado, quedando dentro del trabajo futuro de este proyecto. Si un usuario desea compartir sus propias rutas deberá copiar su archivo al otro terminal.

- **Emplear alto contenido multimedia**

Pretender hacer accesible al usuario todo tipo de contenido multimedia disponible desde un terminal ha supuesto tener que investigar por separado la forma más óptima de disponer y mostrar imágenes, vídeos, texto y texto hablado y por supuesto los objetos 3D derivados de la realidad aumentada que se encuentra disponible en cada punto de interés de cada ruta. Uno de los puntos clave ha sido cómo y dónde almacenar todo este contenido. Finalmente la mejor solución ha sido aprovecharnos de aplicaciones externas para el contenido más pesado, los vídeos, o directamente de la web para imágenes. De forma que la única información que guardamos y mantenemos son los propios ficheros XML de cada ruta.

- **Desarrollar el contenido inicial**

Como muestra del potencial de nuestra aplicación se han creado dos rutas con ámbitos diferentes. Una ruta cultural sobre puntos históricos del centro de Madrid, y otra ruta de ocio que muestra una ruta de un usuario por diferentes bares y lugares de fiesta en la zona de Moncloa.



1.2 Resultados positivos

Al comienzo del curso teníamos grandes dudas sobre la orientación que iba a tomar la aplicación. Se barajaban opciones muy dispares, desde realizar un juego hasta una simple guía histórica. Pero, finalmente, esto se fue enderezando a medida que aprendíamos cómo sacar provecho a los smartphones y se optó por una guía que puede servir para cualquier propósito y con una componente social que permite compartir experiencias a través de la creación de nuevas rutas.

También consideramos muy satisfactorio el aprendizaje del sistema Android adquirido durante el desarrollo. Puede ser de gran utilidad dado el auge del mismo y las previsiones de que aumente aún más. Esto se refleja en la cantidad de nuevas empresas, así como creación de departamentos específicos en empresas ya existentes, cuyo objetivo es el desarrollo de todo tipo de aplicaciones para este tipo de dispositivos móviles.

Del mismo modo, se debe considerar la cantidad de técnicas utilizadas en la aplicación; siendo muy dispares. Por ejemplo la carga de XML, conexión con un servidor, geolocalización, 3D... Esto ha contribuido a que hayamos adquirido práctica en bastantes ámbitos, saliéndonos un poco de lo estándar.

1.3 Resultados negativos

El principal factor negativo ha sido el desconocimiento del sistema Android y de las restricciones y prácticas que conlleva. Esto nos ha causado grandes retrasos y nos ha obligado a pararnos a reflexionar e investigar sobre lo que teníamos ante nuestros ojos. Si bien hemos conseguido un buen resultado global, es bastante probable que de haber conocido con más detalle esta plataforma hubiéramos obtenido este mismo resultado mucho antes e incluso poder optar a añadirle alguna funcionalidad extra.

Las características de hardware, aunque suponen un gran avance, siguen sin ser muy potentes. Esto impide crear modelos o texturas muy detalladas en el campo de la realidad aumentada. Pero la tendencia indica que la potencia de procesamiento crece a pasos agigantados, porque es seguro que se podrá disponer de mayores recursos a muy corto plazo. Tampoco contribuye el tener que desarrollar en Java, esto disminuye la eficiencia del código, aunque



los responsables de Android ya están cambiando la mentalidad y poniendo a disposición un entorno de desarrollo en C++.

El posicionamiento GPS, incluso apoyado por la triangulación de antenas, sigue teniendo un margen de error considerable. Esto afecta a todos los campos pero especialmente al de la realidad aumentada. Sin embargo en unos años, si no cambia la planificación, dispondremos de nuevos sistemas de geolocalización más precisos.

2. Descripción de las alternativas a las empleadas

Para la realización de este proyecto teníamos dos principales plataformas de desarrollo: Android (Google) y iPhone (Apple). La plataforma de Apple tiene mucho renombre, pero el número de terminales con Android crece de forma desmesurada. También es importante recalcar que para el desarrollo en iPhone es necesario disponer de un ordenador con el sistema operativo Mac OS, cosa que supondría un importante desembolso de dinero del cual no disponemos. Otra pega del desarrollo en iPhone es la necesidad de aprender el lenguaje Objective C, de uso exclusivo para los sistemas de Apple.

Una vez elegida la plataforma evaluamos la opción de usar librerías de realidad aumentada. Esto no nos llevó por el camino que queríamos ya que ninguna de ellas introducía elementos 3D.

3. Análisis de la complejidad final

El resultado obtenido ha sido satisfactorio, pero altamente complejo. El primer obstáculo, y enormemente importante, ha sido la carencia de terminales para poder desarrollar. Hemos realizado las pruebas en los teléfonos que usábamos los miembros del grupo. Esto por una parte es bueno, ya que pruebas en distintos hardwares; pero por otra parte nos ha dificultado dado que no todos cumplían con los requisitos mínimos de versión del sistema operativo. Esto nos ha dado muchos quebraderos de cabeza ya que ha impedido poder realizar pruebas de lo realizado.

Otro punto de complejidad ha sido el entorno de desarrollo para Android. Ninguno de nosotros había trabajado con anterioridad en dicho entorno y, por tanto, el desconocimiento era total. Durante una gran parte del inicio del proyecto nos hemos dedicado a realizar pruebas para familiarizarnos y



averiguar qué cosas eran posibles. Nada de lo que habíamos visto con anterioridad en la universidad se parecía a lo realizado, desde el sistema de vistas para la interfaz hasta la manera de gestionar los recursos.

También podemos marcar como compleja la propia visión de la aplicación, que ha sufrido dos importantes cambios según se iba desarrollando. Empezó siendo un juego para pasar a ser una herramienta educativa, y al final cambió para ser una aplicación de ocio más social y colaborativa. Estos cambios se han debido a las distintas dificultades y limitaciones que hemos ido encontrando durante el proceso.

Concluyendo, el desarrollo ha sido bastante complejo pese a lo cual nos hemos enfrentado a ello, adaptándonos y aprendiendo nuevos conocimiento y de nuestros errores.

4. Trabajo futuro

FRAGUEL abre un gran campo de trabajo para la mejora de la aplicación. La mayoría de estos cambios tienen relación con la componente social del sistema. Un posible campo de actuación sería simplificar la creación de contenidos, haciéndolo más accesible para el usuario poco experto. Esto podría involucrar la creación de un sistema web dónde almacenar la información que se va creando desde el terminal y así poder editarlo más tarde desde cualquier navegador.

Algo que es muy factible y podría aportar mucho sería la integración de dicho sistema web con las redes sociales. De esta forma la gente podría compartir sus experiencias publicando sus contenidos (ya sean rutas o puntos de interés) en sus tablones de Facebook, Twitter, Tuenti, etc.



VIII

Apéndices





Manual de usuario

FRAGUEL





Manual de Usuario

FRACUEL

Fraguel es un Framework de Realidad Aumentada para Guías en Entornos Locales válido para teléfonos móviles que empleen como sistema operativo Android 2.1 o superior. Con esta aplicación el usuario podrá disfrutar y navegar a través de rutas de distinta idiosincrasia, sumergiéndole en un mundo multimedia y de realidad aumentada.

Índice de Contenido

Manual de Usuario

Requisitos Previos.....	I
Instalación.....	II
¿Cómo empezar?.....	VI
Inicio.....	VI
Menú principal.....	IX
Inicio mapa y uso de la aplicación.....	XI
Modo libre.....	XII
Modo ruta.....	XIV
Multimedia.....	XV
Gestión de rutas.....	XVII
Nueva ruta.....	XVIII
Plantilla en blanco.....	XVIII
Mediante Captura de puntos.....	XIX



Requisitos previos

El dispositivo móvil en el que desee instalar la aplicación FRAGUEL debe usar el sistema operativo Android 2.1 o superior.

Los requisitos de hardware mínimos necesarios para la instalación y el correcto funcionamiento de la aplicación son:

- Procesador con frecuencia de reloj mínima de 500MHz.
- 256 Mbytes de memoria RAM.
- 5MB de memoria interna para poder instalar la aplicación.
- Receptor GPS.
- Conexiones de datos 3G o Wifi.
- Acelerómetro de 3 ejes.
- Sensor magnético.
- Vibrador.
- Tarjeta de memoria externa con al menos 10 MB de espacio libre.

Para el correcto funcionamiento de la aplicación deberá habilitar el uso de satélites GPS así como tener una conexión permanente a Internet, ya sea mediante conexión de datos o redes Wifi. En la sección *Inicio* encontrará cómo activar dichas funcionalidades en su celular.

Se recomienda que el dispositivo en el que se desee instalar la aplicación, tenga hardware gráfico dedicado para mejorar el rendimiento de la de realidad aumentada.



Instalación

- **Desde Android Market**

- Busque la aplicación 'Android Market' en su teléfono móvil y pulse sobre ella para ejecutarla.



Figura 1: Icono 'Android Market'.

- Al iniciar 'Android Market' se mostrarán las diferentes opciones de búsqueda de aplicaciones por temática así como las aplicaciones más recientes y populares. Para buscar 'FRAGUEL' pulse en el ícono de la Figura 2. En el campo de texto que aparece escriba 'fraguel' y pulse el ícono mostrado en la Figura 3 para iniciar la búsqueda.

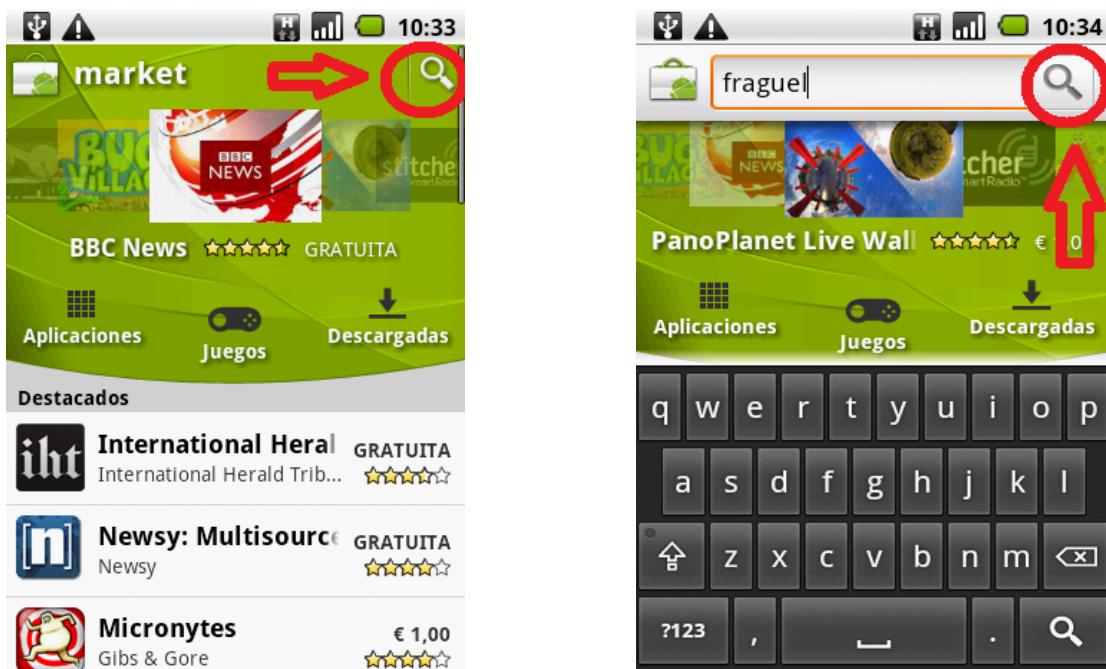


Figura 2: Página principal de Android Market. Figura 3: Inicio de la búsqueda de aplicaciones.

- A continuación se mostrará el resultado de la búsqueda (Figura 4). Pulse en la aplicación 'FRAGUEL' para comenzar su instalación.

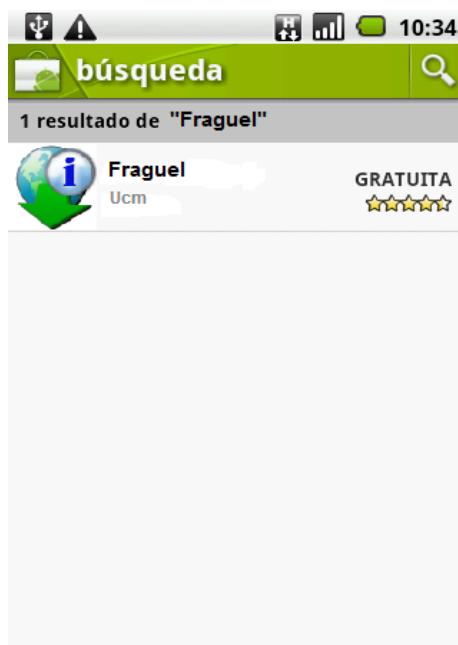


Figura 4: Resultado de la búsqueda.



- Pulse en el botón 'Gratis' como se muestra en la Figura 5 y aparecerán los permisos que utiliza la aplicación (Figura 6). Pulse aceptar y comenzará a descargarse la aplicación 'FRAGUEL'. Una vez descargada podrá ejecutarla desde su terminal.



Figura 4: Instalación 'FRAGUEL'.



Figura 5: Aceptar permisos.

- Desde la Web de FRAGUEL**

- Descargue el archivo *fraguel.apk* de la siguiente dirección web <http://www.blackmesa.es/fraguel/fraguel.apk> y guarde el archivo en la tarjeta de memoria externa de su terminal.
- Utilice un navegador de archivos para navegar por los ficheros de su tarjeta de memoria externa. Si no dispone de navegadores de ficheros puede descargarse gratuitamente del 'Android Market' la aplicación 'AndroZip File Manager' o bien 'Adao File Manager'.

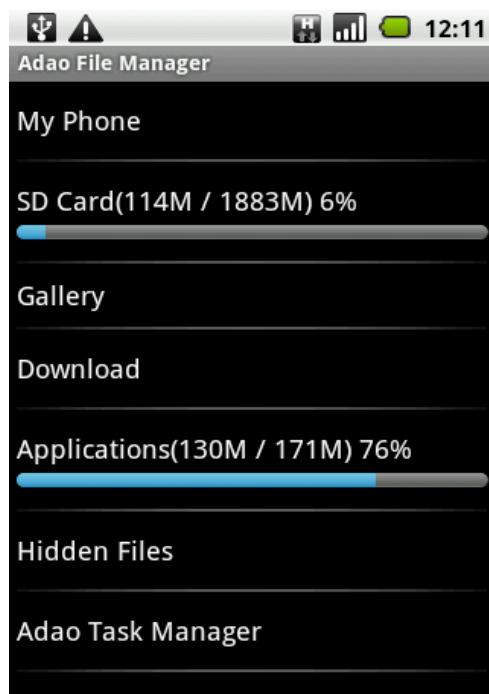


Figura 6: Navegador de archivos 'Adao File Manager'

- Una vez localizado el archivo *fraguel.apk* en su tarjeta de memoria externa pulse sobre él y la aplicación se instalará automáticamente.

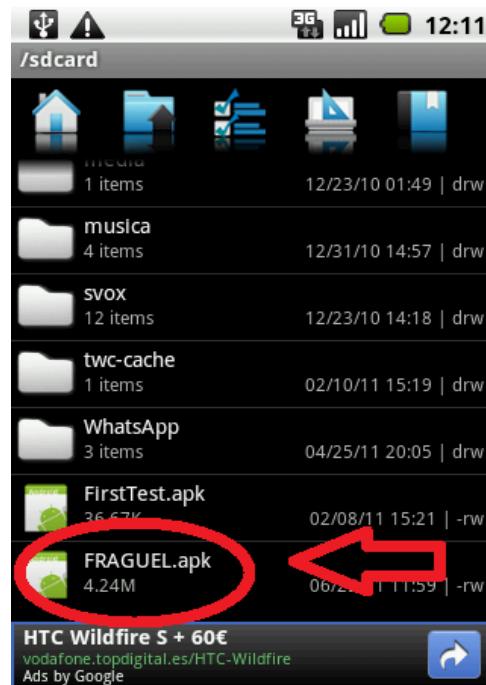


Figura 7: Instalación de 'FRAGUEL' desde la tarjeta de memoria externa



¿Cómo empezar?

Inicio

Una vez descargada la aplicación y habiéndose instalado correctamente, deberá activar el GPS de tu terminal así como la conexión de datos o Wifi en su defecto, tal y como se ha comentado en la sección de *Requisitos previos*. Para ello, diríjase al menú principal, pulse sobre el ícono *Ajustes* donde encontrará los menús *Ubicación y seguridad* para activar el receptor GPS e *Inalámbrico y redes* para activar o bien la conexión Wifi o la conexión de datos (Figura 9).

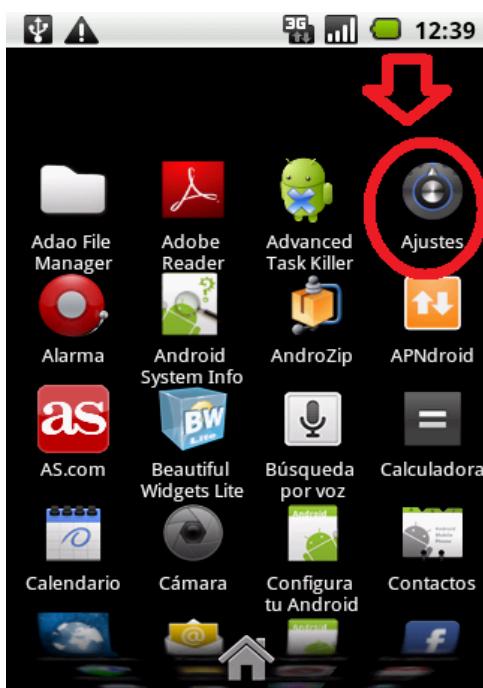


Figura 8: Menú principal.

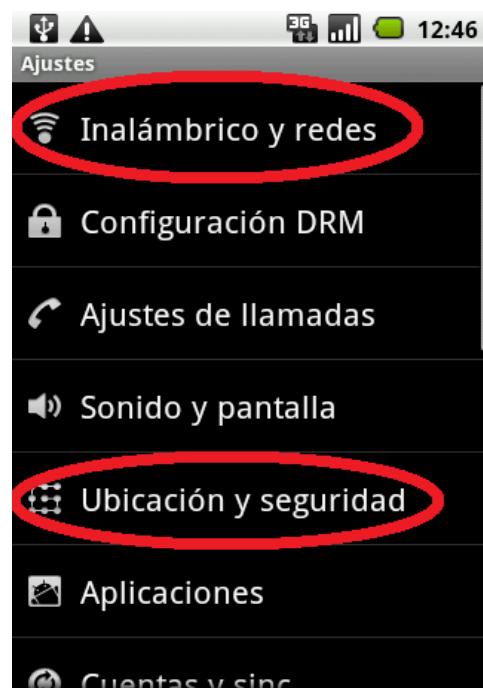


Figura 9: Menú Ajustes.

- Si quiere activar el receptor GPS seleccione *Ubicación y seguridad* en el menú Ajustes (Figura 9). A continuación seleccione *Usar satélites GPS* para activarlos (Figura 10) y observará que el ícono de esa opción torna a color verde (Figura 11).



Figura 10: Activación receptor GPS.



Figura 11: GPS activado.

- Si quiere activar la conexión de datos o Wifi seleccione *Inalámbrico y redes* en el menú de Ajustes (Figura 9). A continuación seleccione *Wifi* o *Redes móviles* según la opción que desee (Figura 12). Si escoge la opción *Redes móviles*, seleccione *Conexión Activa* para activarlas (Figura 13).



Figura 12: Activación conexión Internet activa.



Figura 13: Conexión de datos activa.

Una vez activada la conexión con Internet y el receptor GPS iniciamos la aplicación pulsando sobre la aplicación FRAGUEL en el menú principal (Figura 14).

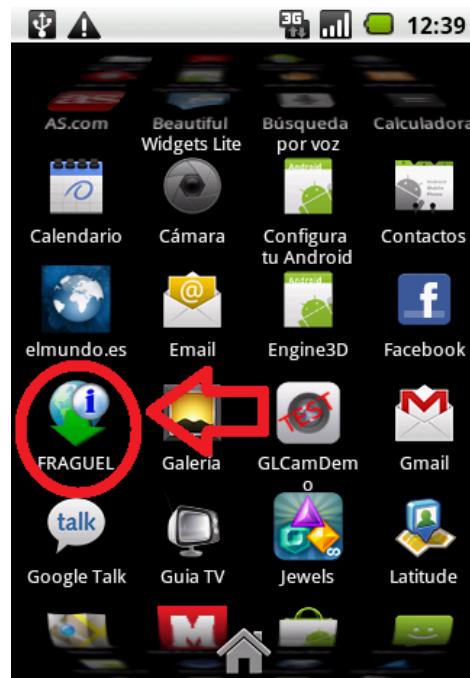


Figura 14: Menú principal.



Al comenzar la ejecución de la aplicación aparecerá una animación de introducción (Figura 15) y posteriormente se le mostrará el menú principal. Si no ha activado el receptor de GPS aún, se le mostrará un aviso pudiendo activar el GPS desde la misma aplicación (Figura 16).



Figura 15: Animación de introducción.
GPS.

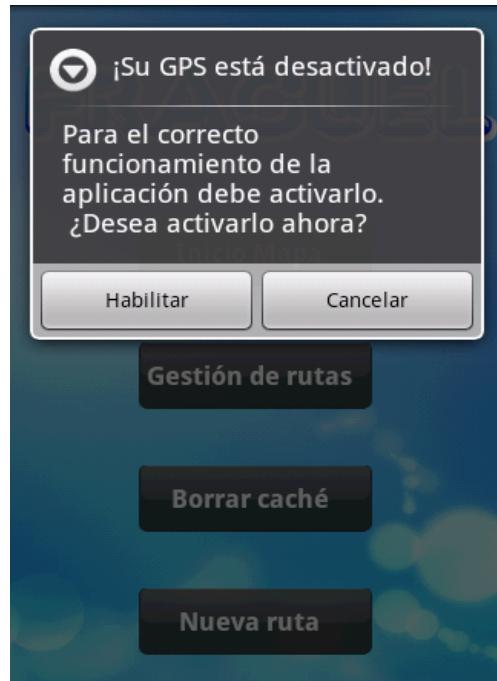


Figura 16: Aviso receptor

Menú principal

En el menú principal de *FRAGUEL* encontrará las siguientes opciones (Figura 17) y la posibilidad de salir de la aplicación, no permitiendo que ésta se ejecute en segundo plano, pulsando sobre el botón *Menú* de su teléfono, el cual desplegará la opción de salir (Figura 18).



Figura 17: Menú principal



Figura 18: Opción salir.

- Si desea comenzar a usar la aplicación pulse *Inicio Mapa*.
- Si desea gestionar las rutas (añadir rutas nuevas, eliminar rutas de su terminal y ver la información de cada punto de las rutas) pulse *Gestión de rutas*.
- Si desea liberar espacio en la tarjeta de memoria y eliminar todas las imágenes de las rutas disponibles ya descargadas en su terminal, seleccione *Borrar Caché*.
- Si desea crear su propia ruta, pulse *Nueva Ruta*.



Inicio Mapa y uso de la aplicación

Tras haber pulsado sobre el botón Inicio Mapa, la aplicación le situará en el mapa mediante un punto azul y podrá aparecer un círculo también azul cuyo centro es su posición, que indica el posible error de localización (Figura 19).

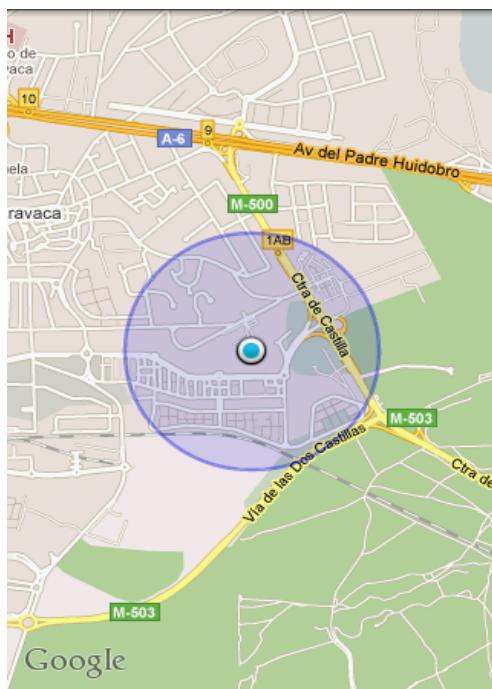


Figura 19: Mapa y su posición.



Figura 20: Menú mapa.

Podrá moverse por el mapa libremente, descubriendo los puntos de interés de las rutas y accediendo a la información multimedia que estos contienen (Figura 21, véase sección *Multimedia*). Si se mueve por el mapa siempre podrá volver a su posición pulsando el botón que le aparece en la parte inferior izquierda de la pantalla (Figura 22).



Figura 21: Información multimedia PI. posición.



Figura 22: Volver a mi

FRAGUEL tiene la posibilidad de usarse de dos modos diferentes:

- **Modo libre:**

Habiendo iniciado el mapa, el usuario podrá caminar libremente y la aplicación le notificará que está dentro de la zona de un punto de interés cuando entre en un radio de 50 metros de distancia con éste, mostrándole y reproduciéndole acústicamente la información y otorgándole la posibilidad al usuario de acceder al contenido multimedia de dicho punto de interés (Figura 23) pulsando sobre cada uno de los iconos (véase sección *Multimedia*).



Figura 23: Punto de Interés e Información multimedia asociada.

Si sale del radio de acción del punto de interés, volverá automáticamente al mapa. Si pulsa el botón Menú de su terminal accederá a las diferentes opciones como: volver a reproducir el texto, detener la reproducción, atrás y volver al menú principal (Figura 24).



Figura 24: Menú del PI.



- **Modo ruta:**

Habiendo iniciado el mapa, el usuario podrá iniciar una ruta por la que la aplicación le guiará visitando todos los puntos de la misma, accediendo a todo el contenido multimedia de la ruta.

Para iniciar una ruta el usuario deberá mantener pulsado el dedo sobre el mapa o bien seleccionar la opción *Comenzar Ruta* en el menú del mapa (Figura 20). En cualquier caso se le mostrará un diálogo para elegir la ruta y el punto de comienzo de la misma (Figura 25 y 26).

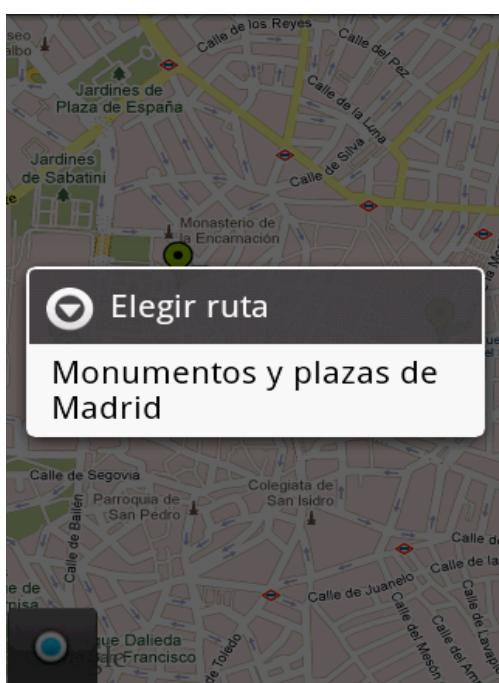


Figura 25: Selección de ruta.



Figura 26: Selección de origen

Al seleccionar el punto de origen se mostrará y reproducirá una introducción a la ruta y posteriormente se le mostrará en el mapa los puntos de la ruta, en rojo los puntos visitados y en verde los puntos que restan por visitar (Figura 27 y Figura 28).



Figura 27: Introducción a la ruta.

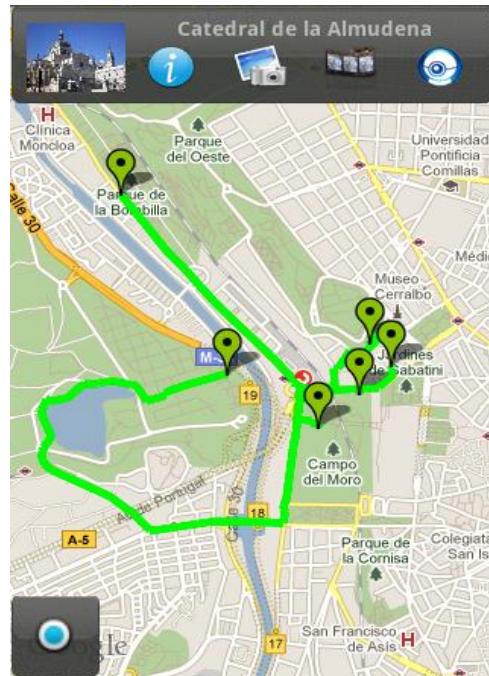


Figura 28: Ruta en el mapa.

Como puede apreciar en la parte superior de la Figura 27, se muestra un mensaje con los metros que quedan al siguiente punto de la ruta y la dirección cardinal a la misma. Cada vez que se entre en el radio de acción del siguiente punto de la ruta se mostrará el punto de interés y se reproducirá la descripción del mismo, pudiendo acceder al contenido multimedia del mismo (Figura 23, véase sección *Multimedia*).

Multimedia

Según el botón que pulsemos en los mensajes o avisos podremos acceder a un contenido multimedia u otro perteneciente al punto seleccionado.

- Si pulsamos podremos acceder a la información textual que además se reproduce acústicamente (Figura 29).
- Si pulsamos accederemos a la galería de fotos de dicho punto de interés.(Figura 30)
- Si pulsamos visualizaremos el vídeo asociado a dicho punto (Figura 31).
- Si pulsamos veremos el contenido de realidad aumentada asociado al punto (Figura 32)



Catedral de la Almudena - Madrid

La catedral de Santa María la Real de la Almudena es la sede episcopal de la Archidiócesis de Madrid, (España). Se trata de un templo de 102 metros de longitud y 73 de altura, construido durante los siglos XIX y XX en una mezcla de diferentes estilos: neoclásico en el exterior, neogótico en el interior y neorrománico en la cripta.

Fue consagrada por el pontífice Juan Pablo II en su cuarto viaje a España, el 15 de junio de 1993, siendo de este modo la única catedral española dedicada por un papa.

Está ubicada en el centro de la ciudad. La fachada principal se encuentra frente al Palacio Real. La fachada del crucero mira hacia la calle de Bailén, y el acceso a la cripta se

Monumentos y plazas de Madrid



Usted está viendo la imagen número 1. Punto 'Palacio Real de Madrid' de la ruta 'Monumentos y plazas de Madrid'

Figura 29: Información textual

Figura 30: Galería de Imágenes.



Figura 31: Vídeo del punto.



Figura 32: Realidad Aumentada del PI.

Gestión de rutas

El usuario podrá añadir o eliminar las rutas descargadas en su terminal así como ver la información de cada uno de los puntos de interés si así lo desea. Despliegue las opciones pulsando sobre el botón *Menú* de su terminal y elija la opción que desee: Añadir o Eliminar una ruta existente (Figura 33).

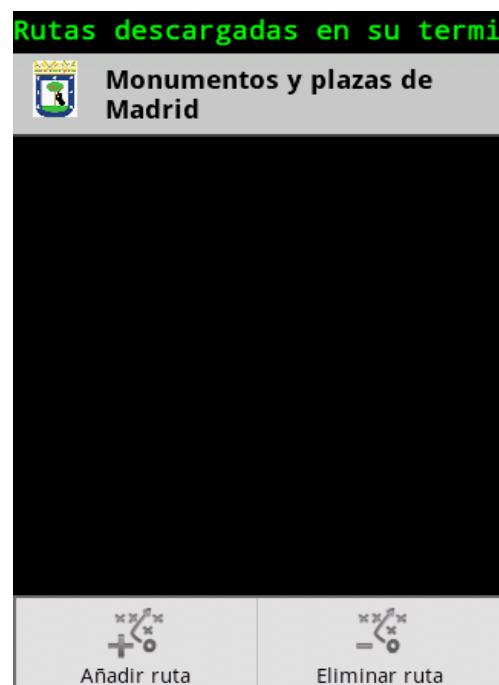


Figura 33: Opciones gestión rutas.



Si selecciona la opción *Añadir ruta*, la aplicación conectará con el servidor y le ofrecerá las rutas disponibles. En cambio si escoge la opción *Eliminar ruta*, podrá eliminar cualquiera de las rutas descargadas en su terminal.

Nueva Ruta

FRAGUEL da la capacidad a los usuarios de crear sus propias rutas a través de dos opciones (Figura 34):

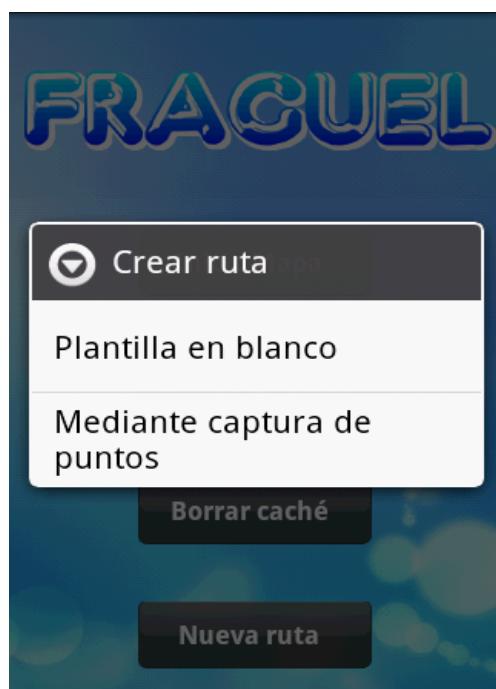


Figura 34: Opciones de creación de rutas.

- Plantilla en blanco:

El usuario introducirá el nombre del fichero, el nombre de la ruta y el número de puntos que desee. Si pulsa aceptar se le notificará que la ruta se ha creado con éxito almacenándose en la tarjeta de memoria externa, en la ruta `/sdcard/fraguel/user` (Figura 35).

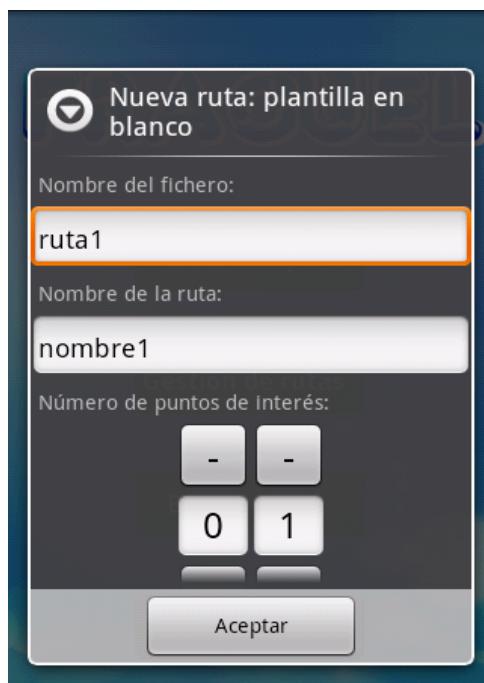


Figura 35: Creación ruta en blanco

- Mediante captura de puntos:

El usuario introducirá el nombre de la ruta y a continuación se mostrará una pantalla donde se mostrarán las coordenadas GPS actuales (Figura 36). Al pulsar *Capturar* se pedirá el nombre para dicho punto y se volverá a la misma pantalla para seguir capturando puntos. Si pulsa *Finalizar*, la aplicación le pedirá en que formato desea guardarlo, si temporal o generar el .xml para añadirlo a sus rutas.

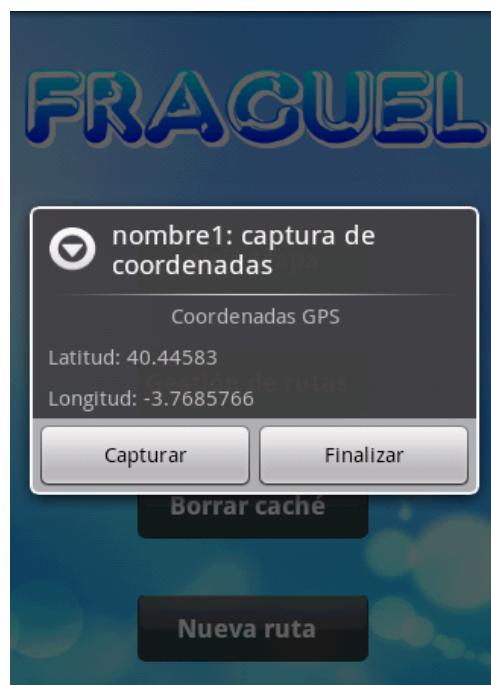


Figura 36: Captura de puntos.

Por último, las rutas generadas por el usuario podrán cargarse en la aplicación si se mueven a la carpeta `/sdcard/fraguel/routes`. Una vez las nuevas rutas estén ubicadas en esa carpeta, éstas estarán disponibles en el sistema la próxima vez que se inicie la aplicación.



IX

Referencias y Bibliografía



Referencias

1. IBM CoordinateConversion: pequeña clase realizada por un empleado de IBM y que realiza conversión de coordenadas de grados a metros sin perder precisión.
 - a. <http://www.ibm.com/developerworks/java/library/j-coordconvert/>
2. OpenGL: OpenGL (Open Graphics Library) es una especificación estándar que define una API multilenguaje y multiplataforma para escribir aplicaciones que produzcan gráficos 2D y 3D. La interfaz consiste en más de 250 funciones diferentes que pueden usarse para dibujar escenas tridimensionales complejas a partir de primitivas geométricas simples, tales como puntos, líneas y triángulos. Fue desarrollada originalmente por Silicon Graphics Inc. (SGI) en 1992 y se usa ampliamente en CAD, realidad virtual, representación científica, visualización de información y simulación de vuelo. También se usa en desarrollo de videojuegos, donde compite con Direct3D en plataformas Microsoft Windows.
 - a. <http://www.opengl.org>
3. Pandora Interactive es una empresa española de carácter tecnológico dedicada al desarrollo de aplicaciones a medida en realidad aumentada y Plataformas Online tanto para teléfonos móviles, consolas o PC.
 - a. <http://www.pandorainteractive.com/>
4. Scrum Home es un modelo de proceso diseñado para la gestión y desarrollo de software es un proceso iterativo e incremental utilizado comúnmente en entornos basados en el desarrollo ágil de software.
 - a. <http://www.scrum.org>
5. Scrum Guide: guía para comprender el funcionamiento de la metodología Scrum.
 - a. <http://www.scrum.org/storage/scrumguides/Scrum%20Guide.pdf>



6. Wikipedia - Realidad aumentada: es el término que se usa para definir una visión directa o indirecta de un entorno físico del mundo real, cuyos elementos se combinan con elementos virtuales para la creación de una realidad mixta en tiempo real. Consiste en un conjunto de dispositivos que añaden información virtual a la información física ya existente, es decir, añadir una parte sintética virtual a lo real. Esta es la principal diferencia con la realidad virtual, puesto que no sustituye la realidad física, sino que sobreimprime los datos informáticos al mundo real.
 - a. http://es.wikipedia.org/wiki/Realidad_aumentada
7. Wikipedia - Smartphone: es un término comercial para denominar a un teléfono móvil que ofrece más funciones que un teléfono celular común.
 - a. <http://es.wikipedia.org/wiki/Smartphone>
8. Wikipedia – Scrum: visión externa sobre este modelo de proceso.
 - a. <http://es.wikipedia.org/wiki/Scrum>
9. Min3D: librería gráfica de alto nivel, pensada para Android, que hace uso de OpenGL. Emplea Java con OpenGL ES, haciéndolo compatible con todas las versiones de Android superiores a v1.5 y a todas las versiones de OpenGL ES superiores a la 1.0.
 - a. <http://code.google.com/p/min3d>
10. OpenGL ES: es una variante simplificada de la API gráfica OpenGL diseñada para dispositivos integrados tales como teléfonos móviles, PDAs y consolas de videojuegos. La define y promueve el Grupo Khronos, un consorcio de empresas dedicadas a hardware y software gráfico interesadas en APIs gráficas y multimedia.
 - a. <http://www.khronos.org/opengles/>



11. Google Sketchup: es un programa informático de diseño y modelaje en 3D para entornos arquitectónicos, ingeniería civil, diseño industrial, GIS, videojuegos o películas. Es un programa desarrollado y publicado por Google.

a. <http://sketchup.google.com>

12. 3D Studio Max: es un programa profesional de creación de gráficos y animación 3D desarrollado por Autodesk, en concreto la división Autodesk Media & Entertainment (anteriormente Discreet).

a. <http://www.autodesk.es>

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

a. <http://www.eclipse.org>

14. Android: Sistema operativo para dispositivos móviles realizado por Google Inc.

a. <http://www.android.com/>

15. API: Interfaz de programación de aplicaciones, es un conjunto de funciones que abstrae al programador del sistema subyacente,

a. http://es.wikipedia.org/wiki/Interfaz_de_programaci%C3%B3n_de_aplicaciones



16. Google: Google Inc. es la empresa estadounidense propietaria de la marca Google, cuyo principal producto es el motor de búsqueda del mismo nombre. Dicho motor es resultado de la tesis doctoral de Larry Page y Sergey Brin (dos estudiantes de doctorado en Ciencias de la Computación de la Universidad de Stanford) para mejorar las búsquedas en Internet.
- a. <http://www.google.es/about/>
17. Patrón state: patrón de programación de estados. Se utiliza cuando el comportamiento de un objeto cambia dependiendo del estado del mismo.
- a. [http://es.wikipedia.org/wiki/State_\(patr%C3%B3n_de_dise%C3%BAo\)](http://es.wikipedia.org/wiki/State_(patr%C3%B3n_de_dise%C3%BAo))
18. Patrón observer: patrón de programación para escucha de eventos. También es conocido como "spider" y define una dependencia del tipo uno-a-muchos entre objetos, de manera que cuando uno de los objetos cambia su estado, el observador se encarga de notificar este cambio a todos los otros dependientes.
- a. [http://es.wikipedia.org/wiki/Observer_\(patr%C3%B3n_de_dise%C3%BAo\)](http://es.wikipedia.org/wiki/Observer_(patr%C3%B3n_de_dise%C3%BAo))
19. Android Views: sistema de vistas que utiliza Android para que las aplicaciones muestren información por pantalla.
- a. <http://developer.android.com/resources/tutorials/views/index.html>
20. Android Text to Speech: sistema que, mediante una voz sintetizada, es capaz de convertir en voz un texto suministrado.
- a. <http://developer.android.com/resources/articles/tts.html>
21. Bugs: es el resultado de un fallo o deficiencia durante el proceso de creación de software. Dicho fallo puede presentarse en cualquiera de las etapas del ciclo de vida del software aunque los más evidentes se dan en la etapa de desarrollo y programación.
- a. http://es.wikipedia.org/wiki/Error_de_software



Bibliografía

Ableson, Frank y Collins, Charlie y Sen, Robi. Android: Guía para desarrolladores. 1º edición. Madrid: Anaya Multimedia, 2010. 464 páginas. ISBN: 9788441526822.

Alcaraz Espín, Juan José. Introducción a la programación en Symbian. 1º edición. Málaga: Arguval, 2011. 203 páginas. ISBN: 9788496912861.

Brunete, Ed. Android (Programación). 1º Edición. Madrid: Anaya Multimedia, 2011. 288 páginas. ISBN: 9788441528765.

Burnette, Ed. Hello, Android: Introducing Google's Mobile Development Platform. 3º edición. Pragmatic Programmers, LLC, The, 2010. 300 páginas. ISBN: 1934356565, 9781934356562.

Huddleston, Rob. Android para todos. 1º edición. Madrid: Anaya Multimedia, 2011. 272 páginas. ISBN: 9788441529526.

Lawrence Murphy, Mark. Android Programming Tutorials. 1º edición. CommonsWare, LLC, 2009. 422 páginas. ISBN: 0981678025, 9780981678023.

Lawrence Murphy, Mark. The Busy Coder's Guide to Advanced Android Development. 1º edición. CommonsWare, LLC, 2009. 260 páginas. ISBN: 0981678017, 9780981678016.

Petzold, Charles. Programming Windows Phone 7. 1º edición. Redmon, Washington: Microsoft Press, 2010. 952 páginas. ISBN: 978-0-7356-4335-2.

Reto Meier. Professional Android Application Development. 1º edición. Wrox, 2011. 432 páginas. ISBN: 978-0-470-34471-2

Richard S., Jr. Wright. OpenGL / OpenGL Super Bible (Spanish edition). 1º edición. Madrid: Anaya Multimedia, 2005. 1102 páginas. ISBN-10: 8441517940.

W. Frank Ableson, Charlie Collins, y Robi Sen. Unlocking Android: A developers guide. 2º edición. Managing Publications Co, 2011. 592 páginas. ISBN: 9781935182726.

Wright, Richard y Sweet, Michael. Programación en OpenGL. 1º edición. . Madrid: Anaya Multimedia, 1997 752 páginas. ISBN: 9788441501768.



- **Enlaces:**

- Android Developers:

<http://developer.android.com/index.html>

- Parámetros Google Maps:

http://mapki.com/wiki/Google_Map_Parameters

- Blog sobre tratamiento de XML:

http://www.anddev.org/write_a_simple_xml_file_in_the_sd_card_using_xmlserializer-t8350.html

- Rutas sobre Google Maps:

http://www.anddev.org/google_driving_directions_-_mapview_overlaid-t826.html

<http://csie-tw.blogspot.com/2009/06/android-driving-direction-route-path.html#ixzz0rfslHaO2>

- Realidad Aumentada sobre Android:

<http://www.devx.com/wireless/Article/43005>

- Cálculo de el '*bearing*':

<http://stackoverflow.com/questions/4308262/calculate-compass-bearing-heading-to-location-in-android>

- Wikipedia:

<http://es.wikipedia.org/wiki/Wikipedia:Portada>

- Librería Min3d:

<http://code.google.com/p/min3d/>