APLICACIÓN MÓVIL ITINERARIOS GUIADOS PARQUE DEL RETIRO

BUENO GÓMEZ, MIGUEL
SEGOVIA SANZ, ALBERTO
TORRES GONZÁLEZ, VÍCTOR

SISTEMAS INFORMÁTICOS, FACULTAD DE INFORMÁTICA,
UNIVERSIDAD COMPLUTENSE DE MADRID

Sistemas Informáticos en Ingeniería Informática

Directoras:
Dra. Guadalupe Miñana Ropero y Dra. Victoria López López

Autorización de Difusión

a 21 de junio de 2013,

Los abajo firmantes, matriculados en Sistemas Informáticos de la Facultad de Informática, autorizan a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente proyecto fin de carrera de Sistemas Informáticos: "APLICACIÓN MÓVIL ITINERARIOS GUIADOS PARQUE DEL RETIRO", realizado durante el curso académico 2012-2013 bajo la dirección de Dra. Victoria López López y Dra. Guadalupe Miñana Ropero y con la colaboración externa de dirección de Rafa del ayuntamiento de Madrid y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo.

MIGUEL BUENO GÓMEZ, ALBERTO SEGOVIA SANZ Y VÍCTOR TORRES GONZÁLEZ

Resumen

Itinerarios: Jardines del Buen Retiro es un proyecto cuyo objetivo principal es desarrollar una aplicación que permite al usuario realizar una visita auto-guiada por los distintos estilos de jardinería del Parque del Buen Retiro de Madrid, informando al usuario de la riqueza paisajística e histórica del entorno del recorrido.

El objetivo del proyecto es dar una solución virtual a la actual visita guiada que gestionan los educadores del Parque del Buen Retiro, sin pretender sustituir a la visita tradicional. La gran demanda de usuarios que desean realizar la visita guiada justifica plenamente el desarrollo de una aplicación que complemente esta actividad y que además, pueda ser utilizada de forma autónoma por cualquier ciudadano interesado. Además el usuario puede realizar el recorrido de los jardines accediendo a la misma información sin restricciones de fechas ni horarios. Para este cometido se pone en la mano del usuario una aplicación *Android* que además de información en texto, incluye otras ventajas como curiosidades de los puntos del recorrido, imágenes históricas comentadas, información locutada, un mapa del parque con el itinerario y el uso de geolocalización para situar al usuario y guiarle en las rutas.

El proyecto se ha desarrollado cuidando especialmente el rendimiento del dispositivo y se ha obtenido como resultado una aplicación amigable y óptima que permite controlar toda esa información de una manera rápida, cómoda y fácil de usar por usuarios no expertos.

Palabras clave: Itinerarios, Parque del Buen Retiro, Aplicación, Android, Auto-guiado, Medio ambiente, Turismo, Madrid.

Abstract

Itinerarios: Jardines del Buen Retiro is a project whose main objective is to develop an application that allows the user to make a self-guided tour through the different styles of gardening of Retiro Park in Madrid, informing the ouser about the rich landscape and historic environment of the route.

The project aims to provide a solution to the current virtual tour that it is managed by educators from Retiro Park. The high demand of users who want a guided visit fully justifies the development of an application that complements this activity and also can be used independenly by any interested citizen. Furthermore the user can follow the ourte of the gardens accessing the same information without blackout dates or times. For this task, it is put into the user's hand also an *Android* with text information, includes other benefits as curiosities of waypoints, historical pictures, sound information, a map of the park with the itinerary and the use of geolocation to locate the user and guide him on the road.

The project has been developed especially caring the device performance and has been obtained as a result an optimum friendly application that lets control all the information in a quick, convenient and easy to use by non-experts.

Keywords: Route, Buen Retiro Park, Application, Android, Auto-guide, Environment, Tourism, Madrid

Índice de contenidos

Prólogo

Este proyecto nace dentro de un convenio entre la Universidad Complutense y el Área de Medioambiente y Movilidad del Ayuntamiento de Madrid. Su principal objetivo ha sido crear una herramienta *Android* para dispositivos móviles que permita al usuario realizar una visita auto-guiada por los distintos estilos de jardinería del parque del Buen Retiro de Madrid. Esta herramienta pretende dar una solución virtual y complementaria a la actual visita guiada que gestiona el Centro de Información y Educación Ambiental de El Huerto del Retiro.

Para afrontar este reto, Víctor, Alberto y Miguel han tenido que acudir a varias reuniones en el Ayuntamiento de Madrid con los encargados del programa. Estas reuniones han servido para fijar las especificaciones del proyecto y decidir los distintos tipos de formato en los que presentar la información que tiene que mostrar la aplicación. En todo momento, ellos han demostrado una gran capacidad de comunicación, tanto a la hora de presentar y defender su trabajo; como a la hora de aportar ideas para solucionar los requisitos que pedía el cliente.

Quiero destacar el trabajo tan bueno que han hecho tanto en audio como en fotografía. Trabajo que aunque queda fuera del currículo de un ingeniero informático han sabido asumir y desarrollar de una manera muy profesional.

Víctor, Alberto y Miguel han demostrado con este proyecto que son capaces de aplicar los conocimientos adquiridos durante su formación universitaria así como investigar y aprender a solucionar problemas de la vida real utilizando nuevas herramientas informáticas o desarrollándolas ellos mismos. Han demostrado con creces que están preparados para desarrollar proyectos profesionales como Ingenieros en Informática.

Como directoras estamos muy orgullosas y deseamos un futuro lleno de éxitos para nuestros alumnos Víctor, Alberto y Miguel

Guadalupe Miñana Ropero y Victoria López

Agradecimientos

A nuestros seres más queridos por estar ahí en el momento en el que realmente lo hemos necesitado.

Capítulo 1 - Introducción

Presentación

Itinerarios:Jardines del Buen Retiro es un proyecto cuyo objetivo principal es desarrollar una aplicación que permite al usuario realizar una visita auto-guiada por los distintos estilos de jardinería del Parque del Buen Retiro de Madrid, mostrando información de la riqueza paisajística e histórica del entorno del recorrido. La aplicación contiene una muestra selecta de texto, locuciones e imágenes de cada uno de los puntos de la ruta, realizada por expertos en el parque. Esta aplicación es la primera de un conjunto de aplicaciones que se van a desarrollar sobre los distintos recorridos temáticos del Parque del Buen Retiro. En este caso nos encontramos ante la temática de los estilos de jardinería presentes en el parque a lo largo de los siglos y de su evolución, cambio y adaptación en el tiempo. Es una herramienta diseñada única y exclusivamente para dispositivos (móviles, *smartphones*, *tablets*…) bajo sistema operativo *Android*.

Motivación

La motivación adherente a este proyecto surge de la llegada de los dispositivos móviles a nuestros hogares. También ha sido fundamental el lanzamiento del sistema operativo móvil libre y de código abierto (*Android*) y la gran utilidad que tiene, hoy por hoy, un Smartphone en la vida diaria. Esto ha provocado que nuestro interés en desarrollar una aplicación bajo esta plataforma sea inmenso y haya desembocado en el desarrollo de este proyecto.

El desarrollo de aplicaciones móviles para dispositivos *Android* no es una temática impartida en ninguna asignatura de los planes de estudio que hemos cursado, pero las asignaturas que hemos estudiado en la carrera a lo largo de estos años, nos han dado una base de conocimientos suficientemente amplia para poder embarcarnos en un proyecto de este tipo.

Las nuevas tecnologías nos han permitido tener internet en el bolsillo, esto ha provocado fuertes cambios en la sociedad y con ello la necesidad de adaptar y modernizar las ciudades en todos sus servicios y sectores. Entre estos servicios se encuentra el que se dedica a fomentar la

cultura y la educación y concienciación del ciudadano tanto en el medio ambiente como en un desarrollo más sostenible de la ciudad.

En este nuevo escenario, el papel del Ayuntamiento es fundamental. Por ello el Área de Medio Ambiente, Seguridad y Movilidad del Ayuntamiento de Madrid se ha mostrado muy interesado en colaborar con la Facultad de Informática de la UCM, para llevar a cabo una serie de aplicaciones móviles que faciliten e incentiven al ciudadano a usar todos los recursos que este área del Ayuntamiento de Madrid tiene actualmente a su disposición.

Uno de los recursos que proporciona el Área de Medio Ambiente Seguridad y Movilidad del Ayuntamiento de Madrid es el Centro de Información y Educación Ambiental del Retiro. El parque del Buen Retiro es algo más que la zona verde más extensa en pleno centro de la ciudad, es un parque centenario y emblemático. Esto lo ha convertido en un parque muy visitado, pero sólo algunos de los que lo visitan conocen la gran riqueza que contiene. Desde este centro se realizan actividades de información y educación ambiental con la intención de dar a conocer los valores naturales, históricos y culturales de este enclave. Actualmente la manera que tiene de comunicarse con el ciudadano es a través de un blog [62].

De las actividades que se ofrecen en este centro, una de las más demandadas es la *Ruta guiada por los distintos estilos de jardines del retiro*. La gran demanda de usuarios que desean realizar esta visita guiada justifica plenamente el desarrollo de una aplicación que complemente esta actividad y que además, pueda ser utilizada de forma autónoma por cualquier ciudadano interesado. Con esta aplicación el usuario podrá realizar el recorrido de los jardines accediendo a la misma información sin restricciones de fechas ni horarios.

Lo que nos motivó a realizar este proyecto fue el poder aportar una solución virtual a la actual visita guiada que gestionan los educadores del Parque del Buen Retiro. Con esto no se pretende sustituir a la visita tradicional, sino más bien introducir las nuevas tecnologías para poder llegar a más ciudadanos cubriendo una demanda que los educadores no pueden cubrir.

El resultado de este proyecto ha sido una herramienta integral y fiable, que pone en manos del usuario una aplicación *Android* que informa de la riqueza paisajística e histórica de cada punto del recorrido mediante textos que muestran información del estilo de jardinería al que pertenece, curiosidades históricas que allí ocurrieron; imágenes históricas y grabados centenarios con una breve descripción e información locutada. La aplicación también proporciona un mapa del Retiro en el que está marcado el itinerario completo y sobre el que se puede hacer *zoom*. Además permite situar al usuario dentro del parque mediante geolocalización y guiarle en las rutas. En la *Figura 1.1* muestra un esquema de las funcionalidades de la aplicación.

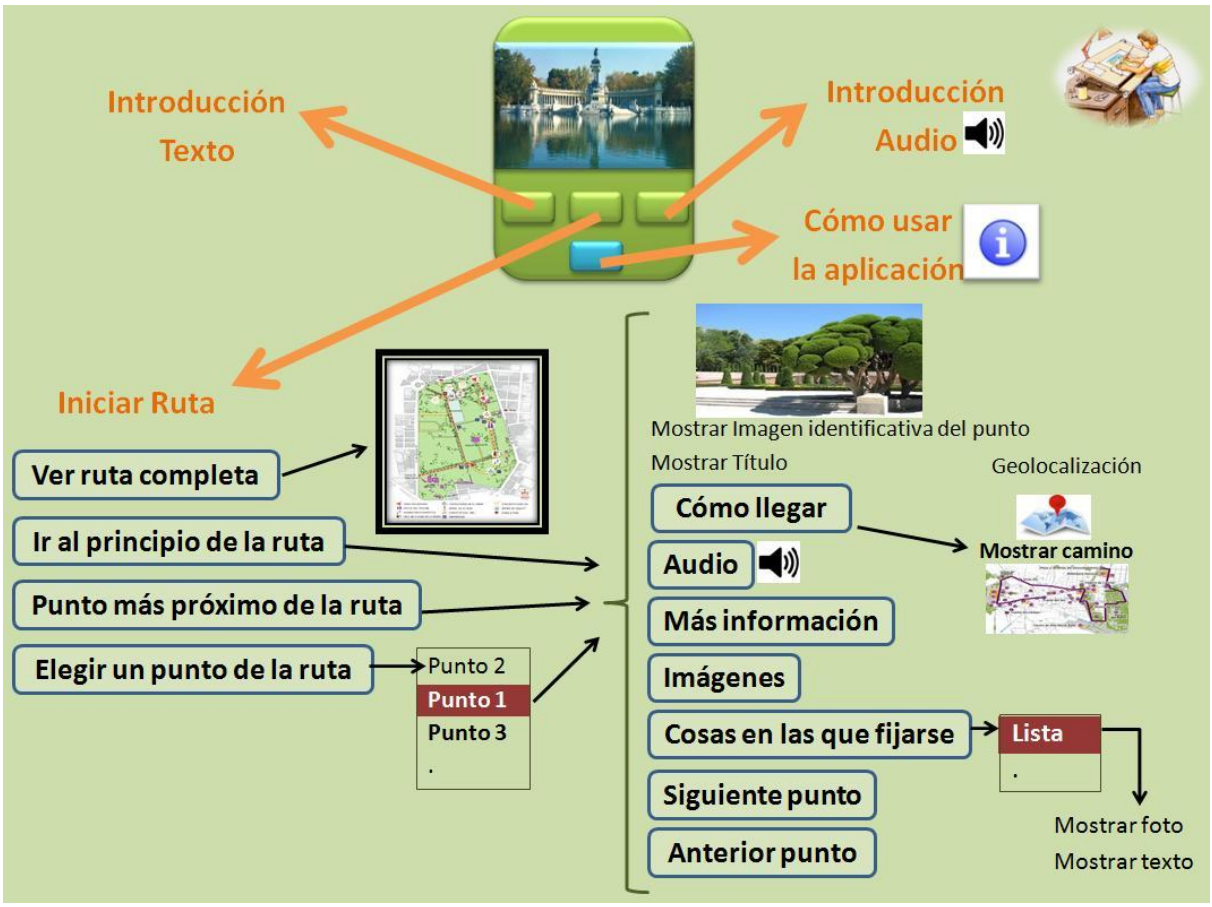


Figura 1.1 .- Esquema general de funcionalidades de Itinerarios:Jardines del Buen Retiro

El proyecto se ha desarrollado cuidando especialmente el rendimiento del dispositivo y se ha obtenido como resultado una aplicación amigable y óptima que permite controlar toda esa información de una manera rápida, cómoda y fácil de usar por usuarios no expertos. La *Figura 1.2* muestra el icono principal de la aplicación.

Figura 1.2 - Icono de Itinerarios: Jardines del Buen Retiro.

Organización de la memoria

Esta memoria está organizada de la siguiente manera: el presente capítulo introductorio donde se encuentra una breve explicación de qué es *Itinerarios: Jardines del Buen Retiro*, la idea y motivación por la cual se desarrolló y sus aspectos fundamentales. En el capítulo 2 se presenta un análisis de las distintas aplicaciones Android presentes en el mercado que han aportado diversas ideas para las funcionalidades de nuestra aplicación. En el capítulo 3 se explican los requisitos, recursos y herramientas necesarias para el desarrollo de este proyecto. El capítulo 4 contiene las especificaciones de todos los casos de uso. La arquitectura del proyecto, el diseño de la aplicación y el diseño de las bases de datos se explican en el capítulo 5. Algunos puntos del proyecto han requerido la implementación de técnicas algorítmicas, estas se describen en el capítulo 6. En el capítulo 7 se pueden ver las operaciones de despliegue. Por último el capítulo 8 recoge las conclusiones y la propuesta de trabajo futuro. La memoria se completa con la bibliografía y con tres anexos; el primero es el historial de revisiones de este documento, el segundo es un manual del usuario en el cual se explica detalladamente el uso de la aplicación y toda su funcionalidad, el último anexo contiene las actas de todas las reuniones llevadas a cabo con el cliente.

Capítulo 2 - Estado del arte

En este capítulo presentamos una selección de aplicaciones presentes en el mercado de la movilidad, que nos han resultado interesantes por su alto parecido a las funcionalidades de nuestra aplicación. Así pues a continuación exponemos un cúmulo de conocimiento ya desarrollado en tecnologías puntera. El objetivo es analizar qué existe en el mercado, qué no gusta, cómo se hace y qué es lo que se puede adaptar a nuestra aplicación.

Guía Madrid – Minube.com

Importante aplicación, no sólo por su buena funcionalidad y crítica, sino además por ser un referente perfecto en nuestro caso concreto, ya que estamos realizando una aplicación sobre Madrid [7]. Aunque nuestro proyecto es sobre una parte muy concreta como es el parque del Buen Retiro de Madrid; esta aplicación, cuya pantalla de inicio se muestra en la *Figura 2.1*, no deja de ser una buena idea a tener en cuenta a la hora de desarrollar.

Figura 2.1 .-Pantalla de inicio de minube.com - Madrid

Con tan solo 10 MB, podemos obtener información de los rincones más recónditos tanto a nivel provincial como estatal. La riqueza cultural que el usuario adquiere al utilizar esta aplicación es realmente notable. La aplicación se puede obtener de forma gratuita a través de *Play Store* de *Android*.

Audioguíasurbanas de Sevilla

Esta aplicación está disponible en el *Play Store* de *Android* [8] y se puede descargar de manera gratuita. Ocupa cerca de 90 *MB*, por lo cual, es posible que no tenga un servidor asociado y desde nuestro punto de vista es demasiado pesado. Su pantalla de inicio se observa en la *Figura 2.2*.

Figura 2.2 .- Pantalla de inicio de audioguiasurbanas - Sevilla

Cuenta con un bonito diseño en la pantalla principal, con una perspectiva del horizonte de la ciudad de Sevilla. Tiene un diseño de botones bastante agradable, idea que captaremos para *Itinerarios: Jardines del Buen Retiro*. Dispone de un botón de información que detalla un resumen de qué hace cada uno de los botones de la aplicación.

Rutas verdes de Barcelona

Esta aplicación está disponible en el *Play Store* de *Android* [10] y se puede descargar de manera gratuita. Ocupa cerca de 17 *MB*. Algunas de las funcionalidades de la aplicación se pueden observar en la *Figura 2.3*:

Figura 2.3 .- Rutas verdes de Barcelona: diferentes tramos de recorrido

Propuesto por el ayuntamiento de Barcelona, invita a todos los ciudadanos a recorrer los espacios verdes de Barcelona con distintos itinerarios. Hay que destacar que informa fielmente acerca de todas las rutas a seguir, de su duración y de la distancia. Es la aplicación de referencia para *Itinerarios: Jardines del Buen Retiro* debido al uso de rutas en mapas.

Guía de Córdoba

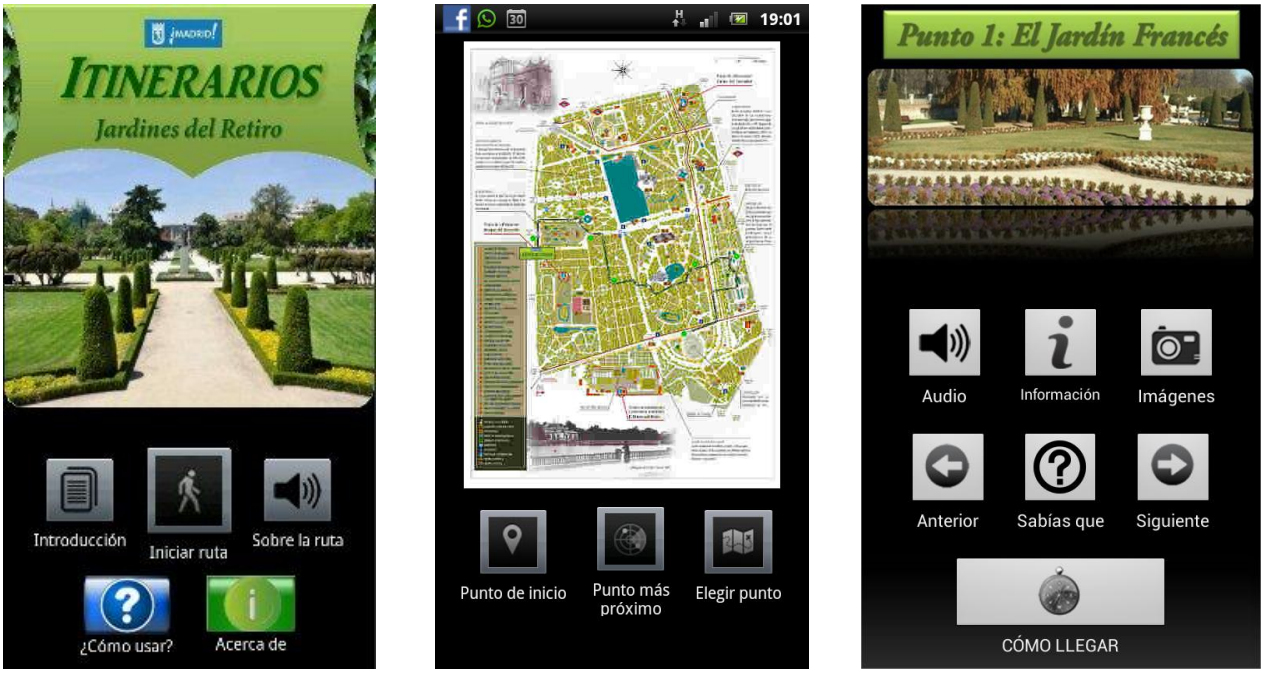
Esta aplicación está disponible en el *Play Store* de *Android* [9] se puede descargar de manera gratuita. Ocupa alrededor de 2 *MB*. Una captura de pantalla interesante sobre cómo se muestra el texto en esta aplicación se puede observar en la *Figura 2.4*:

Figura 2.4 .- Información de monumentos relevantes de Córdoba

Se trata de una aplicación gratuita, de interés turístico y con un diseño muy fino. Técnicamente lo que contiene es información sobre qué hacer en la ciudad de Córdoba, con opciones como localizaciones exactas de los lugares por *Google Maps*, teléfonos de contacto de los lugares, reservas de habitaciones, etc. Muy interesante a nivel de cómo manejar el contenido a mostrar para el usuario.

Nuestra aplicación

Tras el análisis funcional y de diseño de las aplicaciones anteriores hemos obtenido ciertas ideas adaptadas a nuestra aplicación, algunas de las cuales quedan plasmadas en la *Figura 2.5* con subapartados: (a) Pantalla principal de la aplicación; (b) pantalla del inicio de ruta con opciones de elección de punto del itinerario, comienzo desde el inicio, o geolocalización al punto más cercano y (c) captura de pantalla genérica de cada uno de los siete punto del recorrido, dentro de la cual podemos acceder al contenido audio, texto e imágenes; además de poder geolocalizar dicho punto para saber cómo llegar.



(a)Pantalla de inicio (b) Pant. de inicio de ruta (c) Pant. genérica de un punto

Figura 2.5 .- Pantallas principales de Itinerarios: Jardines del Buen Retiro

Capítulo 3 - Especificación de Requisitos y Herramientas

En este capítulo se describe la funcionalidad de la aplicación desde una manera algo más práctica. El usuario, el sistema y los actores externos son clave a la hora de satisfacer los requisitos y dotar de la funcionalidad establecida a la aplicación sirviéndose de toda la tecnología al alcance. Para ello, se ha dividido el capítulo en dos apartados y se explica a continuación el alcance de cada uno de ellos.

- **Especificación de requisitos**, que por un lado describe los requisitos exigidos por el cliente (funcionales y de interfaz), los componentes físicos necesarios para que se cumplan los primeros (hardware) y de los requisitos en los que el usuario no interactúa directamente con el sistema (no funcionales).
- **Herramientas**, en el que se describen las herramientas software usadas para el desarrollo, destacando las importantes (imprescindibles) y las que no (de soporte), pero que sin ellas, hubiese sido más difícil realizar el proyecto. También se incluyen los materiales que se han utilizado para probar la aplicación (recursos físicos).

Especificación de Requisitos de Software

Este apartado se compone de la recopilación de los distintos requisitos que deberá cumplir el sistema a desarrollar. Estos requisitos son tomados como referencia para el desarrollo de la herramienta, por lo que el cumplimiento de funcionalidad del futuro sistema es un chequeo de cada acción permitida por la herramienta con los requisitos aquí descritos. El nivel de detalle expuesto en cada uno de los requisitos es suficiente para que el equipo de desarrollo pueda diseñar un sistema que satisfaga los requisitos y los encargados de las pruebas puedan determinar si éstos se satisfacen.

Cada requisito definido a continuación debe solucionar uno o parte de los objetivos generales definidos, para ello, a cada requisito se le ha asignado un identificador, una descripción de su funcionalidad, el objetivo asociado, una secuencia de ejecución del requisito y el nivel de estabilidad, definición y utilización.

Requisitos hardware

Para el uso de la aplicación será necesario que el usuario cuente con:

- Un *smartphone* o una *tablet* con el sistema operativo *Android* instalado y en la versión 2.3.3 (*Gingerbread*) o superior, ya que la aplicación no puede ser instalada en versiones anteriores.
- Conexión a internet ya sea vía *Wi-fi* o por tarifa de datos contratada. Esta conexión es necesaria para descargarse los recursos consultados en la base de datos y para conectarse a los servicios ofrecidos por *Google Maps.*
- Memoria interna (o SD) en el teléfono suficiente para albergar la aplicación (no más de 3,5 MB).
- GPS para la geolocalización del usuario cuando quiera saber cómo llegar a un punto
- Opcional: memoria extra para almacenar los archivos descargados de la base de datos.

Requisitos de la interfaz

El sistema ofrece una navegación a través de una aplicación intuitiva y visual evitando, en la medida de lo posible, que sea necesaria una formación previa por parte del usuario. Cualquier duda sobre el manejo de la interfaz puede ser resuelta en una pantalla tras pulsar el botón de "cómo usar" de la pantalla del menú principal que hará las veces de manual de instrucciones de la misma. Para mejorar la interfaz se han llevado a cabo lo siguiente:

- Se ha estudiado los colores, imágenes e iconos para que la interfaz sea lo más amena, agradable y fácil de usar posible. Además, los logos fueron diseñados por los integrantes del equipo.
- Se ha pedido consejo a los clientes, tanto al personal del Retiro, como al del ayuntamiento de Madrid

Requisitos funcionales

Estos son los requisitos que el equipo consideró necesarios para poder implementar lo demandado por el cliente y que constituirán la utilidad de la aplicación. Las distintas funcionalidades que el sistema deberá permitir son las siguientes:

- Consultar el Plano del Retiro
- Iniciar una ruta
- Elegir un punto de ruta
- Empezar desde el punto de ruta más cercano

- Ver la galería de imágenes
- Reproducir audios
- Ver textos informativos
- Ver el apartado “Sabías que...”
- Cómo llegar a un punto de ruta
- Cómo usar

Lo cual se verá a continuación en los siguientes requisitos funcionales:

Nº Requisito: RF-1	
Versión 1.0	**Revisado por:** Todo el equipo
Nombre función: Consultar el Plano del Retiro **Prioridad:** Alta **Descripción:** El usuario puede observar, ampliar, alejar y desplazarse por el plano del retiro con la ruta indicada, así como ver los puntos de interés. **Actores:** Usuario **Salida:** Imagen ampliable del plano del retiro a pantalla completa con la ruta y los puntos de interés especificados. **Necesita:** Ninguna necesidad aparte de tener la aplicación instalada. **Acciones:** 1. Abrir la aplicación 2. Pulsar el botón de iniciar ruta 3. Pulsar sobre la imagen en miniatura del plano del retiro	

Tabla 3.1 .- Requisito de la consulta del Plano del Retiro

Nº Requisito: RF-2	
Versión 1.0	**Revisado por:** Todo el equipo

Nombre función: Iniciar ruta

Prioridad: Alta

Descripción: El usuario comienza la ruta desde el primer punto (en el punto del Jardín francés)

Actores: Usuario

Salida: Pantalla con las distintas opciones propias del primer punto de interés.

Necesita: Ninguna necesidad aparte de tener la aplicación instalada.

Acciones:

1. Abrir la aplicación
2. Pulsar el botón de iniciar ruta
3. Pulsar en el botón “Punto de inicio”

Tabla 3.2 .- Requisito de Iniciar ruta

Nº Requisito: RF-3	
Versión 1.0	**Revisado por:** Todo el equipo

Nombre función: Elegir punto

Prioridad: Alta

Descripción: El usuario puede elegir, de entre los 7 puntos que conforman la ruta, el punto de partida que desee.

Actores: Usuario

Salida: Pantalla del punto elegido.

Necesita: Ninguna necesidad aparte de tener la aplicación instalada.

Acciones:

- Abrir la aplicación
- Pulsar el botón de iniciar ruta
- Pulsar el botón “Elegir punto”
- Pulsar un punto de los elegidos

Tabla 3.3 .- Requisito de Elegir punto

Nº Requisito: RF-4	
Versión 1.0	**Revisado por:** Todo el equipo

Nombre función: Empezar desde el punto más cercano

Prioridad: Alta

Descripción: El usuario puede iniciar su ruta desde el punto más cercano a la posición actual del mismo.

Actores: Usuario

Salida: Pantalla del punto más cercano.

Necesita: Disponer de GPS.

Acciones:

- Abrir la aplicación
- Pulsar el botón de iniciar ruta
- Pulsar sobre el botón de “Empezar desde el punto más cercano”

Interacciones: */+GPS.

Precondición: El GPS debe estar habilitado.

Situaciones anormales: Si la señal de GPS es débil, el sistema podría tardar demasiado.

Tabla 3.4 .- Requisito de Empezar desde el punto más cercano

Nº Requisito: RF-5	
Versión 1.0	**Revisado por:** Todo el equipo

Nombre función: Ver la galería de imágenes

Prioridad: Alta

Descripción: El usuario tiene la posibilidad de elegir entre las imágenes disponibles dispuestas en forma de lista con una breve descripción. Además se puede ver cada una de las imágenes a pantalla completa y deslizar el dedo para pasar de una a otra imitando diapositivas.

Actores: Usuario

Salida: Lista de las imágenes disponibles en un punto.

Necesita: Internet y 20MB libres en la memoria SD.

Acciones:

- Abrir la aplicación
- Pulsar el botón de iniciar ruta
- Usar cualquier combinación que lleve a un punto de la ruta
- Pulsar el botón imágenes

Interacciones: Base de datos

Precondición: Disponer de conexión

Situaciones anormales: Posible demora en la carga si la conexión no es buena. Imposibilidad de guardar las imágenes en la memoria SD si no se posee el espacio suficiente.

Tabla 3.5 .- Requisito de ver la galería de imágenes

Nº Requisito: RF- 6	
Versión 1.0	**Revisado por:** Todo el equipo

Nombre función: Reproducción de audios

Prioridad: Alta

Descripción: El usuario puede escuchar locuciones narradas que informan sobre detalles importantes del punto en el que se encuentren. Se puede repetir el audio, pausar y reanudar en cualquier momento y también existe una barra de progreso que muestra el estado de la reproducción.

Actores: Usuario

Salida: Pantalla de reproducción del audio, mientras el audio comienza a reproducirse.

Necesita: Internet

Acciones:

Existen dos posibilidades:

1.
   - Abrir la aplicación
   - Pulsar el botón de audio
2.
   - Usar cualquier combinación que lleve a un punto de la ruta
   - Pulsar el botón de audio

Interacciones: Base de datos

Precondición: Disponer de conexión

Situaciones anormales: Cortes en el audio si la conexión es de un bajo ancho de banda.

Tabla 3.6 .- Requisito de reproducción de audios

Nº Requisito: RF-7	
Versión 1.0	**Revisado por:** Todo el equipo
Nombre función: Ver textos informativos **Prioridad:** Alta **Descripción:** El usuario podrá leer información de un punto en formato de texto, pudiéndose desplazar por él deslizando el dedo. **Actores:** Usuario **Salida:** Pantalla con el texto en cuestión. **Necesita:** Ninguna necesidad aparte de tener la aplicación instalada. **Acciones:** • Usar cualquier combinación que lleve a un punto de la ruta • Pulsar el botón de información.	

Tabla 3.7 .- Requisito de visualización de textos informativos

Nº Requisito: RF-8	
Versión 1.0	**Revisado por:** Todo el equipo
Nombre función: Ver sabías qué **Prioridad:** Alta **Descripción:** El usuario podrá leer acerca de curiosidades y árboles importantes de un punto en formato de texto, pudiéndose desplazar por él deslizando el dedo. **Actores:** Usuario **Salida:** Pantalla con el texto en cuestión. **Necesita:** Ninguna necesidad aparte de tener la aplicación instalada. **Acciones:** • Usar cualquier combinación que lleve a un punto de la ruta • Pulsar el botón de “Sabías qué”.	

Tabla 3.8 .- Requisito de visualización del apartado “Sabías que...”

Nº Requisito: RF-9	
Versión 1.0	**Revisado por:** Todo el equipo

Nombre función: Mostrar ruta

Prioridad: alta

Descripción: El usuario tiene la posibilidad de averiguar cómo llegar desde el punto en el que se encuentra hasta el punto que quiere visitar dibujándose la ruta en un mapa.

Actores: Usuario, *Google Maps*

Salida: Pantalla con el mapa y la ruta.

Necesita: GPS habilitado y conexión a internet.

Acciones:

- Usar cualquier combinación que lleve a un punto de la ruta
- Pulsar el botón de cómo llegar.

Interacciones: Google maps y GPS

Precondición: Existe conexión a internet y el GPS está habilitado

Situaciones anormales: GPS con señal baja, puede hacer que tarde mucho en calcular la ruta. Posibles fallos de conexión o problemas con Google maps.

Tabla 3.9 .- Requisito de reproducción de audios

Nº Requisito: RF-10	
Versión 1.0	**Revisado por:** Todo el equipo
Nombre función: Cómo usar la aplicación **Prioridad:** alta **Descripción:** El usuario tiene la posibilidad de consultar una pantalla de ayuda si no sabe cómo usar la aplicación **Actores:** Usuario **Salida:** Pantalla con iconos que ilustran cómo se usa la aplicación **Necesita:** necesidad aparte de tener la aplicación instalada. **Acciones:** • Iniciar la aplicación • Pulsar en el botón "Cómo usar"	

Tabla 3.10 .- Requisito de Cómo usar la aplicación

Requisitos no funcionales

En este apartado se describen todas aquellas condiciones que deben cumplirse y que, a su vez, son invisibles al usuario, para que la aplicación desarrolle toda su funcionalidad de manera óptima:

- Aquellas funcionalidades que requieran de otros servicios para ser realizadas (bases de datos, *Google Maps* o GPS), deberán desistir en las peticiones realizadas pasado un tiempo determinado por los desarrolladores. Esto se ha pensado así para dar fluidez y dinamismo a la aplicación y evitar que la misma se bloquee y no permita al usuario seguir usándola.
- Se debe informar al usuario de cualquier comportamiento inesperado del sistema, así como de cualquier error o demora excesiva en alguna consulta, de manera que éste sepa si algo ha fallado y por qué.

Herramientas

En el desarrollo de un proyecto, es necesario servirse de apoyo de multitud de servicios que están disponibles para aquel que quiera utilizarlos. En este sentido se pretende informar de lo que se ha usado para llevar a cabo *Itinerarios: Jardines del Buen Retiro.* En esta sección se presentan, con definiciones breves, las herramientas y recursos que se han utilizado a lo largo del proyecto y muchos de estos recursos, así como tutoriales de uso, que también están almacenados en la bibliografía del final del documento. En cada apartado se explica una breve motivación de su utilización, lo que ofrece y dónde se puede encontrar más información sobre lo aquí escrito.

Herramientas software imprescindibles

Para desarrollar un programa informático, ya sea una aplicación móvil o de escritorio, se necesitan una serie de programas, por ejemplo para compilar el código fuente, o para observar que lo programado es correcto. En este apartado se quiere informar de los programas que, sin ellos, no hubiese sido posible el desarrollo de *Itinerarios: Jardines del Buen Retiro*.

Sistema operativo de la aplicación

El desarrollo de un proyecto informático en la actualidad supone la elección del sistema operativo sobre el que se quiera utilizar. Independientemente del ámbito del proyecto y el usuario final que vaya a utilizarlo, siempre es interesante apostar por las plataformas *open-source*, así como el software no privativo.

Hoy en día, estamos en la era del teléfono móvil. Se utiliza para leer el correo, navegar por Internet, comunicarse con los amigos e incluso para videoconferencias. Según las estadísticas, en este año 2013, existirán tantos dispositivos móviles como personas existan en el mundo [54].

Por tanto, es importante desarrollar un proyecto para tecnologías móviles que llegue al máximo número de personas. Para ello, si utilizamos como sistema operativo una plataforma *open-source*, se llegará a más público, ya que no se tendrá que pagar por el propio sistema operativo. Además, si este sistema operativo tiene muchas similitudes con el lenguaje de programación Java, y no sólo eso, sino que también está basado en él, se hace muy atractiva la idea de usar *Android* como plataforma sobre la que sentar los cimientos de una aplicación móvil.

Android versión 2.3.3 - Gingerbread

Se trata de un sistema operativo basado en Linux, diseñado principalmente para dispositivos móviles con pantalla táctil como *smartphones* o *tablets*. La versión 2.3.3 (*Gingerbread*) ha sido elegida por el equipo de *Itinerarios: Jardines del Buen Retiro* principalmente por los siguientes recursos que ofrece:

- Soporta Google Maps
    - Los dispositivos móviles que tienen *Gingerbread* vienen ya con GPS incorporado
- Reproduce archivos multimedia
- Utiliza SQLite como base de datos nativa
- Es un sistema operativo *multithreading*
- Soporta descarga externa mediante el protocolo *HTTP*

Estos recursos son aquellos que nuestra aplicación necesita y al no ser una de las versiones más modernas, se cubre el 80% de los usuarios que utilizan un dispositivo móvil con sistema operativo *Android*. Esto es posible gracias a la *retro-compatibilidad* de las aplicaciones de sistema operativos más modernos. Por ejemplo, una aplicación diseñada para la versión 2.0 de *Android*, también funciona en las versiones más modernas, como la 4.1.2. En la *Figura 3.1* se puede comprobar lo dicho anteriormente, y cabe mencionar que el porcentaje de usuarios que pueden usar una aplicación diseñada para *Gingerbread* va creciendo día a día. Además, cuenta con una extensa y útil documentación que ha guiado al equipo siempre que ha tenido alguna duda [14]. A continuación en la *Figura 3.1*, se puede ver que el 95,1% de los usuarios con dispositivos *Android* puede utilizar la aplicación debido a la retro-compatibilidad.

Versión	Nombre en código	Fecha de distribución	API level	Cuota (1 Mayo, 2013)
4.2.x	*Jelly Bean*	Noviembre 13, 2012	17	2.3%
4.1.x	*Jelly Bean*	Julio 9, 2012	16	26.1%
4.0.x	*Ice Cream Sandwich*	Diciembre 16, 2011	15	27.5%
3.2	*Honeycomb*	Julio 15, 2011	13	0.1%
2.3.3–2.3.7	*Gingerbread*	Febrero 9, 2011	10	38.4%
2.3–2.3.2	*Gingerbread*	Diciembre 6, 2010	9	0.1%
2.2	*Froyo*	Mayo 20, 2010	8	3.7%
2.0–2.1	*Eclair*	Octubre 26, 2009	7	1.7%
1.6	*Donut*	September 15, 2009	4	0.1%
1.5	*Cupcake*	April 30, 2009	3	

Figura 3.1 .- Versiones de Android más usadas desde Mayo del 2013

Entorno de desarrollo

Para programar en *Android* se usa el ***Java Development Kit (JDK)*** que permite la ejecución de programas Java y para el desarrollo se utiliza el entorno de desarrollo integrado **Eclipse**. Desde este último, es necesaria la integración del **ADT *Plugin*** para Eclipse con la finalidad de desarrollar proyectos *Android*. Además, con este *plugin* se pueden descargar las APIs de *Android* con el SDK *Manager* y emular dispositivos virtuales *Android* con el **AVD *Manager*** para observar cómo se comporta un dispositivo por dentro. Para terminar, se necesita conocer la propia base de datos que se crea en el dispositivo virtual, exportarla y analizarla con el programa **SQLite Administrator.**

Java (JDK – Java Development Kit versión 1.7.0_21)

El sistema operativo *Android* está basado en la máquina virtual *Dalvik*, muy parecida a la de *Java* [15]. A lo largo de la carrera se ha tenido que programar en Java, y gracias a ello la curva de aprendizaje desarrollando en *Android* se hizo más horizontal. En la *Figura 3.2* se muestra el logo de Java.

Figura 3.2 .- Logo de Java

Java se necesita para el uso de múltiples aplicaciones como Eclipse y su *ADT plugin* para proyectos *Android.* Además, este proyecto consta de un mini-proyecto Java con la motivación de la pre-ejecución de algoritmos cargados para su siguiente inserción en la aplicación.

Eclipse (IDE, versión Juno 4.2.2)

Es un programa *software* muy extendido que está compuesto por un conjunto de herramientas de programación de código abierto multiplataforma [16]. Básicamente es un entorno de desarrollo integrado que nos ha permitido programar aplicaciones *Android* gracias a la incorporación del *plugin ADT* a la batería de *plugins* propia de Eclipse. A continuación en la *Figura 3.3* se muestra el logo de Eclipse.

Figura 3.3 .- Logo de Eclipse

Este IDE (*Integrated Development Environment*) supone grandes facilidades y atajos a la hora de programar y de depurar programas en Java. Se tiene mucha experiencia en el uso de sus herramientas y era interesante para empezar a programar en *Android* debido a sus similitudes con Java.

ADT Plugin

El *plugin* ADT (*Android Developer Tools*) de Eclipse permite explotar todas las opciones al programador para construir aplicaciones *Android* con un entorno de desarrollo que integra todo

[17]. Así como la exportación del paquete (.apk) a instalar en los dispositivos *Android* para, a posteriori, proceder a su ejecución. A continuación se visualiza el logo de *ADT Plugin* en la *Figura 3.4*.

Figura 3.4 .- Logo de ADT Plugin

El desarrollo en Eclipse con este *plugin* favorece la programación y supone la incorporación a la depuración del emulador de dispositivos *Android AVD Manager*. Después de descargarlo, se podrá usar el *SDK Manager* desde donde se pueden obtener todas las mejoras, actualizaciones y herramientas relacionadas con *Android*, así como el uso y disfrute de sus APIs como explicaremos en el apartado siguiente.

Simulador de dispositivos Android – AVD Manager

El *AVD Manager* es un emulador de dispositivos *Android* que nos sirvió en los inicios de nuestro proyecto [18]. Desde entonces, este simulador bastó para un aprendizaje rápido y para la puesta en contexto de qué era *Android* y cómo era el uso más común de un dispositivo.

Posteriormente, se descubrió que se podía depurar directamente desde nuestro dispositivo *Android*. Con ello se dejó de utilizar el emulador para utilizar nuestro propio dispositivo con el que se tenía más control. Aún así, para el gestor de la base de datos el emulador fue una gran ayuda debido a que se puede ver en tiempo real la distribución de la memoria y descargar la propia base de datos SQLite creada. SQLite Administrator

Con este programa *freeware* se puede administrar de manera visual la base de datos SQLite interna del dispositivo [19]. Ha sido de gran ayuda a la hora de crear la base de datos y ver su correcto funcionamiento en ejecución.

Librerías externas utilizadas

Cuando se desarrolla *software*, es importante conocer todos los recursos disponibles en la actualidad dispuestos a ser usados. La principal razón de la programación orientada objetos es la de reutilizar código: si algo sirve y funciona bien, ¿por qué no utilizarlo?

Para este proyecto se han utilizado tres librerías externas a Android y a nuestro código. Las elegidas son las siguientes:

- *Google Maps* debido a la necesidad de trazar una ruta de un punto cualquiera del parque del Retiro a un punto destino específico, mientras internamente se recalculaba el camino a seguir
- *UniversalImageLoader* debido al requisito de obtener varias imágenes descargadas de Internet desde un punto específico, además de mostrarlas de una manera elegante
- La clase *ScaleImageView* sirve para aplicar *zoom* a imágenes del proyecto

A continuación, se ofrece más información de cada una de ellas.

Google Maps APIv2 para Android

Una de las funcionalidades de la aplicación es el trazado de rutas por lo que ha sido necesario la utilización de librerías externas para el posicionamiento del usuario y la interfaz del mapa. La opción más fácil, referente y gratuita fue *Google Maps* y su API (*Application Programming Interface*).

Mediante el *SDK Manager* se pueden descargar todos los paquetes necesarios para el correcto funcionamiento dentro de la aplicación. Pero no sólo es necesario eso, ya que se necesita la API *Key* de *Google Maps* para identificar la aplicación ante *Google* [44].

Para la integración de mapas en un proyecto Android, puede seguir la siguiente guía que se encuentra en la bibliografía [45].

Galería de imágenes – UniversalImageLoader

Otra de las funcionalidades de nuestra aplicación, es la descarga de una galería de imágenes desde el lugar de una ruta dada una URI (*Uniform Resource Identifier*). La librería *UniversalImageLoader* se descarga de Github [37]. A continuación en la *Figura 3.5* se visualiza un ejemplo práctico de esta librería aplicado a nuestro proyecto.

Figura 3.5 .- Un ejemplo de uso de UniversalImageLoader en el proyecto

Esta librería ofrece muchas funcionalidades, pero lo que se necesitaba se encontró en la lista de imágenes que ofrece la descripción y descarga de una imagen alojada en un servidor externo. De esta manera, se pudo hacer que el peso del paquete de la instalación rondase los 3 *MB.*

Esta libería se puede usar a partir del sistema operativo 2.0 de Android, una versión anterior a la *Gingerbread* de la que partimos en un inicio. Se siguieron los pasos para su uso e instalación dentro del proyecto, con el fin de integrarlo y que sirviera para nuestro proyecto.

Zoom – ScaleImageView

De la misma manera que en el apartado anterior, se necesitaba algo para poder realizar *zoom* en las imágenes que muestra la aplicación. Como siempre, se intenta investigar qué hay disponible y si se ajusta a lo necesitado. Se llegó a la conclusión de que el uso de la clase *ScaleImageView* era lo más indicado. Se puede descargar desde *Github* [43]. Destacar que se ha hecho uso de estas dos últimas librerías combinadas entre sí, aprovechando al máximo sus utilidades, dejando un buen acabado en la aplicación.

Herramientas software de soporte

Se han utilizado las siguientes herramientas de soporte para llevar a cabo este proyecto. No son imprescindibles en el desarrollo, pero sí han ayudado bastante a la gestión del mismo y merecen una ligera mención.

SVN – Subversion

Subversion es un sistema de control de versiones diseñado para reemplazar al popular *CVS*. *Subversion* puede acceder al repositorio a través de redes, lo que permite que distintas personas desde distintos lugares puedan acceder al mismo, modificar sus ficheros. Esto permite, por supuesto, el trabajo en paralelo. De esta manera, se ha incrementado el ritmo de trabajo de los integrantes del equipo *Itinerarios: Jardines del Buen Retiro.* A continuación se explicará el cliente de *Windows TortoiseSVN*.

TortoiseSVN

TortoiseSVN es un sencillo cliente SVN que se integra dentro del propio explorador de carpetas de *Windows* y permite una gestión más cómoda de todo sin tener que estar escribiendo órdenes desde línea de comandos. Este sencillo programa a la par que completo, se puede descargar desde su página web [20]. A continuación se muestra en la *Figura 3.6* el logo de TortoiseSVN.

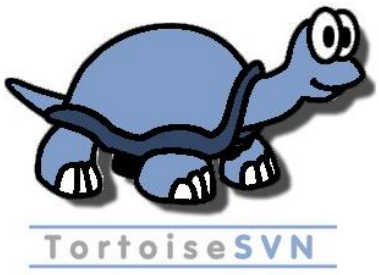


Figura 3.6 .- Logo de TortoiseSVN

Edición de recursos multimedia (imágenes y de audio)

En *Itinerarios: Jardines del Buen Retiro* se visualizan imágenes propias del diseño de la aplicación y se escuchan algunos audios:

- Cada imagen tiene:
    - Una resolución aproximada de 1024 x 800 píxeles

    - Un tamaño cercano al megabyte.
- Cada locución o audio tiene:
    - Una duración entre 30 y 50 segundos
    - Un tamaño cercano al megabyte
    - Una locución grabada por un profesional
    - Filtrado del sonido

A continuación se introducirán las herramientas utilizadas para las imágenes: *GIMP*, *ImageConverter* y *PhotoShop*, y en el caso del audio se hablará del *Cubase*.

GIMP

GIMP (*GNU Image Manipulation Program*) es un programa de edición de imágenes digitales en forma de mapa de bits, tanto de dibujos como fotografías. Es un programa libre y gratuito que forma parte del proyecto GNU y se distribuye bajo la licencia pública general de GNU (más conocida como GNU GPL). A continuación se muestra el logo de GIMP en la *Figura 3.7*.

Figura 3.7 .- Logo de GIMP

A la hora de redimensionizar las imágenes o bajarles la calidad para que no ocupasen demasiado se ha utilizado este programa. Es bastante completo y dispone de muchas utilidades y herramientas. El programa se puede descargar en la página web referenciada por la bibliografía [39].

ImageConverter

Este software ha sido utilizado únicamente para la redimensionamiento de las imágenes por lotes o *batch [38]*. Cuando se programa para dispositivos de diferente tamaño y resolución, es necesario hacer muchas modificaciones por cada imagen de la aplicación y de ahí el uso de este programa. Ir con el GIMP editando imágenes una a una es costoso de tiempo y muy poco

eficiente, así pues, ha sido una alternativa bastante útil al GIMP porque se hace la misma operación a la vez.

Adobe Photoshop CS6

Habitualmente, cuando se dispone de recursos gráficos que se han de enseñar y tienen que lucir bien, se necesita de recursos potentes, como el Photoshop. Por ejemplo, se ha utilizado para el retoque de los fondos y de algunas fotos incluidas en la aplicación. A continuación el logo de Adobe Photoshop CS6 en la *Figura 3.8.*

Figura 3.8 .- Logo de Adobe Photoshop CS6

La última versión liberada el 7 de mayo del 2012 ha llevado a la vanguardia del diseño gráfico y 3D esta herramienta. Tiene opciones muy interesantes. Puede saber más desde la página web referenciada en la bibliografía [40].

Cubase5 – plugin waves

Cubase es una serie de aplicaciones informáticas para editar audio digital, creadas por la firma alemana *Steinberg* en 1989. Este programa es usado por reputadas bandas musicales para la edición de sus canciones, por lo que es un referente. En nuestro caso, se ha utilizado este *software* para editar el audio de cada punto de la ruta de *Itinerarios: Jardines del Buen Retiro*. Más detalles acerca del porqué elegir esta herramienta y no otras en el enlace de la bibliografía [55].

Edición de documentos, diagramas y diapositivas

Cuando se documenta un proyecto, es necesario realizarlo con las herramientas adecuadas, así como con las pertinentes ilustraciones para favorecer la comprensión de lo que se está escribiendo. De esta manera, para generar documentación escrita, se ha utilizado Word, LyX y Adobe Acrobat X; para la creación de dibujos y diagramas, se usa One Note y el programa DIA y, por último, Powerpoint ha permitido la realización de transparencias basadas en diapositivas.

LyX

LyX es un procesador de documentos que combina la fuerza y flexibilidad de *TeX/LaTeX* con la utilidad de tener una interfaz gráfica para la edición de estos ficheros. Se basa en convertir a *LaTeX* el texto introducido para posteriormente hacer con ello todo lo que se pueda imaginar. Es una herramienta poderosa que ha sido utilizada para la redacción de las actas de reunión. Para descargarlo, se necesita tener instalado MyKTeX, ya que sin ello no funcionará. A continuación, los enlaces a lo referenciado anteriormente:

- MyKTeX [47]
- LyX [48]

Microsoft Office: Microsoft Word 2007

Word es un software desarrollado por *Microsoft* dedicado al procesamiento de textos, de hecho, ha llegado a ser el más allegado a todos los hogares. Aunque ya ha sido sustituida en la mayoría de artículos científicos por LaTeX, nuestra documentación no requería de fórmulas complejas a redactar. Así pues, la mayoría de esta memoria está redactada con *Word*, debido a su facilidad y familiaridad con la que se ha tenido que trabajar día tras día a lo largo de los años. Si bien *Word* ha evolucionado y actualizado su fachada de múltiples maneras, también ha velado por la eficiencia en el trabajo del usuario. Siempre ha estado ahí para hacer la vida más fácil. Puede exportar a múltiples formatos como .odt, .html o .pdf

Microsoft Office: Microsoft PowerPoint 2007

PowerPoint es un programa de presentación ampliamente utilizados en distintos campos como la enseñanza, negocios, etc... Es un programa diseñado para hacer presentaciones con texto en forma de esquema, así como la inclusión de animaciones de texto o imágenes añadidas desde el propio ordenador. A lo largo de la carrera se ha usado este software con muy buen resultado, por lo que no se vio ningún requisito nuevo por lo que cambiar para utilizarlo en la presentación final de Sistemas Informáticos.

Microsoft Office: Microsoft One Note 2007

One Note es un producto de *software* que sirve para facilitar la toma de notas, la recopilación de información y la colaboración multiusuario. Permite muchas opciones, como el uso compartido de archivos o WebDAV. El uso de este software nació de la necesidad de explicar entre el grupo

que conforma el proyecto algunas ideas y desarrollos del proyecto a distancia. Sirve para responder de manera ilustrada a cualquier pregunta sobre diseño que se proponga. No olvidar que esto no es un sustituto al lápiz y al papel, sino una solución ágil al problema de la movilidad.

Dia – Diagram Editor

Dia es una aplicación bajo licencia GNU GPL para la creación de diagramas técnicos. Sus interfaces y características son totalmente parecidas al programa *Visio* de *Windows*. Las características de Dia incluyen la impresión multipágina, la exportación a formatos como EPS, SVG, CGM y PNG, y la habilidad para personalizar las formas creadas por el usuario mediante un simple XML. Dia es útil para dibujar diagramas UML, mapas de redes y diagramas de flujo. A continuación en la *Figura 3.9* se ilustra el logo de Dia – Diagram Editor.

Figura 3.9 .- Logo de Dia - Diagram Editor

Por parte del equipo se ha utilizado Dia para los diagramas de flujo del funcionamiento de la aplicación, así como en la descripción de las tablas y relaciones de las bases de datos. Puede saber más de este programa, así como proceder a su descarga desde el enlace de la bibliografía [50].

Adobe Acrobat X

Para la edición de la presente memoria ha sido necesario utilizar este programa debido a la fusión de contenidos de las actas de reunión. También se ha usado para la visualización de pdf (*Portable Document Format*) de este y más ficheros que se nos han facilitado a lo largo del proyecto. Puede encontrar más información en la bibliografía [49].

Herramientas de sistemas informáticos

En cualquier proyecto que se lleva a cabo, es necesario disponer de servicios levantados en algún ordenador o de la transferencia de ficheros de una máquina a otra, entre otros. En *Itinerarios: Jardines del Buen Retiro*, se ha utilizado un **servidor Apache** para la obtención de recursos multimedia del servidor, un **servicio SSH** para el acceso al mismo, una colección de **servidores de *hosting* gratuito** desde los cuales se descargan recursos, un cliente FTP y SFTP para la transferencia de ficheros con **FileZilla** y el almacenamiento de ficheros pesados en la nube con

Dropbox. A continuación se explica con mayor detalle cada una de las herramientas que se han usado.

Servidor HTTP Apache (2.0)

Apache es un servidor web HTTP (*Hypertext Transfer Protocol*) de código abierto para plataformas UNIX, Microsoft Windows, Macintosh, entre otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual. Es el servidor HTTP más usado desde 1996, y la mayoría de vulnerabilidades de seguridad descubiertas tan sólo se pueden explotar por usuarios locales y no remotamente. Su arquitectura es muy modular y es fácilmente configurable.

Con la necesidad de levantar un servidor web externo y dedicado a este proyecto, la fiabilidad aportada por este servicio ha sido enorme. Hasta la fecha, nunca se ha caído, ni ha oportunado alguna mala circunstancia. Más información en la bibliografía [21].

Servicio SSH

De sus siglas *Secure Shell* es el nombre de un protocolo y del programa que lo implementa. Su utilidad es la de soportar el acceso a máquinas remotas a través de una red. Permite manejar por completo la computadora mediante el intérprete de comandos. Este servidor ha sido útil para el *testing* de la funcionalidad del servidor dedicado, así como para la parada y arranque del apache cuando se ha necesitado.

Servidores web gratuitos

Un servidor web gratuito es un servicio que permite a los usuarios publicar sitios web de manera gratuita. Con la necesidad de alojar audio e imágenes en algún servidor web. Para hacer que la aplicación no ocupase demasiado al ser descargada, se intentó usar algún recurso del ayuntamiento de Madrid con el que se colabora. Ante el desconocimiento a mediados de febrero de algún *host* en el que alojar los datos, se decidió buscar *hosting* gratuito en el que registrarse para replicar los recursos multimedia que descarga la aplicación. Se han usado los siguientes servidores, a los cuáles se les debe agradecimientos por haber servido de puente entre lo que se quería y lo que se necesitaba. Algunos *hostings* gratuitos son los siguientes:

- Site11.com [26]
- Vpla.net [27]
- Zz.mu [28]

- 260mb.org [29]

Ante la posibilidad de desconexión o inhabilitación de alguno de nuestros servidores registrados, se decidió levantar un servicio *Apache2* en un ordenador de los ponentes como método de emergencia en caso de que todos los servidores anteriores fallasen.

Filezilla

FileZilla es un cliente FTP (*File Transport Protocol*) que soporta SFTP (Secure File *Transport Protocol*) y es muy fácil de usar. Es un proyecto de *open source* distribuído bajo los términos de la licencia GNU GPL. Puede descargar este software desde la página web de la bibliografía [22].

Se hace patente la necesidad de transferir recursos multimedia de una máquina a otra. Con este software nuestras necesidades se han visto suplidas de una manera muy fácil y, sobretodo, muy recomendable. Con el servicio SSH levantado en el servidor, se ha podido conectar por SFTP a nuestro servidor externo, lo cual ha ahorrado levantar otros servicios, como el FTP.

Dropbox

Dropbox es un servicio de alojamiento de archivos de diferentes plataformas en la nube. El servicio permite a los usuarios almacenar y sincronizar archivos en línea entre ordenadores y otros usuarios. A continuación en la *Figura 3.10* se ilustra el logo de Dropbox.

Figura 3.10 .- Logo de Dropbox

Ha sido clave para la compartición de archivos de gran envergadura en cuanto a tamaño se refiere. Los archivos de más de 10 MB como algunos videos, han sido compartidos por este sistema, debido a que supera los límites establecidos de archivos adjuntos en los correos electrónicos. Puede obtener una cuenta y empezar a compartir sus ficheros con *Dropbox* [23].

Herramientas para la comunicación y la supervivencia del equipo

En todo proyecto llevado a cabo, es muy importante la comunicación entre los compañeros que desarrollan de manera conjunta. Además, no sólo es importante desde el interior del proyecto,

sino que también la comunicación es vital con los agentes externos como los clientes o el jefe del proyecto. Por ello, el equipo se ha visto influenciado por las herramientas de colaboración que ofrece *Google* como el **gestor de correo Gmail** y el acceso a **Google Docs** desde dentro de Google Drive. Además, si se necesita ayuda urgente en algún tema, se tiene habilitado un grupo de **WhatsApp** dedicado enteramente al proyecto. Para la supervivencia al hacer acopio de documentación de APIs de librerías escritas en inglés, se ha utilizado el traductor de definiciones de **WordReference**.

Gmail

Gmail, también llamado Google Mail, es un servicio de correo electrónico con posibilidades POP3 e IMAP gratuito proporcionado por la empresa estadounidense Google. Aparte del servicio de correo, también deja la posibilidad de usar Google Calendar, Google Docs (ahora integrado con Google Drive), Google Talk y el antiguo Google Buzz (ya cerrado). A continuación se ilustra el logo de Gmail en la *Figura 3.11*.

Figura 3.11 .- Logo de Gmail

Gmail ha sido la vía de comunicación del proyecto, por diferentes aspectos:

- permite la comunicación entre los miembros del proyecto, así como con miembros de otros proyectos, las codirectoras de los mismos y personal del ayuntamiento y del Retiro
- permite la creación de grupos en GoogleGroups. Un grupo es un correo asociado a unas cuentas Gmail vinculadas al mismo, por lo que se tiene un correo que proporcionar en el caso de que se interesen por nuestro proyecto.
- permite la interacción entre los componentes del grupo de proyecto de manera activa gracias al chat de GTalk integrado con Gmail
- permite que se recuerden fechas importantes gracias a Google Calendar, así como compartir el calendario entre los propios miembros para tener la agenda bien cubierta

Puede obtener una cuenta de correo para acceder a los servicios previamente mencionados (y alguno más) desde el siguiente enlace de la bibliografía [25].

Google Docs: hojas de cálculo y de texto

Google Docs merece un apartado por sí solo debido a la utilidad y la importancia que ha tenido en la realización del proyecto. Cuando se trabaja en equipo es necesario trabajar en paralelo sobre un mismo documento o ver en tiempo real los cambios para poder discutirlos en el momento. Estas herramientas son muy útiles porque se tiene un canal de chat dentro del propio documento por donde se pueden debatir estas opciones. Además, aunque no suelen ser documentación entregable, siempre sirve para ponerse de acuerdo sobre el reparto de tareas.

WhatsApp

WhatsApp es una aplicación de mensajería multimedia que permite enviar y recibir mensajes mediante internet. De esta manera, se sustituyen los servicios tradicionales de mensajes cortos o sistemas de mensajería multimedia debido al coste que ello suponía. Se pueden crear grupos y enviar en el mismo un número ilimitado de mensajes, así como de imágenes, vídeos y de mensajes de audio. Está disponible para *Windows Phone*, *iOS*, *Blackberry*, *Android* y algunos *Symbian*. A continuación, la *Figura 3.12* ilustra el logo de WhatsApp.

WhatsApp

3.12 .- Logo de WhatsApp

Esta aplicación ha sido vital para el flujo de la comunicación en el seno de los desarrolladores. Además, ha servido para arreglar dudas y para comunicar algunos errores. Puede descargarse esta útil aplicación móvil desde el mercado de aplicaciones del dispositivo, véase *Google Play* para *Android* y *AppleMarket* para *iOS*. Más información en la bibliografía [53].

WordReference

WordReference es una página web que hace de traductor de palabras de inglés a español, o a casi cualquier idioma. La utilidad fundamental no es la traducción, si no la definición de la palabra en el propio inglés. A la hora de mediar con APIs la documentación suele estar en inglés, así que se

ha encontrado el baluarte definitivo al que acudir cuando haya alguna duda. Puede comprobarlo usted mismo a través de la página web que se proporciona en la bibliografía [51].

Recursos físicos utilizados

En este apartado hablaremos de todas las partes tangibles del sistema informático utilizado para llevar a cabo el proyecto. Por mayor comodidad a la hora de programar, se ha decidido hacer muy poco uso de los laboratorios habilitados para prácticas que están disponibles en la Facultad de Informática. De esta manera, se ha ganado autonomía al no depender de un tercero para el desarrollo del proyecto. Estructuraremos este apartado en tres: equipos de desarrollo, operativa móvil y *táblets*.

Equipos de desarrollo

Debido a su poco tamaño, ha sido posible llevarlos a reuniones y usarlos *in situ*. Se han utilizado los portátiles que vienen a continuación junto con sus características más importantes.

- Portátil de Alberto: **Samsung R540 [30]**
  - **Procesador:** *Intel Core* i5 COU M480 2.67 GHz
  - **Sistema operativo:** *Windows 7 Professional* – 64 bits
  - **Memoria RAM:** 4 GB de RAM
  - **Gráfica externa:** *ATI Mobility Radeon HD* 5400
  - **Espacio en disco:** 500 GB de disco duro
- Portátil de Miguel: **Samsung RV511 [31]**
  - **Procesador:** *Intel Core* i3 M380 2,53 GHz
  - **Sistema operativo:** Windows 7 Home Premium – 64 bits
  - **Memoria RAM:** 4GB de RAM
  - **Gráfica externa:** *NVIDIA GeForce* 315M
  - **Espacio en disco:** 320 GB
- Portátil de Víctor: **Acer Aspire 5920G [32]**
  - **Procesador:** *Intel Core* 2 Duo T7500 2.2 GHz
  - **Sistema operativo:** *Windows 7 Home Premium* – 64 *bits*
  - **Memoria RAM:** 4 GB de RAM
  - **Espacio en disco:** 160 GB de disco duro

Además, no sólo han sido usados estos dispositivos portátiles, sino que también se han utilizado otros ordenadores de sobremesa, como los siguientes:

- Servidor externo dedicado de Alberto:
    - **Procesador:** *Pentium Dual-Core* 2,5 GHz
    - **Sistema operativo:** Debian 6 – Sin interfaz gráfica
    - **Memoria RAM:** 4 GB de RAM
    - **Espacio en disco:** 350 GB de disco duro

Dispositivos móviles

A principios del 2012, cuando se estaba gestando este proyecto, los desarrolladores no disponían de ningún dispositivo *Android* en el que probar lo que futuramente creáramos, por lo que se decidió adquirir dispositivos nuevos para integrarnos en el mundo de *Android*. Estos dispositivos tenían que tener o disponer de los siguientes aspectos que eran a priori necesarios para ejecutar la futura aplicación:

- **Dispositivos con *Android* 2.3.3 (*Gingerbread*) o superior:** con esto se ha impuesto unos requisitos *hardware* (y económicos) antes del comienzo del proyecto, ya que la mayoría del equipo tenía dispositivos de otra generación o más bien obsoletos.
- **Dispositivos con GPS:** para poder al menos usar *Google Maps* y poder tener alguna referencia al programar.
- **Tarifa de datos:** al principio no se sabía realmente cómo iba a ir todo, si con recursos internos de la aplicación o con descarga de *Internet*, por lo que también era necesario tener esta opción. Esto supone gastos mensuales por parte de los desarrolladores para la ejecución de las pruebas de campo.

Los dispositivos de los creadores de *Itinerarios: Jardines del Buen Retiro* vienen a continuación.

Samsung Galaxy S II

Elegido por Alberto, y con la intención de que fuera una inversión segura para una década, se contó con él en lugar de con el *Samsung Galaxy S III* por razones prácticas y económicas. Sin más preámbulos, satisfizo desde el primer momento todas las necesidades de su comprador [33]. Se puede ver más adelante en la *Figura 3.13*.

Sony Xperia U

Elegido por Miguel como dispositivo *Android*. El Sony Xperia U es el miembro menor de los nuevos *smartphons Android* de Sony. Este dispositivo se basa en las múltiples capacidades de personalización, desde accesorios de colores hasta perfiles de software [34]. Se puede ver una ilustración en la *Figura 3.14* que está más adelante.

Sony Ericsson Xperia neo V

Por último, falta por comentar el dispositivo de Víctor, el *Sony Ericsson Xperia neo V* [35]. Este dispositivo es un *smartphone* que mantiene las formas del *Xperia* neo original, salvo que corre sobre la versión 2.3.3 de *Android Gingerbread*. A continuación se puede ver en la *Figura 3.15.*

Figura 3.13

Figura 3.14

Figura 3.15

Más dispositivos Android: Tablets

No sólo se ha probado la aplicación en dispositivos móviles como los *smartphons* comentados hace un momento, si no que, además, se ha podido disponer de las *tablet* que explicaremos a continuación.

ICOO iCou8GT (Táblet)

Esta tableta traída desde China y adquirida por Alberto, no tiene nada que envidiar al resto de las actuales [36]. La innovación de la *Tablet PC* con el nuevo sistema operativo *Android* 4.1 supone una experiencia moderna e inmediata, fácil de usar y más rápido que nunca, ya que la interfaz gráfica ha sido mejorada. En la Figura 3.16 se ilustra esta tableta *Android*.

Figura 3.16 .- ICOO iCou8GT (Táblet)

Capítulo 4 - Especificación de Casos de Uso

En este capítulo se describe con gran nivel de detalle la secuencia de operaciones, interacciones, actores, etc. de cada uno de los requisitos funcionales especificados con anterioridad. El esquema de dependencias entre los distintos casos de uso existentes y los agentes externos se muestra en la *Figura 4.1*, mientras que la relación (*Figura 4.2*) entre los casos de uso se podrá observar con más detalle en las tablas posteriores.

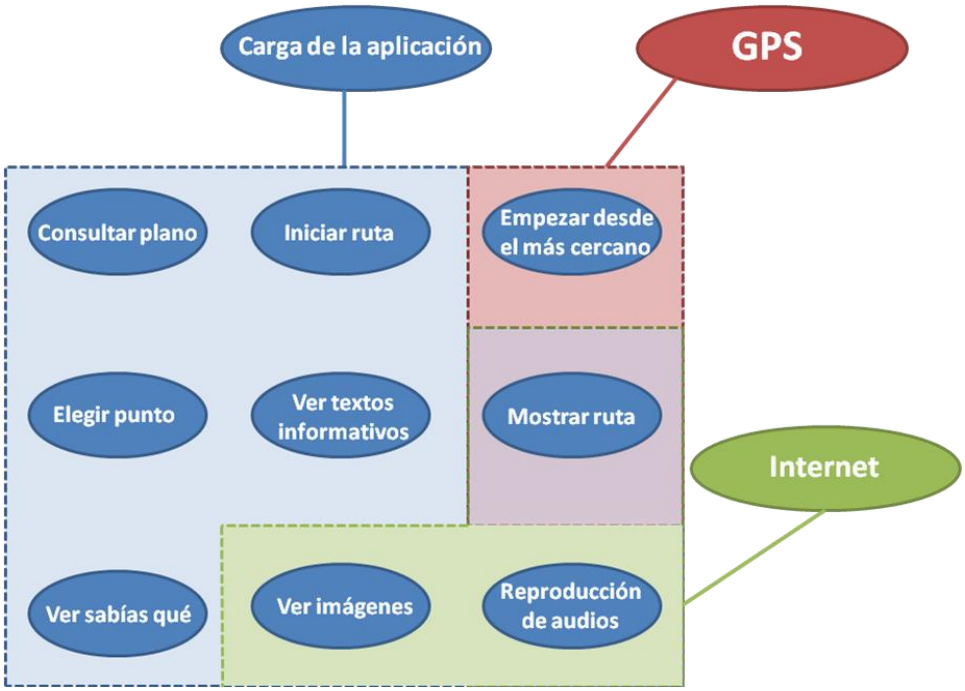


Figura 4.1. Esquema de dependencias entre casos de uso y agentes externos

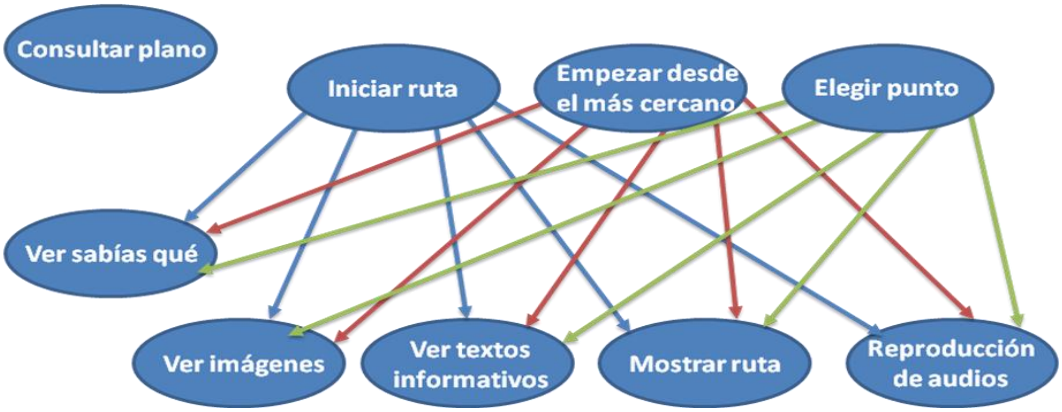


Figura 4.1 .-Relación entre los casos de uso

RF-1	Nombre: Consultar plano	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá ver el plano a pantalla completa pudiendo interactuar con el zoom	
Objetivo	Consulta del plano	
Precondición	-	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	El usuario pulsa sobre el mapa en miniatura en la pantalla de iniciar ruta	
Operaciones permitid	**Paso**	**Acción**
	1	Si se realiza la doble pulsación sobre el plano, éste se ampliará o alejará del todo dependiendo del estado actual del zoom
	2	Haciendo el gesto de un pellizco el plano se alejará progresivamente
	3	Haciendo el gesto de un pellizco a la inversa, el plano se acercará progresivamente
	4	Si el plano se ha ampliado algo, deslizando el dedo, el usuario podrá desplazarse por él
Pos condición	El plano se visualiza a pantalla completa con el zoom adecuado en cada momento	
Excepciones	-	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	

Tabla 4.1 .- Caso de uso: consultar plano

RF-2	Nombre: Iniciar ruta	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá iniciar la ruta desde el primer punto de interés, según la numeración dada por el cliente	
Objetivo	Empezar la ruta desde el punto de inicio	
Precondición	-	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	El usuario pulsa sobre el botón punto de inicio en la pantalla de iniciar ruta	
Operaciones permitid	**Paso**	**Acción**
	1	Pulsar el botón de información (RF-7)
	2	Pulsar el botón de Audio (RF-6)
	3	Pulsar el botón de imágenes (RF-5)
	4	Pulsar el botón de sabías qué (RF-8)
	5	Pulsar botón cómo llegar (RF-9)
	6	Pulsar botón siguiente (avanzar al siguiente punto de la ruta)
	7	Pulsar botón anterior (retroceder al punto anterior de la ruta)
Pos condición	Cumplir el requisito adjunto a cada una de las posibles opciones	
Excepciones	-	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	

Tabla 4.2 .- Caso de uso: iniciar ruta

RF-3	Nombre: Elegir punto	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá elegir, de entre los disponibles el punto por el que prefiera empezar el recorrido	
Objetivo	Empezar la ruta desde el punto elegido por el usuario	
Precondición	-	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	El usuario pulsa sobre el botón elegir punto en la pantalla de iniciar ruta	
Operaciones permitid	**Paso**	**Acción**
	1	Las posibilidades se mostrarán en una lista y valdrá con pulsar sobre una de ellas
Pos condición	Mostrar la pantalla del punto seleccionado	
Excepciones	-	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	
Comentarios	-	

Tabla 4.3 .- Caso de uso: elegir punto

RF-4	Nombre: Empezar desde el más cercano	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá empezar el recorrido desde el punto más cercano a su posición actual	
Objetivo	Empezar la ruta desde el punto más cercano al usuario	
Precondición	GPS habilitado	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil, GPS	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	El usuario pulsa sobre el botón "Punto más próximo" en la pantalla de iniciar ruta	
Operaciones permitid	**Paso**	**Acción**
	1	Pulsar el botón de información (RF-7)
	2	Pulsar el botón de Audio (RF-6)
	3	Pulsar el botón de imágenes (RF-5)
	4	Pulsar el botón de sabías qué (RF-8)
	5	Pulsar botón cómo llegar (RF-9)
	6	Pulsar botón siguiente (avanzar al siguiente punto de la ruta)
	7	Pulsar botón anterior (retroceder al punto anterior de la ruta)
Pos condición	Cumplir el requisito adjunto a cada una de las posibles opciones del punto que el GPS ha calculado como el más cercano	
Excepciones	GPS deshabilitado. El usuario será informado si esto ocurre	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	
Comentarios	Posible demora debida al GPS	

Tabla 4.4 .- Caso de uso: empezar por más cercano

RF-5	Nombre: Ver imágenes	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá navegar a través de las imágenes de cada punto de interés	
Objetivo	Ver imágenes de cada punto	
Precondición	Conexión a internet ó imágenes almacenadas en la tarjeta SD.	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil, Base de datos o memoria SD	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	El usuario pulsa sobre el botón imágenes en cualquier pantalla de punto de interés	
Operaciones permitidas	**Paso**	**Acción**
	1	Las imágenes se dispondrán en una lista con su pie de foto correspondiente y se podrá pulsar cada una de ellas, pasando al paso 2
	2	Todas las posibilidades recogidas en el RF-1 están contempladas en cada imagen
	3	Se podrá apaisar el teléfono y con ello la imagen girará
	4	Podrá deslizarse el dedo para pasar a la siguiente o a la anterior imagen
Pos condición	Las imágenes se mostrarán y todo lo contemplado arriba funcionará correctamente	
Excepciones	Imágenes no encontradas por fallo en la conexión con el servidor. Se le informará al usuario de que las imágenes no se han podido cargar	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	
Comentarios	Posible demora en la carga de imágenes si la conexión con el servidor no es buena y aún no se han descargado	

Tabla 4.5 .- Caso de uso: ver imágenes

RF-6	Nombre: Reproducción de audios	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá escuchar locuciones de cada uno de los puntos así como una pequeña introducción a la ruta	
Objetivo	Escuchar explicaciones acerca de la ruta.	
Precondición	Conexión a internet	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil, base de datos	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	O bien en la pantalla principal, pulsando el botón Sobre la ruta, o bien pulsando el botón audio en cualquiera de los puntos	
Operaciones permitid	**Paso**	**Acción**
	1	Se podrá pulsar el botón de pausa para parar el audio en cualquier momento
	2	Si el audio está pausado, se podrá pulsar el botón de *play* para reanudar la reproducción
	3	Podrá pulsarse el botón reiniciar para volver a escuchar el audio desde el principio
Pos condición	Se reproducirá el audio permitiendo al usuario las operaciones descritas	
Excepciones	Audio no encontrado por fallo en la conexión. El usuario será informado	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	
Comentarios	Posibles cortes en el audio a causa de una mala conexión	

Tabla 4.6 .- Caso de uso: reproducción de audios

RF-7	Nombre: Ver textos informativos	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá leer textos de explicación de cada punto de la ruta	
Objetivo	Leer información detallada en modo texto	
Precondición	-	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	El usuario pulsa sobre el botón información en cualquiera de los puntos	
Operaciones permitid	**Paso**	**Acción**
	1	Se podrá deslizar el dedo a modo de scroll para ver todo el texto
Pos condición	Se mostrará el texto	
Excepciones	-	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	
Comentarios	-	

Tabla 4.7 .- Caso de uso: ver textos informativos

RF-8	Nombre: Ver sabías qué	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá leer textos acerca de curiosidades y árboles característicos del punto de la ruta	
Objetivo	Ampliar información con detalles curiosos en formato de texto	
Precondición	-	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	El usuario pulsa sobre el botón sabías qué en cualquiera de los puntos	
Operaciones permitidas	**Paso**	**Acción**
	1	Se podrá deslizar el dedo a modo de scroll para ver todo el texto
Pos condición	Se mostrará el texto	
Excepciones	-	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	
Comentarios	-	

Tabla 4.8 .- Caso de uso: ver sabías qué

RF-9	Nombre: Mostrar ruta	
Iteración	Versión: 1 Última modificación: 25 de Mayo 2013	
Descripción	El usuario podrá averiguar cómo llegar a un punto y se dibujará la ruta en un mapa	
Objetivo	Visualizar la ruta cuyo destino es el punto que quiere verse	
Precondición	Hay conexión a internet y el GPS está habilitado	
Actor Principal	Usuario	
Canal al actor principal	Aplicación Itinerarios a través de dispositivo móvil, GPS, Google maps	
Actores secundarios	-	
Canal a actores secundarios	-	
Disparador	Pulsar el botón Cómo llegar en la pantalla de cualquier punto de interés en la ruta	
Operaciones permitid	**Paso**	**Acción**
	1	Se podrá pulsar en las marcas del mapa mostrándose con un cuadro de texto qué representa cada una de ellas
	2	Se podrá pulsar en el icono de la izquierda en el panel de abajo (icono de un hombre andando en color verde) para saber dónde se encuentra el usuario actualmente
	3	Se podrá pulsar en el icono de la derecha en el panel de abajo (icono de un hombre parado con árboles detrás) para saber dónde se encuentra el punto de destino
	4	Todas las operaciones que nos permite la API de google maps podrán efectuarse (rotar, mover, ampliar, alejar, etc.)
Pos condición	Se mostrará un mapa con la ruta dibujada así como los puntos de destino y posición actual	
Excepciones	GPS deshabilitado. El usuario será informado. No se tiene conexión a internet (el mapa no se mostrará)	
Frecuencia de uso	Media	
Importancia	Alta	
Estabilidad	Alta	
Comentarios	Posible demora en la obtención de las coordenadas por GPS	

Tabla 4.9 .- Caso de uso: mostrar ruta

Capítulo 5 - Diseño de *Itinerarios: Jardines del Buen Retiro*

Este capítulo está dedicado a todo lo relativo al diseño de la aplicación. En primer lugar se explica el diseño de la arquitectura del software de la aplicación tanto a nivel de interfaz, como de base de datos y de clases funcionales. En segundo lugar se describe de forma detallada el diseño de la base de datos que alberga los contenidos de la aplicación. Por último se muestra un resumen de algunos de los cambios y problemas surgidos en el desarrollo del proyecto que han obligado a diversos cambios en el diseño de la aplicación.

Diseño de la Arquitectura del Software

En esta sección se trata la división hecha por el equipo de desarrollo de Itinerarios para la implementación de la aplicación. Para empezar hay que tener en cuenta lo que se pretende con esta aplicación. Según el encargo del cliente y las sucesivas reuniones con el mismo, el proyecto fue tomando forma, dando como resultado una aplicación de carácter turístico para la difusión de la cultura que contempla con gran detalle la ruta a través de los estilos de jardinería del parque del Buen Retiro. Para hacer esta ruta modulable y accesible se tuvo muy en cuenta el diseño de la interfaz para que fuese sencilla, visual e intuitiva.

Por otro lado hay que tener en cuenta el volumen de datos necesarios para la aplicación. Se precisa de usar una base de datos que permite que la aplicación no ocupe demasiado espacio.

Por lo que se acaba de explicar se ha planificado una división del proyecto en tres módulos de clases claramente diferenciados:

1. **Clases de interfaz:** tienen un código xml asociado para definir la estructura visual, además de métodos que darán funcionalidad a los distintos componentes de la misma.
2. **Clases de la base de datos:** hacen consultas en la base de datos en función de las peticiones que se realicen por el usuario a través de la interfaz.
3. **Clases funcionales:** Se encargan de dotar de funcionalidad a la aplicación en general.

La *Figura 5.1* refleja un esquema general de la arquitectura del software de la aplicación en el cual se observa el proceso de la descarga de la aplicación y las principales interacciones entre las clases descritas con anterioridad.

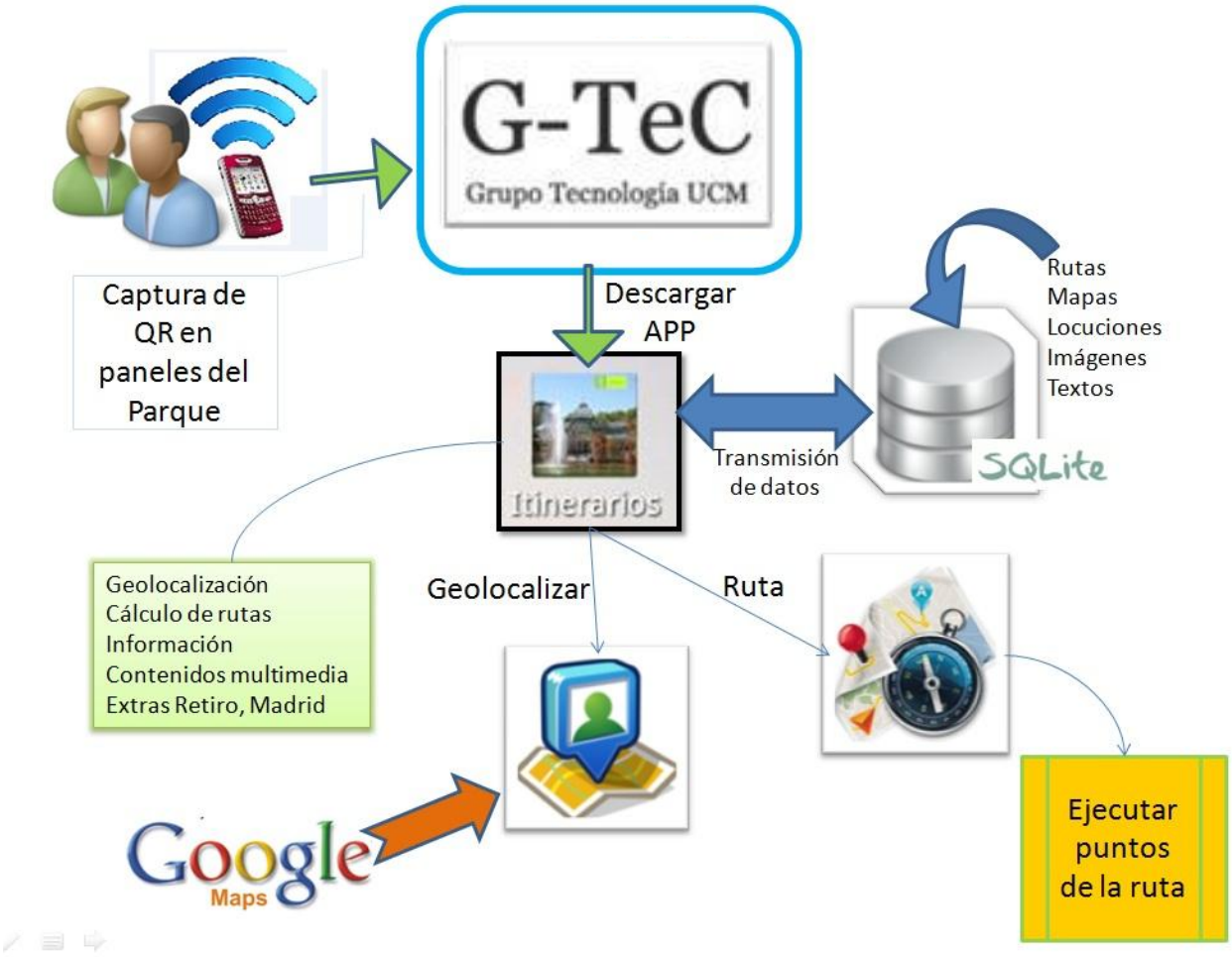


Figura 5.1 .- Esquema general de la arquitectura software de la aplicación.

En la *Figura 5.2* muestra un diagrama la interrelación de estos tres módulos:

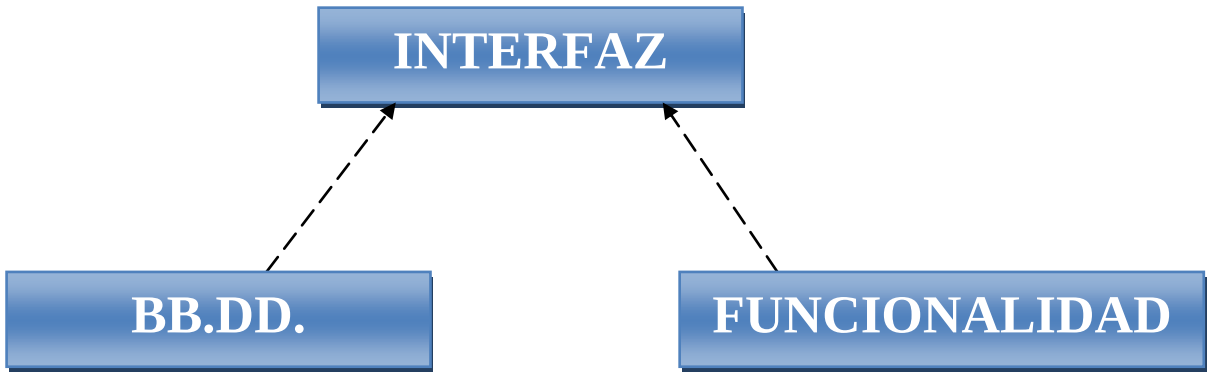


Figura 5.2 .- Diagrama de interrelación entre los módulos de las clases.

A continuación se detallan las clases que conforman cada uno de los módulos:

Diseño de la interfaz

Este conjunto de clases están conformadas por código en xml para determinar dónde se ubicará cada componente y código java que dotarán de funcionalidad a cada uno de ellos.

Todas las clases de interfaz tienen los métodos onCreate(Bundle) que inicializará todos los componentes y se conectará a la base de datos y onBackPressed(View) que definirá el botón de retroceso del móvil.

Las clases diseñadas por el equipo para esta aplicación son:

- **MainActivity**. Muestra la pantalla inicial de la aplicación. En la *Figura 5.3* se muestra un diagrama UML de la clase y la visualización de la *activity* en pantalla.

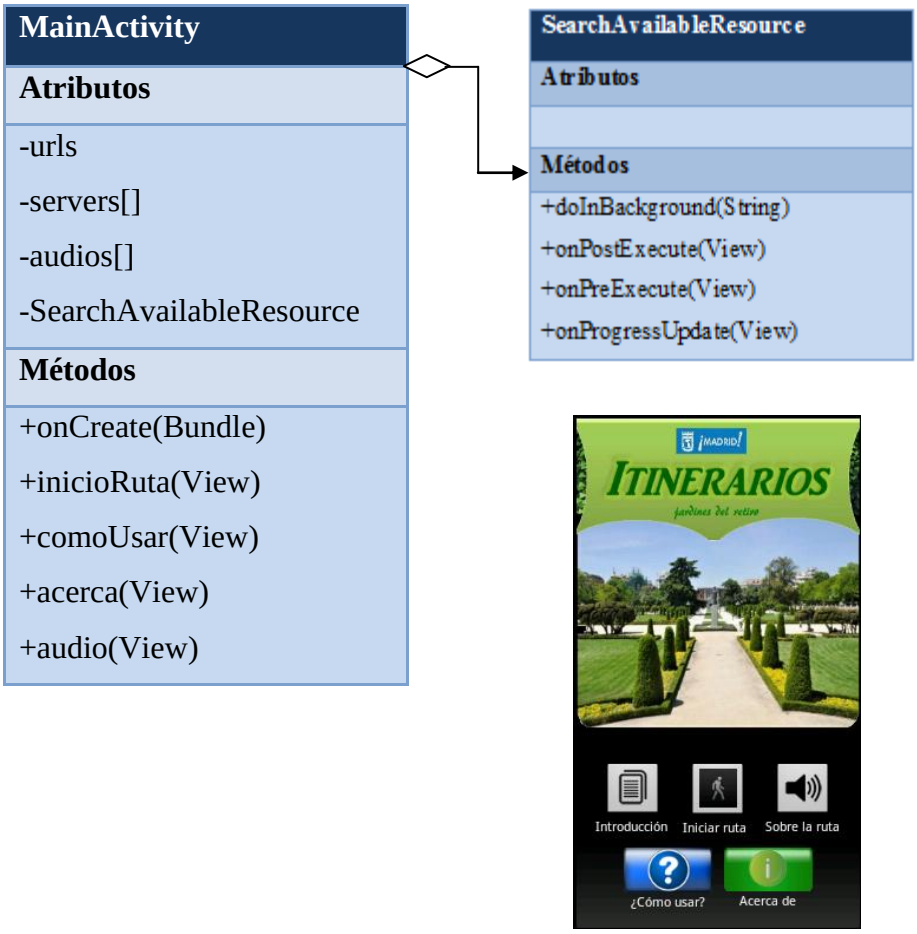


Figura 5.3 .- UML y captura de MainActivity

- **InformacionActivity**. Muestra los textos informativos de la aplicación, tanto de introducción al parque como de información concreta de cada punto del recorrido. En la *Figura 5.4* se observa un diagrama UML de la clase y la captura de la *activity* al ser ejecutada.

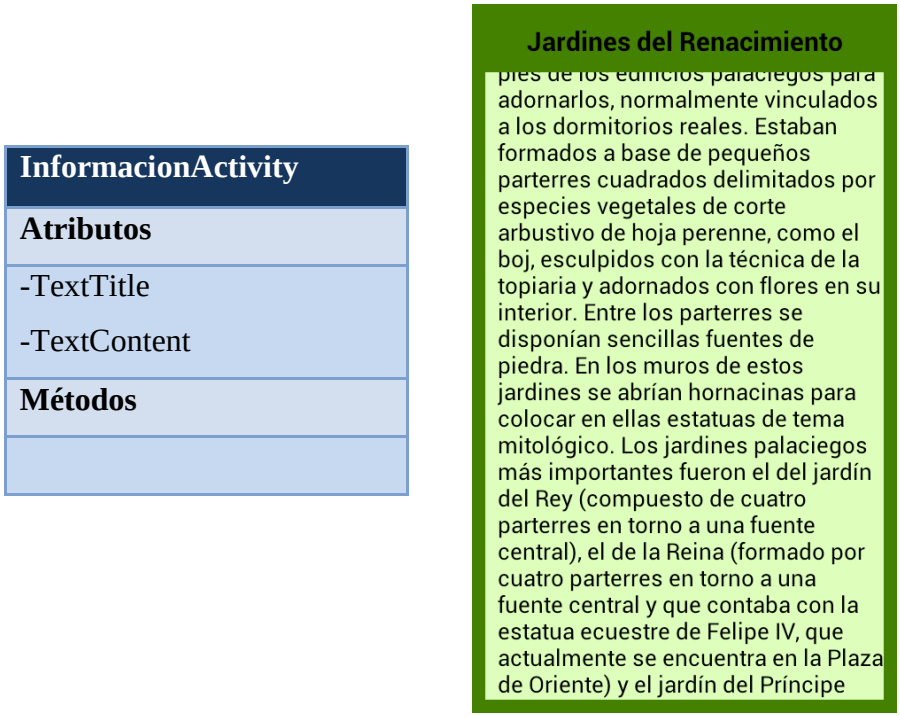


Figura 5.4 .- UML y captura de InformacionActivity

- **AudioActivity.** Muestra el reproductor de audio informativos de la aplicación, tanto de introducción al parque como la locución concreta de cada punto del recorrido. En la *Figura 5.5* se observa un diagrama UML de la clase y la captura de la *activity* al ser ejecutada con su correspondiente barra de progreso de audio.

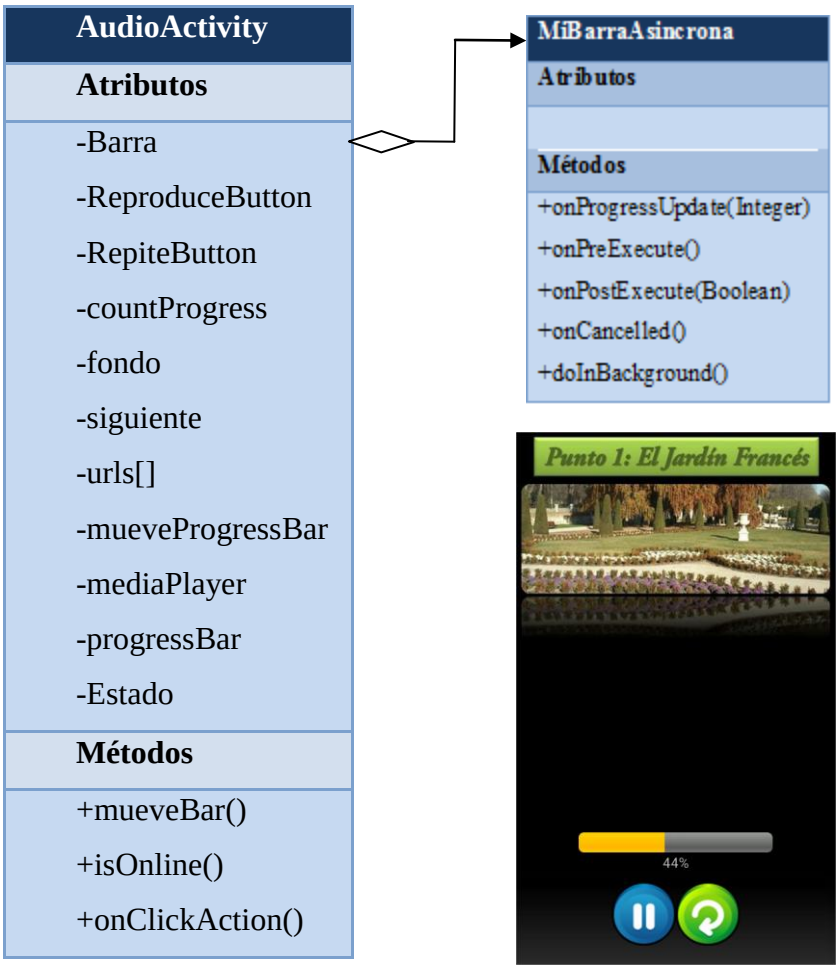


Figura 5.5.- UML y captura de AudioActivity

- **ComoUsarActivity:** Muestra un manual de uso de la aplicación. En la *Figura 5.6* se observa un diagrama UML de la clase y la captura de la activity al ser ejecutada.

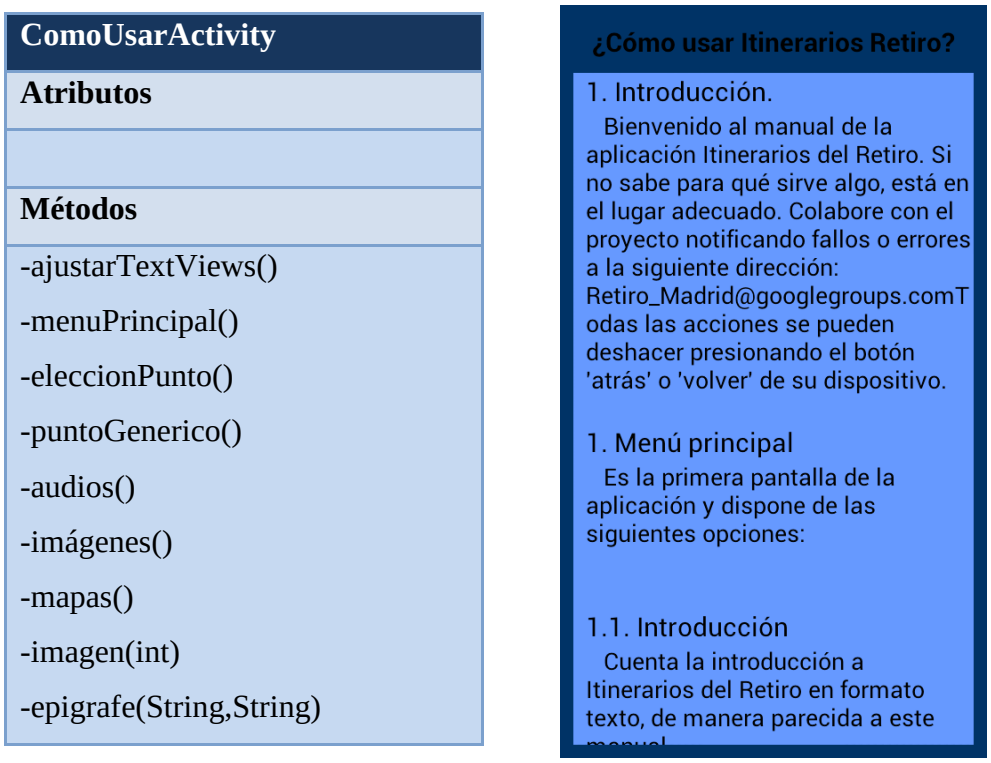


Figura 5.6 .- UML y captura de ComoUsar

- **UCMGTec:** Muestra enlaces de interés y los nombre del equipo de desarrolladores de la aplicación. En la *Figura 5.7* se observa un diagrama UML de la clase y la captura de la *activity*. Si se pulsa sobre las imágenes, éstas abrirán la página en el navegador de la imagen pulsada.

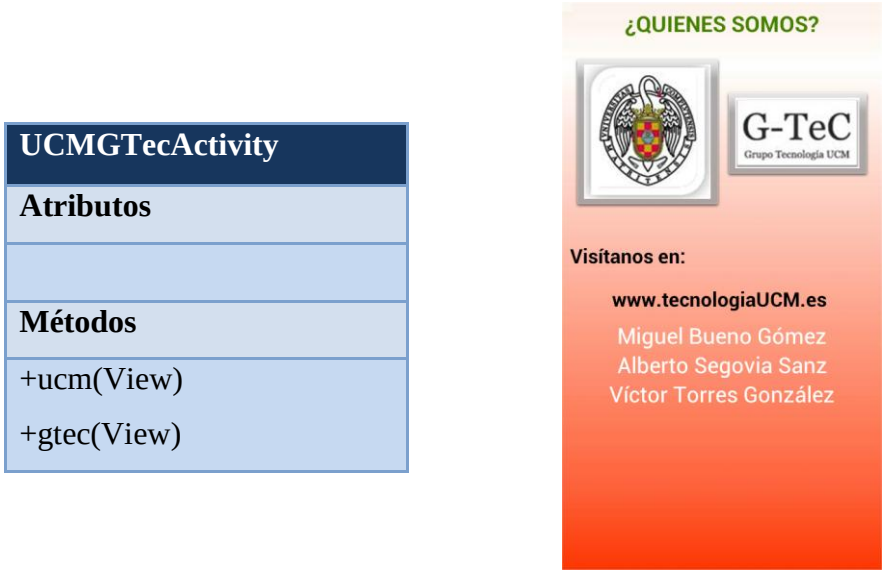


Figura 5.7 .- UML y captura de UCMGTecActivity

- **ElegirInicioActivity.** Muestra la pantalla de inicio de ruta con el mapa del Retiro y la posibilidad e iniciar la ruta desde el principio, geolocalizar al usuario hacia el punto más cercano del recorrido o elegir un punto concreto del itinerario. Si se pulsa sobre el mapa se puede hacer *zoom* sobre el mismo. En la *Figura 5.8* se observa un diagrama *UML* de la clase y la captura de la *activity*.

ElegirInicioActivity
Atributos
-locManager -locListener -latitud -longitud -aLPuntoMapa
Métodos
+puntoCercano(View) +mapa(View) +empezar(View) +elegirLista(View)

Figura 5.8 .- UML y captura de ElegirInicioActivity

- **ListaActivity:** Muestra la elección de un punto concreto de la ruta de entre los siete posibles. Si se pulsa sobre un elemento de la lista iniciará el punto seleccionado. En la *Figura 5.9* se observa un diagrama UML de la clase y la captura de la *activity*.

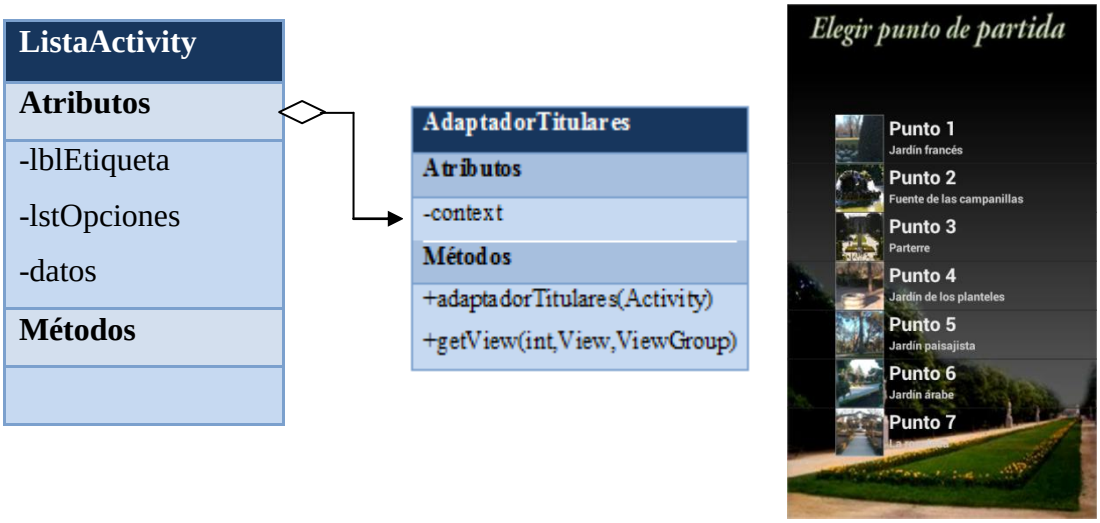


Figura 5.9 .- UML y captura de ListaActivity

- **PlanoRetiroActivity:** Muestra el mapa del parque ocupando toda la pantalla. Se puede realizar zoom sobre dicha imagen. En la *Figura 5.10* se observa un diagrama UML de la clase y la captura de la *activity*.

Figura 5.10 .- UML y captura de PlanoRetiro

- **PuntoGenericoActivity**. **.** Muestra la pantalla funcional de cada uno de los puntos. El fondo y el enlace a la base de datos vendrán dados por el *Bundle* del método *onCreate(Bundle)*. Nos devuelve en qué punto se encuentra el usuario. En la *Figura 5.11* se observa un diagrama UML de la clase y la captura de la *activity*.

PuntoGenericoActivity
Atributos
-fondo -parada -urls[] -audios[] -imágenes[]
Métodos
+siguiente(View) +anterior(View) +Imagenes(View) +audio(View) +irAlMapa(View) +info(View) +sabiasQue(View)

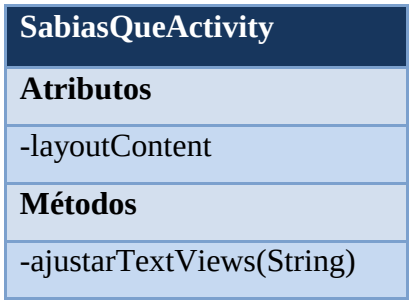
Figura 5.11 .- UML y captura de PuntoGenericoActivity

- **ComoLlegar:** Muestra la pantalla del mapa de *Google Maps* con la ruta dibujada que debe seguir el usuario desde su localización hasta el punto destino que ha elegido. En la *Figura 5.12* se observa un diagrama UML de la clase y la captura de la *activity*.

ComoLlegarActivity
Atributos
-latitud -longitud -latPunto -longPunto -nombre -puntos[]
Métodos
+camino(int) +masCercano(double,double) +camLoc(View) +camDest(View)

Figura 5.12 .- UML y captura de ComoLlegar

- **SabiasQueActivity.** Es similar a informacionActivity. Cambian los colores del diseño y su utilidad ya que contiene textos acerca de curiosidades y árboles interesantes de cada punto. En la *Figura 5.13* se observa un diagrama UML de la clase y la captura de la *activity*.



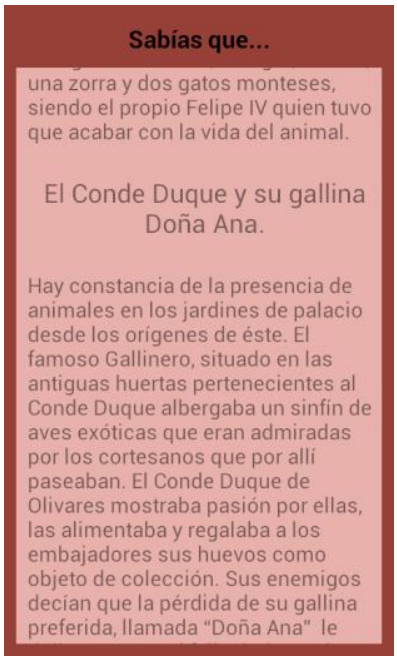


Figura 5.13.- UML y captura de SabiasQueActivity

El resto de clases relacionadas con la interfaz se han omitido de esta sección porque hacen uso de librerías externas y serán explicadas más adelante.

Diseño de la base de datos

El diseño de las clases de la base de datos es objeto de estudio en el siguiente apartado. En éste nos limitaremos a decir los recursos multimedia están alojados en varios servidores por si fallara alguno y que hay clases específicas en el proyecto encargadas de gestionar y hacer las consultas pertinentes a la base de datos.

Todas estas clases se explicarán en el siguiente apartado.

Diseño de las clases funcionales

La mayoría de las clases usadas para completar la funcionalidad de la interfaz han sido descritas anteriormente como clases asociadas de las mismas. En este subapartado describiremos las restantes. Por un lado describiremos la clase Utils, que nos proveerá de un conjunto de métodos muy útiles para dotar de la funcionalidad deseada al sistema. En la *Figura 5.14* se puede ver el diagrama de dicha clase:

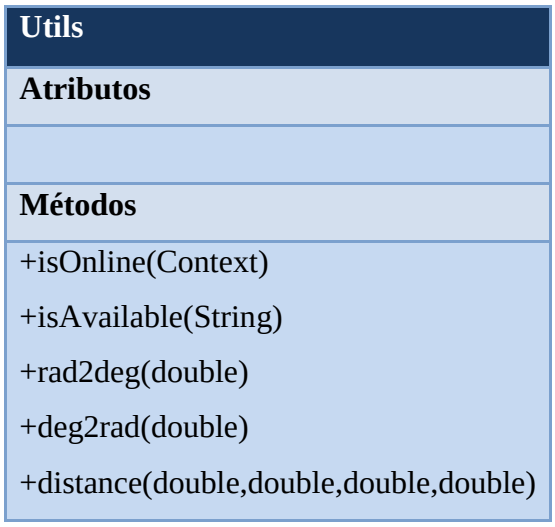


Figura 5.14 .- UML de la clase Utils.

- isOnline, determinará si la aplicación está conectada a internet actualmente.
- isAvailable, nos informará de si el servidor al que se accede está disponible, si está caído o no ofrece el recurso.
- Rad2deg convertirá un ángulo de radianes a grados.
- Deg2rad convertirá de grados a radianes
- Distance determinará la distancia entre dos puntos cuya latitud y longitud vendrán determinados por el primer y segundo parámetro, y el tercero y el cuarto respectivamente.

Por otro lado y para acabar con el apartado de la arquitectura del software, comentamos un paquete separado de la aplicación, pero desarrollado por el mismo equipo y muy relacionado con la parte de geolocalización de la aplicación. Se trata de un conjunto de cuatro clases que actuarán conjuntamente para, dado un grafo, determinar todos los caminos mínimos posibles entre puntos y almacenarlo en un fichero que sí está incluido en la aplicación:

- verticeGrafo. Representa cada uno de los vértices del grafo además de las aristas que salen de él. Su diagrama se muestra en la *Figura 5.15*:

verticeGrafo
Atributos
-distAdyacentes[] -predecesor -nombre -adyacentes
Métodos
//getters y setters +masCercano(verticeGrafo[]) +compareTo(verticeGrafo)

Figura 5.15 .- UML de la clase verticeGrafo.

- GeoPunto. Representará un punto geográfico con su longitud y latitud. Su diagrama se muestra en la *Figura 5.16* :

GeoPunto
Atributos
-latitud -longitud
Métodos
//getters y setters

Figura 5.16 .- UML de la clase GeoPunto.

- Grafo. Representa el conjunto de vértices pertenecientes al grafo. El diagrama de la clase de expone en la *Figura 5.17*:

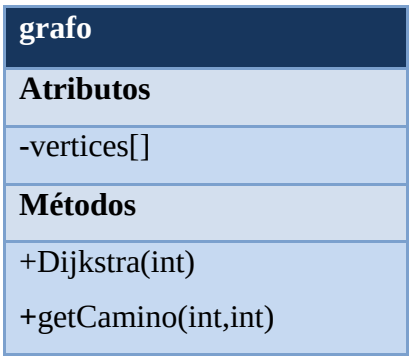


Figura 5.17 .- UML de la clase Grafo.

- Lector. Esta clase se encargará de leer un fichero con el formato "nombre_del_punto \t latitud \t longitud \t lista_vertices_adyacentes \t lista_distancias_a_cada_uno_de_los_puntos", donde \t indica separación por tabuladores; y crear un grafo. A continuación, efectuará el algoritmo de Dijkstra sobre el mismo y guardará el resultado en un fichero de texto. El diagrama de la clase Lector se muestra en la *Figura 5.18*:

Figura 5.18 .- UML de la clase Lector.

La razón por la que el último paquete está al margen del proyecto global es porque, dado que el grafo siempre iba a ser el mismo y que el momento en el que va a ser necesario consume muchos recursos, el equipo decidió que escribir el resultado en un fichero y leerlo desde él era la manera más rápida, efectiva y eficiente de afrontar el problema.

Diseño de la Base de Datos

La aplicación *Itinerarios: Jardines del Buen Retiro* utiliza una base de datos SQLite. Esta base de datos es una base de datos *Open Source* que está embebida dentro del propio sistema operativo *Android* a partir de la versión 2.2 (denominada como *Froyo*). En esta aplicación se está usando como mínimo la versión 2.3.3 – *Gingerbread* de *Android*, por lo que la versión mínima de la base de datos SQLite será la 3.6.22. Esto cambia según la versión de *Android* del dispositivo, pero las versiones más nuevas siguen siendo retro-compatibles.

Motivación de la Base de Datos

Al principio del proyecto se tuvieron graves problemas de diseño causados por el desconocimiento de los requisitos del cliente (ayuntamiento de Madrid y personal del Retiro). Dichos problemas, o más bien dudas sin una diligente respuesta, fueron solucionados más tarde por los desarrolladores, amparándose en lo que estaba en su mano, desde el consejo de Guadalupe y Victoria y basándose en lo que creían mejor para la aplicación. Las dudas que han surgido se pueden clasificar en dos tipos:

- Tipo de recursos: texto, audio e imágenes. ¿A descargar o integrados dentro de la aplicación?
- Tipo de apartados: ¿un punto de la Ruta tiene imágenes asociadas? ¿un punto de interés también tiene imágenes? ¿estas imágenes tienen textos asociados?

Las respuestas a estas preguntas se deben resolver en más de una línea, pues teniendo en cuenta que este proyecto se empezó en octubre y **hasta el día 20 de marzo no se concretaron todos los aspectos de diseño**, el equipo de Itinerarios se fue anticipando a todas las restricciones que se especificaran como requisitos.

Tipo de almacenamiento de recursos

Al principio del proyecto, se sabía que iba a haber texto e imágenes, y también se pensó en la posibilidad de que hubiese locuciones (aunque no se sabía muy bien qué contendrían, ni quién las grabaría, algo que, actualmente, ha sido solucionado por los propios desarrolladores). Se tenían dos vertientes para desarrollar esto; la primera muy primitiva y poco ampliable (incluía todo dentro de la aplicación), mientras que la segunda no permite al usuario usar la aplicación si

no tuviese acceso a Internet (como WI-FI ó tarifa de datos). A continuación se cuentan las desventajas de las dos vertientes:

- Almacenar toda la información incluyendo todas las imágenes y audios necesarios para su funcionamiento **dentro** de la aplicación, es decir en una base de datos del dispositivo móvil, que cuenta con las siguientes características:
    - El tamaño de la apk (o instalador del paquete en la aplicación) sería muy grande, pues contendría fotos y audios internos en la aplicación. Además, esto haría **ocupar mucha memoria** (entre 20 y 50 MB) al dispositivo, y que fuese un motivo de rechazo para descargar la aplicación.
    - Si se quisiera añadir más información habría que introducir nuevo código en la aplicación y sacar una nueva versión. Hacer *a priori* algo **no extensible** no es muy buena *praxis*.
    - La **base de datos es local** a cada persona, por tanto, nunca se podría modificar su estado
    - La única ventaja de esta opción es que no requiere tener acceso a Internet para acceder a los recursos multimedia. Pero sí para ir a un lugar vía GPS.
- No almacenar nada en la aplicación y **bajar todos los recursos de Internet cada vez que se utilice la aplicación**. Esto conlleva unas serias penalizaciones:
    - Se debe disponer de **Internet en todo momento**.
        - En el Retiro no hay *WI-FI,* ni cabe la posibilidad de que lo hubiese algún día.
    - No todo el mundo se puede permitir **tarifa de datos** con Internet en su dispositivo de móvil, la cultura debe estar al alcance de cualquier bolsillo.
    - Si el servidor donde se alojase la base de datos quedase inoperativo por alguna razón, nuestra aplicación **no podría ser usada**. Esto es un problema muy grave
    - Descargar los recursos es **lento** y no se tendría un buen tiempo de respuesta. Haría que la aplicación fuese muy lenta. Esto no es un gran problema, ya que el usuario final no necesita que la información se despliegue muy rápidamente.
    - Se puede modificar la **base de datos común**, ya que todos los usuarios podrían, si se fuese necesario, modificar el estado de la misma.

Finalmente, el equipo de *Itinerarios: Jardines del Buen Retiro* se ha decantado por **una solución híbrida** para el almacenamiento de la información. No tiene sentido tener toda la información almacenada en una base de datos externa cuya estabilidad, en primera instancia, se pudiera caer. Ni tampoco lo tiene que todos los recursos estén dentro del propio paquete instalador (apk). La solución elegida dispone de las siguientes características, que introducidas de la siguiente forma da la solución y acaba con todas las desventajas expuestas en los puntos. A continuación aparecerán las decisiones del diseño de la base de datos y con ellas las ventajas de la esta solución híbrida:

- Nuestra **base de datos es interna** y sólo contiene texto plano: el texto plano ocupa muy poco, apenas tiene límite y sirve para almacenar *URL* de servidores y de los endpoints de cada recurso, además de los extensos textos.
    - Personas sin acceso a Internet pueden recuperar el 60% de la información, ya que tendrán a su disposición todos los textos y el plano del Retiro.
    - No podrán ver los audios, ni las imágenes, ya que se necesita Internet para descargar estos dos tipos de recursos. Las imágenes se pueden almacenar en la memoria interna del dispositivo para verlas sin necesidad de Internet.
        - Siempre que antes se descarguen desde un lugar con WI-FI antes de ir al parque del Retiro
    - Tampoco podrá saber cuál es el punto exacto del punto de la ruta al que quiera llegar mediante la opción “Cómo llegar”, debido a que no podrá conectar con los servidores de *Google*.
- Dentro de la aplicación sólo hay iconos y fondos como recursos multimedia, además del propio plano del Retiro
    - Este último queda incluido dentro de la aplicación para que se pueda consultar cuando se desee, sin dependencia de conexión externa.
- La información de los textos está dentro de la base de datos interna del dispositivo en forma de **texto plano**. El texto plano ocupa muy poco y a la base de datos interna siempre se puede acceder.
- El acceso a los recursos multimedia se realiza de la siguiente forma (ir a al capítulo de Técnicas Algorítmicas para más detalle):

- Se coge la lista de servidores de la tabla Servidor, por ejemplo de la lista { "http://itinerariosretiro1.zz.mu/", "http://itinerariosretiro1.260mb.org/" }, que contiene las URL de los servidores disponibles. Esta lista de servidores está ordenadas de prioridad por la que se quiera acceder a los recursos
- Un recurso cualquiera, por ejemplo, la locución de la introducción, tiene el siguiente *endpoint*: "audio/introduccion.mp3" donde se está especificando el lugar bajo la estructura de directorios donde está el recurso
- Se concatena cada elemento de la lista servidor con el *endpoint* del recurso. En primer lugar se formará http://itinerariosretiro1.zz.mu/audio/introduccion.mp3
    - Se hará previamente una petición HTTP con la primitiva REST GET al recurso donde se consultará la cabecera y se validará el peso del contenido del recurso al que se está accediendo
    - Si todo va bien, se dará el visto bueno y la elección del servidor al que se accederá para recoger el recurso se habrá completado. En caso de que vaya mal, bien por un *timeout* de la conexión o por la simple razón de que el recurso no exista, se elegirá el siguiente host y se volverá al punto anterior
    - Si no hay ningún host que contenga el recurso, o están todos caídos, cabe la posibilidad de acceder al servidor externo dedicado para este proyecto que sí tiene los recursos, pero al que se accederá de manera muy lenta a causa de ser un particular quien nos ofrece el recurso
- De esta manera, cuando se tenga que pedir el recurso, ya se sabe de dónde obtenerlo, pues se ha pre-calculado antes.

- Almacenando de esta manera toda la información, la apk **ocupa 3MB**, lo que es un tamaño bastante pequeño para toda la información y recursos que ofrece
- Además, dado que nuestra aplicación únicamente recupera los recursos de la base de datos y nunca escribe en ella (no hay ningún proceso de registro). Esto propicia una gran ventaja: la base de datos siempre estará operativa y, por tanto, no supondrá una pérdida de funcionalidad
- Por último, se ha minimizado la dependencia externa de servidores, pues se puede dar de alta por ejemplo 20 servidores gratuitos y, con replicar la información que contiene cada

servidor y usar un script de base de datos, se puede aumentar la disponibilidad *online* de la aplicación

La auto-carga de este script *renovado* desde servidores queda pendiente para futuro desarrollo. La idea es la de hacer una opción de *actualización de la base de datos* dentro de la propia aplicación que permitiera incluir nuevas líneas en el script de creación de la base de datos. Como no se sabía el modo de distribución final que iba a tener la aplicación: si iba a estar en la *Play Store* (de esta manera se podría subir nuevas versiones y pedir al usuario que actualice la aplicación de manera trivial), o directamente desde un servidor del ayuntamiento, se decidió posponer la idea para el futuro.

Tipo de apartados: desconocimiento del contenido

Por tipo de apartados se refiere a qué contiene cada pantalla, y de qué manera están relacionados los objetos de un sitio con los de otro. Al principio del proyecto, *Itinerarios: Jardines del Buen Retiro* **no iba a ser únicamente una ruta por los estilos de jardinería del Buen Retiro**, si no que iba a tener distintos tipos de ruta, como la de los monumentos e histórica. Estos cambios de requisitos a mitad de proyecto trajeron una serie de inconvenientes:

- El tener que empezar a desarrollar sin tener claros los requisitos fue una ardua tarea para los desarrolladores, principalmente porque la base de datos tiene que estar asentada desde un primer momento para poder insertar los datos lo más rápido posible y sin entorpecer el desarrollo de la aplicación
- El exceso de trabajo realizado en la creación de tablas y de relaciones, debido a los requisitos que se impusieron más tarde
- Algunas relaciones entre tablas han quedado vacías y sin ningún uso, aunque en futuros proyectos sí que serán utilizadas

No hay que olvidar que este trabajo por adelantado también ha permitido un rápido ingreso de los datos en la base de datos, y ha llevado una serie de anticipaciones a los requisitos del cliente. La mayoría de las relaciones estaban contempladas antes de que oficialmente se especificase lo que quería nuestro cliente. A continuación, se ofrecen algunas características de este avance:

- El ingreso de todos los datos del proyecto mediante los *scripts* de inserción de datos no llevó más de dos horas.
- Ya se disponía del API de la base de datos con las consultas más probables, un ejemplo de estos métodos serían los siguientes (clase *BBDDAPI.java*):
    - ***getImagenesDeRuta(String ruta)*** que devuelve imágenes en forma de una lista de objetos como si fuera la propia tabla de la base de datos en función del punto de la ruta en la que queramos recuperar la información
    - ***getServidores()*** que devuelve todos los servidores en una lista de objetos como si fuera la propia tabla Servidor de la base de datos
- El desarrollo de esta API supuso dejar a un lado la sintaxis de SQL y ocultar todos los detalles de la base de datos al resto del equipo. Con la aplicación de un patrón *singleton*, ya se pudo usar la base de datos con la siguiente llamada al método estático de la clase *BBDDManager.java*:
    - *BBDDAPI db = BBDDManager.getInstance(this, nombre_bbdd, version_bbdd)*
        - *this* es el contexto de la *Activity* donde se llama a la API de la base de datos
        - *nombre_bbdd* es el nombre propio de la base de datos. Está en la clase *Constants.java*
        - *version_bbdd* es la versión de la base de datos. Está en la clase *Constants.java*. Una versión más nueva que la anterior actualiza la base de datos por completo.
- El control de versiones de la base de datos permitía comprobar poco a poco que la inserción en los scripts era correcta.
- Esta base de datos queda como punto de partida para futuros proyectos de este tipo. Es estable y contempla muchos casos posibles en las relaciones y ya están implementadas las *querys* (peticiones o consultas a base de datos de algún tipo de información) más probables.

Por tanto, y como resumen, decir que el trabajo anticipado ha merecido mucho la pena para el rápido desarrollo de la aplicación. Sin él, hubiese costado más tiempo y esfuerzo desarrollarlo, y probablemente se hubiese construido todo más deprisa y las bases de los cimientos serían algo menos sólidos.

Abstracción del contexto

Antes de explicar con más detalle cómo está construida la base de datos, se ha de notificar a modo de aviso o *disclaimer,* de la intención de hacer que la aplicación fuera lo más **versátil** posible.

En un principio, la aplicación iba a ser un multi-itinerario y para este proyecto se iba a contar con tres guías, la de estilos de jardinería del Buen Retiro, la senda Botánica del Retiro y la de Monumentos e Historia del Buen Retiro. Al final se decidió que la aplicación hiciera una sola ruta, siendo la primera de una serie de aplicaciones de itinerarios guiados.

También se habló de unos puntos de interés que estarían geolocalizados y que se podría ir a ellos de la misma manera que se va a un punto de la ruta. Más tarde, esto tampoco entró.
De esta manera, y como se ha adelantado en el apartado anterior, se fue desarrollando la base de datos, siendo también lo más **permisiva** posible para no tener que cambiar demasiado hasta que finalmente el diseño fuese el definitivo.

Para terminar, cabe decir que, con objetivo de servir para más auto-guías que no tengan la necesidad de ser en un parque, no se ha hablado de 'jardines', 'paradas' o estatuas/fuentes' en el esquema de la base de datos. Se describen tipos de ruta, puntos de ruta o puntos de interés. De esta manera, se está simplificando el modo de obtener la información de los lugares en los que hacer una ruta (**abstracción**), con el objetivo de la construcción de una base de datos similar en los futuros proyectos que desarrolle el equipo en el futuro.

Introducción a SQLite

El paquete *android.database.sqlite* de *Android* contiene una API de clases gestoras de la base de datos SQLite que cualquier actividad puede controlar en su propia base de datos (interna, y por tanto, privada). Las aplicaciones (o actividades) pueden modificar el estado de esta base de datos, así como recuperar la información almacenada.

Clases importantes de SQLite y de la base de datos

En este apartado se darán unas leves pinceladas de cómo se abre una base de datos interna SQLite, así como cómo el cierre y la interacción con ella. También se definen las primitivas más básicas y se detallan algunas cuestiones no tan triviales.

La clase SQLiteOpenHelper

Esta clase básica es la que se ha utilizado para crear una base de datos que tiene un breve control de versiones. Se puede consultar en la documentación de Google con más detalle si así se desea [14]. Esta clase tiene una constructora que se llama de la siguiente manera:

- new SQLiteOpenHelper(Context ctx, String database_name, CursorFactory cursores, int version)
    - *ctx* es el contexto de la aplicación, es decir, la *activity* que llamará a esta base de datos
    - *database_name* es el nombre de la base de datos que se guardará en nuestro terminal
    - *cursores* es el patrón factoría de cursores que queremos utilizar en la búsqueda y recuperación de datos. No es necesario usarlo y puede ir a *null*.
    - *version* es la versión de la base de datos, enfrentada con el de la aplicación
        - una versión de la aplicación equivalente a la de la base de datos del terminal llamará al método *onOpen* (abrirá la base de datos y devolverá tal objeto)
        - una versión de la aplicación superior a la de la base de datos del terminal llamará al método *onUpgrade* y posteriormente llamará al método *onOpen*
        - una versión de la aplicación menor a la de la base de datos del terminal llamará al método *onDowngrade* y posteriormente al método *onOpen*

El sistema de la construcción del objeto de la base de datos es intuitivo: se mira si la base de datos existe en el dispositivo. Si existe, se abre (método *onOpen*), y si por el contrario no existe, se crea (se llama al método onCreate) y posteriormente se abre (método onOpen).

Se manifiesta la necesidad de aplicar un patrón *singleton* que sólo permita una instancia de la base de datos y un conjunto de métodos (a gran nivel de abstracción) para ofrecer servicio externo a las actividades que desarrollen otros compañeros mediante el patrón *facade*.

También tiene otros métodos útiles que la propia cabecera de los mismos define sus características:

- *SQLiteDatabase getReadableDatabase()*: Devuelve un objeto SQLiteDatabase creado o abierto para lectura únicamente

- *SQLiteDatabase getWritableDatabase()*: Devuelve un objeto SQLiteDatabase cread o abierto para escritura únicamente

SQLiteDatabase

Es el objeto que se recibe cuando se ejecutan los métodos últimos del apartado anterior. De la misma manera, se puede ver las características de esta clase en la [Bibliografía]. Este objeto se puede realizar con las siguientes primitivas:

- *void close():* cierra la base de datos apuntada por el objeto *SQLiteDatabase*
- *void execSQL(String query): query* es la consulta a hacer en la base de datos
- *Cursor rawQuery(String query, String[] argumentos)*
    - *query* es la consulta a hacer en la base de datos
    - *argumentos* es un array de Strings para la substitución de variables dentro de la *query*.

Cursor

Esta abstracción de la recuperación de datos tras una consulta de la base de datos es muy potente. Permite el procesamiento de lo recuperado de la misma manera que un patrón *iterator*. Dispone de los siguientes métodos:

- *boolean moveToFirst()*: permite seleccionar el cursor a la primera fila de la recuperación de datos procedente de la base de datos
- *boolean moveToNext():* nos mueve el cursor a la fila siguiente de lo recuperado
- *String getString(int columna)*
    - *columna* es el número de columna de la fila donde está apuntando actualmente el cursor

Un ejemplo se puede ver en la *Figura 5.19* la cual muestra extracto de código donde se puede observar la manera de recorrer esta clase:

```
public static ArrayList<TipoRuta> queryTipoRutas(SQLiteDatabase db) {
    // Creamos un ArrayList que contiene Tipos de Ruta
    ArrayList<TipoRuta> aL = new ArrayList<TipoRuta>();
    // Hacemos una consulta a base de datos y la guardamos en el cursor
    Cursor c = db.rawQuery("SELECT * FROM tipo_ruta", null);
    // Movemos el cursor a la primera fila
    if (c.moveToFirst()) {

        do {
            // Empezamos a leer de izquierda a derecha
            //Cogemos el valor de la primera columna
            String id = c.getString(0);
            //Cogemos el valor de la segunda columna
            String tipo = c.getString(1);
            //Cogemos el valor de la tercera columna
            String comentario = c.getString(2);
            //Creamos nuestra representación del objeto Tipo de Ruta
            //con lo que hemos recogido
            TipoRuta tipo_ruta = new TipoRuta(id, tipo, comentario);
            //Lo añadimos a nuestra lista
            aL.add(tipo_ruta);
            //Nos movemos a la siguiente fila hasta que haya filas a las
            //que movernos
        } while (c.moveToNext());

    } // fin if
    //Devolvemos el ArrayList con todas las
    return aL;
} // fin método queryTipoRutas
```

Figura 5.19 .- Modo de recorrido de un cursor

Concurrencia

A la hora del desarrollo de la base de datos, no se sabía con seguridad si habría servicios por debajo de una actividad haciendo peticiones a base de datos cada cierto intervalo de tiempo. Aunque la aplicación sólo hace una petición por cada carga de una actividad, el recurso del objeto de la base de datos (gracias al patrón *singleton*), no permite ser usado por distintos hilos del sistema. Era una mejora de la base de datos que no está siendo aprovechada, pero que merece la pena mencionar para futuros proyectos.

Inserción de datos y actualización de la base de datos

Toda base de datos tiene algún script de creación de tablas según un diseño pre-establecido de tablas y relaciones. Estos ficheros de información se leen en forma de *StringBuffer*, de esta manera se almacena en memoria la cadena del *StringBuffer* y las *querys* pueden ser tan largas

como se quieran. Estos scripts están en la carpeta *assets* dentro del proyecto y se llaman de la siguiente manera.

- *generadorTablas.txt:* como bien dice su nombre, es un *script* de generación de todas las tablas de la base de datos, así como las relaciones entre las mismas.
- *generadorDatos.txt: script* que genera todos los datos que se consultan en la base de datos dentro de la aplicación
- *borradorTablas.txt: script* que borra las tablas de la base de datos. A la hora de actualizar, gracias al versionado de la base de datos es posible que se quiera quitar una tabla obsoleta o cambiarle el nombre por otra, por lo que es necesario poder borrar todas las tablas del dispositivo para poderlas posteriormente volver a crear desde cero. Este proceso con pocas tablas (y de tamaño reducido) en el sistema no tiene apenas coste

Tipos de SQLite

En este apartado se quiere seleccionar únicamente los tipos que se han utilizado. Pese a que el lenguaje SQLite tiene una gran variedad de tipos, hemos elegido únicamente tres, ya que el resto no nos hacía falta. Estos tipos se enumeran a continuación:.

- *INT*: utilizado para organizar las tablas por índices y ordenar los registros por una clave primaria
- *NVARCHAR*: utilizado para almacenar texto dentro de la base de datos y para posteriormente recuperarlo mediante alguna consulta en la mayoría de los sistemas como *SQLServer*, *NVARCHAR* es codificado en *Unicode* a diferencia de *VARCHAR* que está en *ASCII*. En *Android* todo está en *Unicode*, por lo que el resultado es el mismo.
- *FLOAT*: utilizado para los puntos de mapa y su representación en coordenadas decimales para la integración de las APIs de *Google Maps*

Diagramas entidad-relación

Como viene siendo habitual, para hablar de base de datos, se tiene que hablar de diagramas Entidad – Relación, apartado que se ha estructurado en tres partes, con el objetivo de definir cada una de ellas por separado.

La entidad Servidor

Esta entidad es la única que no está relacionada con ninguna otra, ya que no forma parte de 'tiene un' típica de una relación. En la siguiente *Figura 5.20* se muestra el modelo de entidad-relación de la entidad Servidor.

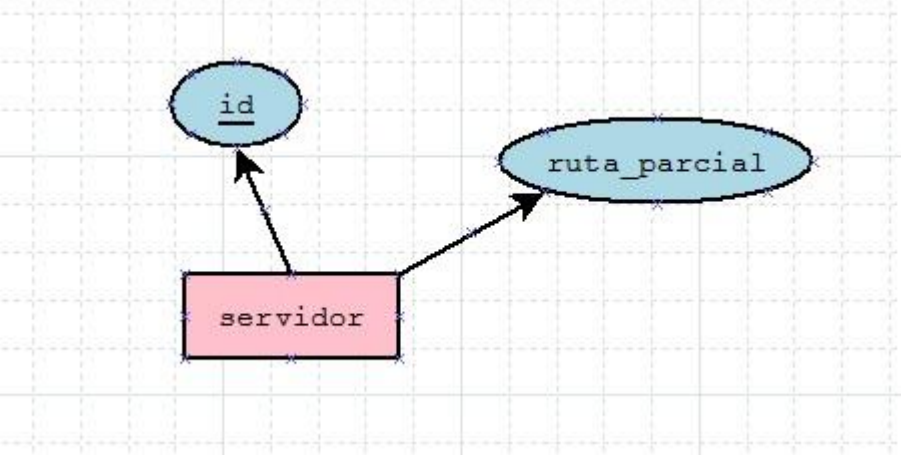


Figura 5.20 .- Modelo Entidad-Relación de servidor

Como se puede observar, tiene los siguientes atributos:

- *id*: es clave primaria y, por tanto, no se puede repetir otro mismo *id* en esta propia tabla. Este atributo sirve para recuperar la lista de los servidores ordenados por prioridad, el que tenga menor id tendrá más prioridad
- *ruta_parcial*: contiene el directorio raíz de la URI de un servidor, como en el apartado de Motivación, un buen ejemplo sería "http://itinerariosretiro1.zz.mu/"

La entidad Tipo de Ruta

La entidad Tipo de Ruta iba a ser la entidad principal de la aplicación *Itinerarios: Jardines del Buen Retiro*. El problema, como se ha comentado a lo largo de este documento, es el acorte de rutas a sólo una. En un primer inicio, se pensó que también tendría un texto asociado introductorio que definiera la aplicación, aunque también ha quedado en desuso para este proyecto, pero no para los siguientes. En la *Figura 5.21* se puede ver el modelo Entidad – Relación de la entidad Tipo de Ruta.

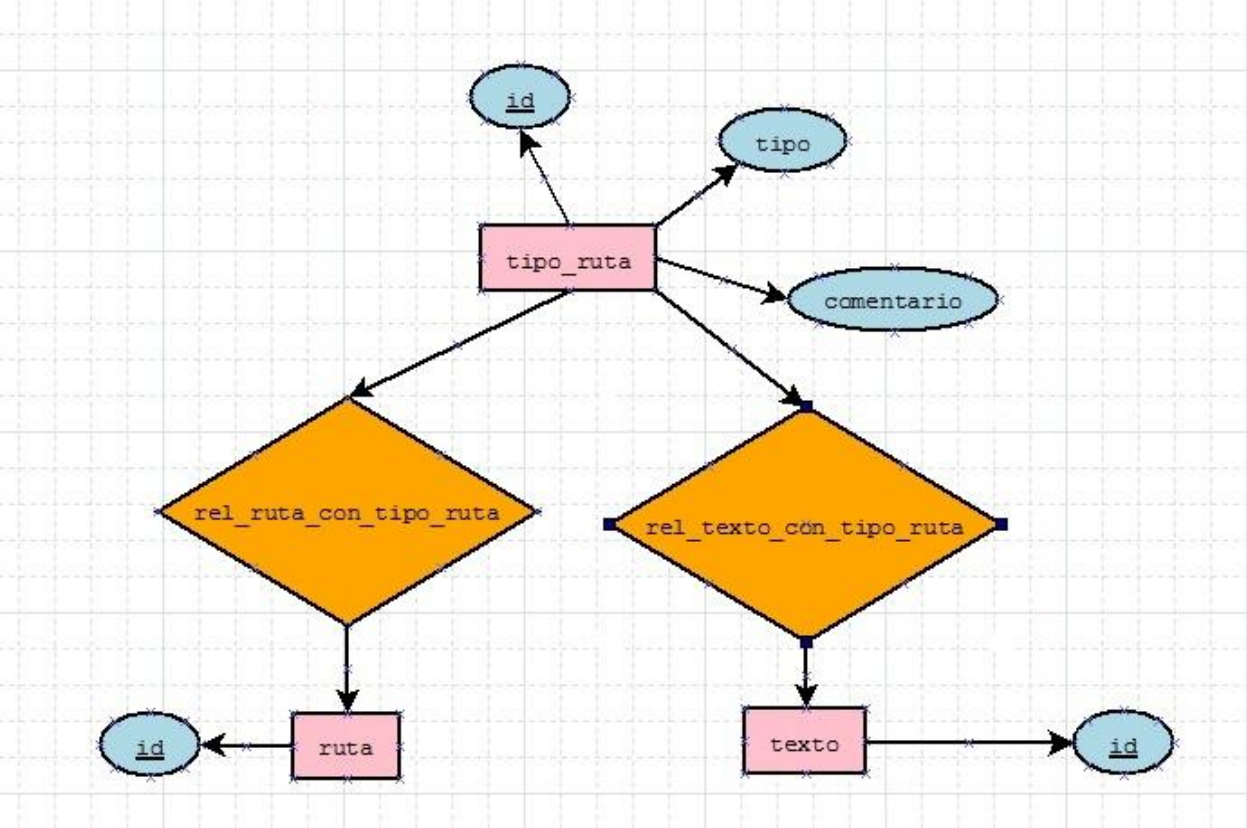


Figura 5.21 .- Modelo Entidad – Relación de Tipo de Ruta

Se puede apreciar que han aparecido rombos de color naranja, esto expresa las relaciones entre entidades. Cada relación está formada por los atributos de clave primaria de las entidades que relaciona. Los atributos de la entidad Tipo de Ruta son los siguientes:

- *id*: como siempre, la clave primaria de cada registro contenido en la tabla
- *tipo*: tipo de Itinerario, como podría ser 'Jardines del Buen Retiro'
- *comentario*: una breve descripción de la ruta a realizar

Resto de entidades y sus relaciones

Viendo la dinámica del modelo Entidad – Relación del diseño de la base de datos, con respecto al anterior diagrama de la *Figura 5.21*, es fácil deducir las relaciones y origen de los atributos en el diagrama de la *Figura 5.22* que viene a continuación:

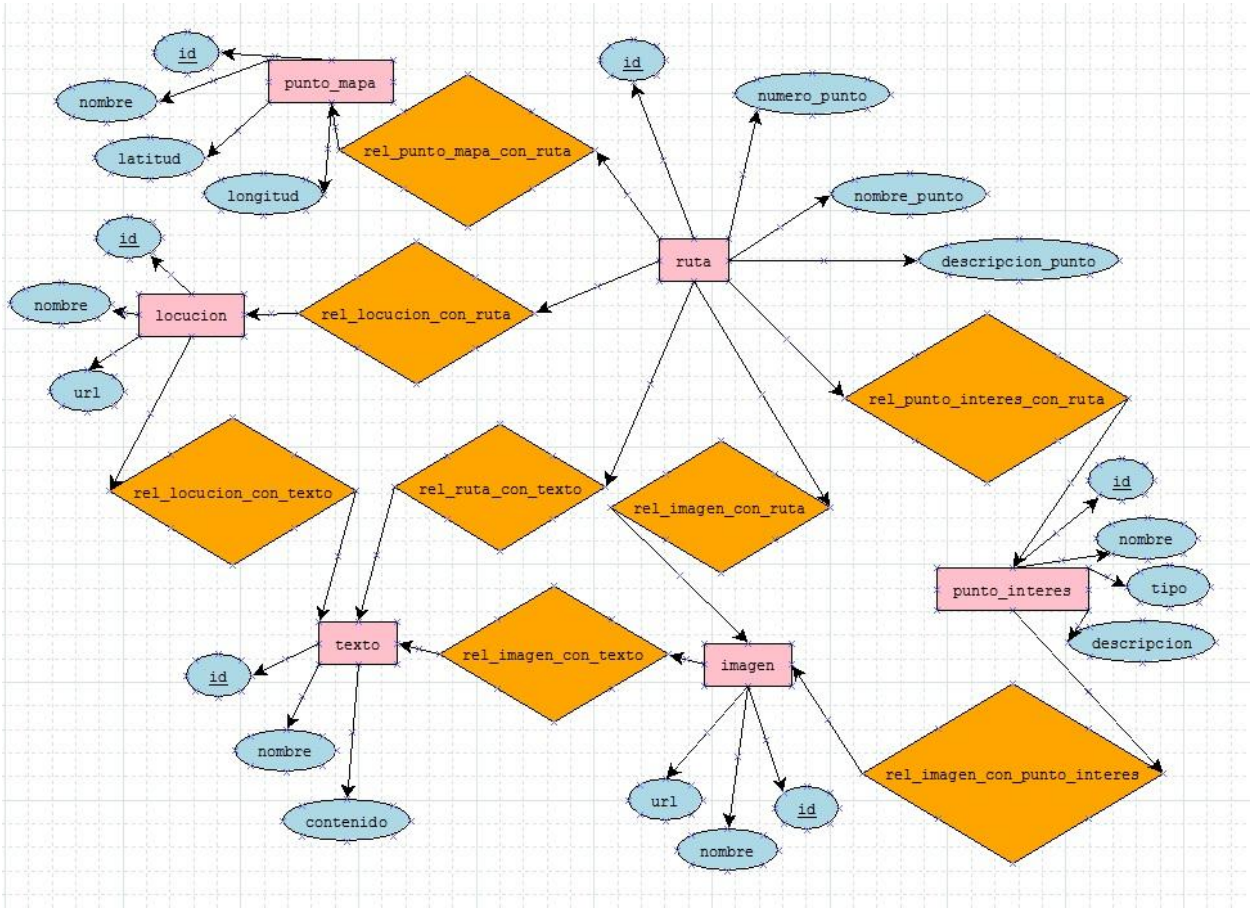


Figura 5.22 .- Modelo de Entidad – Relación del resto de entidades

A continuación se defininen las entidades y sus atributos de manera breve:

- **Entidad Ruta:** representa cada punto de una Ruta específica.
    - *id*: clave primaria que distingue los registros
    - *numero_punto*: posición ordinal del punto de la ruta que ocupa (por ejemplo, 1º, 2º...16º)
    - *nombre_punto*: denominación corta del lugar del punto
    - *descripcion_punto*: descripción extendida del lugar del punto

- **Entidad Punto de Mapa:** representa los puntos del mapa para la integración con *Google Maps*
    - *id*: clave primaria que distingue los registros
    - *nombre*: breve nombre del lugar del punto asociado
    - *latitud*: medida decimal de la ubicación vertical. Desde el ecuador, hacia el Norte es positivo y hacia el Sur es negativo
    - *longitud*: medida decimal de la ubicación horizontal. Desde el meridiano de *Greenwich*, hacia el Este es positivo y hacia el Oeste es negativo
- **Entidad Locución:** representa las locuciones o reproducciones de audio
    - *id*: clave primaria que distingue los registros
    - *nombre*: breve nombre de la locución a reproducir
    - *url*: parte de la URI para formar el endpoint, por ejemplo: "audio/introduccion.mp3"
- **Entidad Texto:** representa los textos a mostrar de cualquier lugar
    - *id*: clave primaria que distingue los registros
    - *nombre*: breve nombre del texto asociado
    - *contenido*: una vasta extensión de texto
- **Entidad Imagen:** representa las imágenes a descargar de la aplicación
    - *id*: clave primaria que distingue los registros
    - *nombre*: breve nombre de la locución a reproducir
    - *url*: parte de la URI para formar el *endpoint*, por ejemplo: "fotos/CecilioRodriguez3.JPG"
- **Entidad Punto de Interés:** representan los lugares del "sabías qué" de la aplicación
    - *id*: clave primaria que distingue los registros
    - *nombre*: breve nombre del punto de interés
    - *tipo*: actualmente sólo hay dos tipos: lugar/árbol
    - *descripción*: extensa descripción del punto de interés

Las relaciones se creen que son fácilmente deducibles y, como se dijo al principio, estas relaciones pueden cambiar para futuros proyectos. En la siguiente sección se explica con más detalle el diagrama de tablas SQL.

Diagrama de Tablas SQL y su representación en Java

La traducción inmediata del diagrama Entidad – Relación son las tablas en el lenguaje SQL. Cada entidad no-débil se representa como una tabla y si se quieren mantener las formas normales propias de los cánones de bases de datos, las relaciones de tablas tienen que ser una a una, cumpliéndose así con los esquemas de normalización de una base de datos

En la *Figura 5.23* se puede ver un diagrama que ilustra gráficamente la implementación en SQLite de las tablas. Idealmente, cada tabla que representa a una identidad será una clase Java dentro del código y una relación será un método propio de la *BBDDAPI.java* (interfaz que actúa de fachada o *facade*). Por ejemplo:

- dame todas las imágenes de este punto de ruta:
    - se devolverá un ArrayList<Imagen> en este método
    - *public ArrayList<Imagen> getImagenesDeRuta(Ruta ruta);*

En el diagrama se pueden resolver, gracias a la ilustración, todas las preguntas que quedasen pendientes en cuanto a implementación en el diagrama del modelo Entidad – Relación:

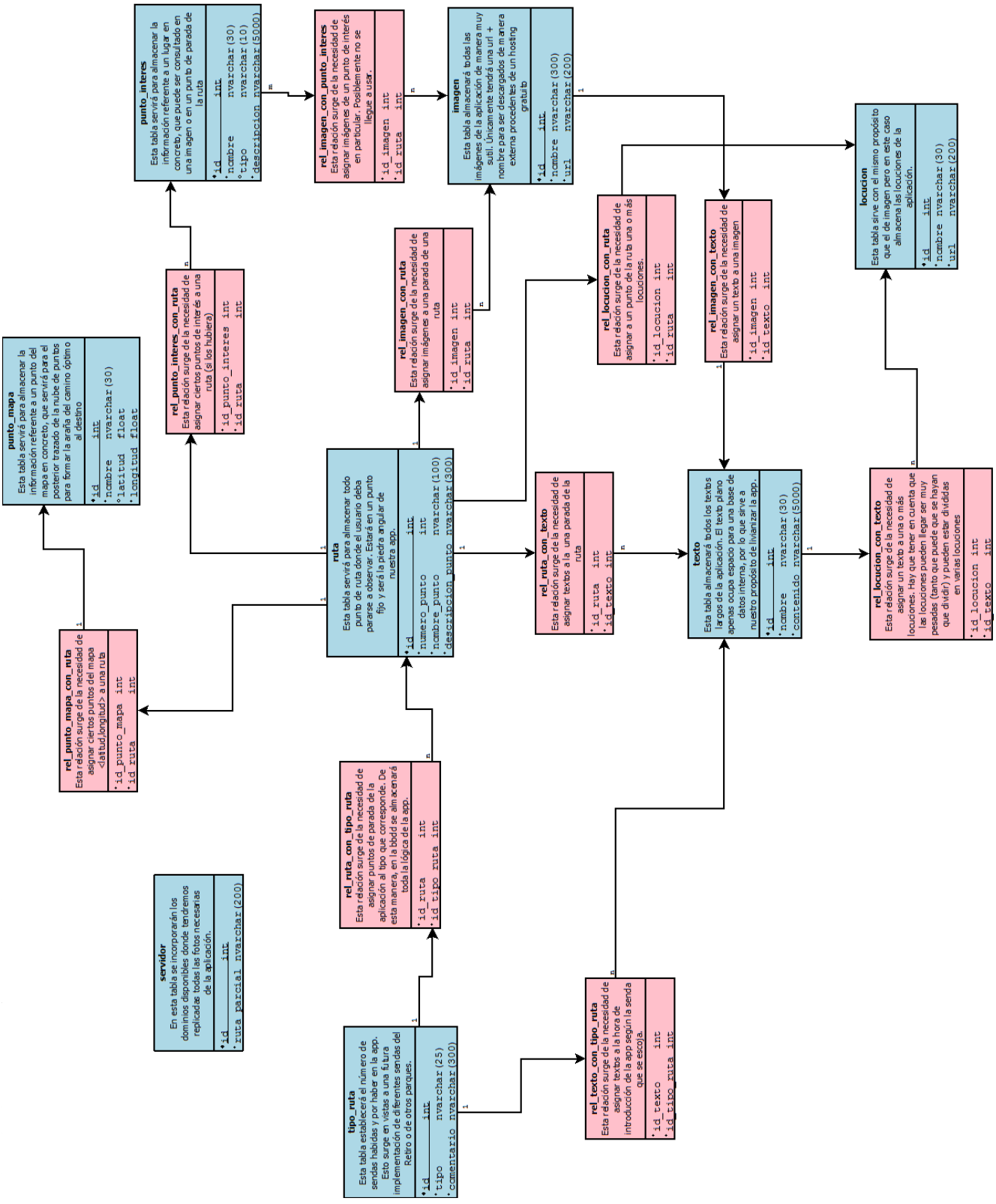


Figura 5.23 .- Diagrama de Tablas SQL de la Base de Datos

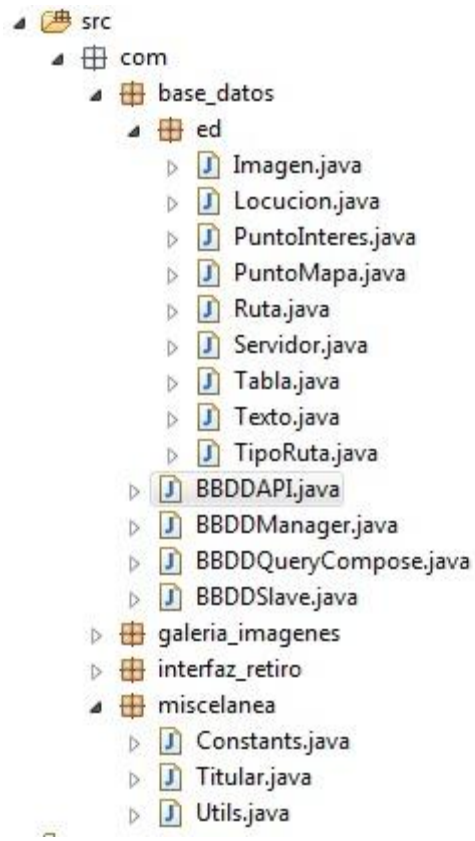


Figura 5.24 .- Clases Java que gestionan la base de datos

De todas las clases que se muestran en la *Figura 5.24*, las que **representan entidades** son las siguientes:

- *Imagen.java*: representa a la entidad Imagen
- *Locucion.java*: representa a la entidad Locución
- *PuntoInteres.java*: representa a la entidad Punto de Interés
- *Ruta.java*: representa a la entidad Ruta
- *Servidor.java*: representa a la entidad Servidor
- *Tabla.java*: no representa a ninguna entidad, pero todas las entidades heredan de ella y todas las entidades tienen en común el *id*
- *Texto.java*: representa a la entidad Texto
- *TipoRuta.java*: representa a la entidad Tipo de Ruta

Por otra parte están las **clases gestoras** que llevan la gestión de la base de datos interna de la aplicación y aparecen a continuación:

- *BBDDAPI.java*: sirve de interfaz común desde la base de datos a la aplicación que la invoca. Tiene métodos útiles de recuperación de información.
- *BBDDManager.java*: es la piedra angular de la gestión de la base de datos
    - hereda de SQLiteOpenHelper e implementa a la interfaz BBDDAPI.java
    - trata la creación/apertura/actualización de la base de datos
    - delega las consultas a la clase BBDDQueryCompose.java
    - llama a BBDDSlave.java cuando se tienen que leer ficheros de script para la creación o borrado de tablas
- *BBDDQueryCompose.java*: ejecuta las queries y devuelve los registros obtenidos en forma de vectores de entidades y se los sirve a BBDDManager.java
- *BBDDSlave.java*: se encarga de la lectura de ficheros. Los ficheros pueden ser cadenas de texto extremadamente largas y esto se soluciona almacenando la información en búferes de cadenas de caracteres.

Sin olvidarnos de la clase de constantes que son las que contienen todos los asuntos comunes concernientes al resto de clases:

- *Constants.java*: posee todas las constantes de la aplicación

En la *Figura 5.25* se ilustra con un diagrama UML estas relaciones entre clases que acabamos de explicar:

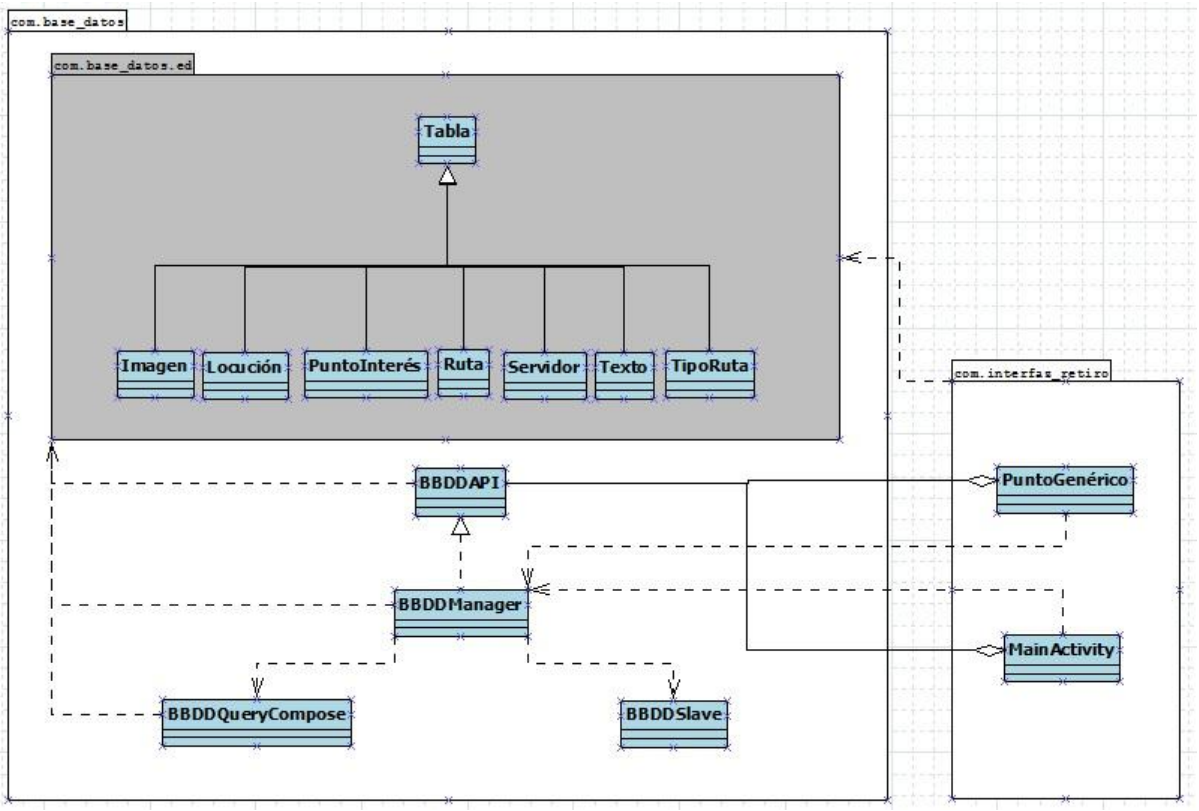


Figura 5.25 .- Diagrama UML de las clases de la base de datos

Librerías externas utilizadas

La clase ScaleImageView

Para dar una mejor visualización tanto a las imágenes presentes en cada punto como al mapa del parque, se ha adaptado una librería externa que permite realizar zoom sobre dichos elementos multimedia. A continuación se van a comentar e ilustrar algunos de los fragmentos y funciones más importantes del código de dicha clase.

Como se puede ver en la *Figura 5.26* la clase *ScaleImageView*[42][43] hereda de la clase *ImageView*, una clase propia de *Android* para la muestra o carga de imágenes de distintas fuentes. Además implementa la clase *OnTouchListener*, redefiniendo sus métodos que veremos más adelante. También se observan las importaciones para el uso de clases externas propias de *Android*, sobre todo aquello relativo a partes gráficas del entorno (*android.graphics.**)

```
import android.content.Context;
import android.graphics.Bitmap;
import android.graphics.Matrix;
import android.graphics.drawable.Drawable;
import android.util.AttributeSet;
import android.util.FloatMath;
import android.view.MotionEvent;
import android.view.View;
import android.view.View.OnTouchListener;
import android.widget.ImageView;

public class ScaleImageView extends ImageView implements OnTouchListener {
```

Figura 5.26 .- Herencia e implementación de la clase.

Algunos los atributos más importantes de la son aquellos relacionados con la escala de la imagen sobre la cual se va a aplicar el aumento y variable numéricas para contabilizar las coordenadas donde se hace el zoom. También se observan atributos booleanos para comprobar el aumento y la matriz donde encuadramos la imagen. La *Figura 5.27* representa dicho código.

```
private float MAX_SCALE = 2f;
private int DOUBLE_TAP_SECOND = 400;

private Matrix mMatrix;
private final float[] mMatrixValues = new float[9];

private int mIntrinsicWidth;
private int mIntrinsicHeight;

private float mScale;
private float mMinScale;

// double tap for determining
private long mLastTime = 0;
private boolean isDoubleTap;
private int mDoubleTapX;
private int mDoubleTapY;

private float mPrevDistance;
private boolean isScaling;

private int mPrevMoveX;
private int mPrevMoveY;

String TAG = "ScaleImageView";
```

Figura 5.27 .- Atributos de la clase.

También es importante comentar algunas de las accesoras, mutadoras y constructoras de la clase. Entre las accesoras se observan aquellas que devuelven valores de coordenadas de la matriz o de la escala de la imagen. En cuanto a constructoras, se inicializa el contexto, los valores de la matriz y se inicializa la recepción de eventos de presión sobre la imagen en la pantalla con *setOnTouchListener(this)*. En la *Figura 5.28* se puede ver el código relativo a dichos métodos.

```
public ScaleImageView(Context context, AttributeSet attr) {
    super(context, attr);
    initialize();
}

public ScaleImageView(Context context) {
    super(context);
    initialize();
}

public boolean isScaling() {
    return isScaling;
}

public void setScaling(boolean isScaling) {
    this.isScaling = isScaling;
}

@Override
public void setImageBitmap(Bitmap bm) {
    super.setImageBitmap(bm);
    this.initialize();
}

private void initialize() {
    isScaling = false;
    this.setScaleType(ScaleType.MATRIX);
    this.mMatrix = new Matrix();
    Drawable d = getDrawable();
    if (d != null) {
        mIntrinsicWidth = d.getIntrinsicWidth();
        mIntrinsicHeight = d.getIntrinsicHeight();
        setOnTouchListener(this);
    }
}

protected float getValue(Matrix matrix, int whichValue) {
    matrix.getValues(mMatrixValues);
    return mMatrixValues[whichValue];
}

protected float getScale() {
    return getValue(mMatrix, Matrix.MSCALE_X);
}

protected float getTranslateX() {
    return getValue(mMatrix, Matrix.MTRANS_X);
}

protected float getTranslateY() {
    return getValue(mMatrix, Matrix.MTRANS_Y);
}
```

Figura 5.28 .- Getters, Setters y Contructores de la clase.

Un par de métodos interesantes a comentar sobre el aumento de la imagen son *maxZoomTo* y *zoomTo*. Ambos métodos combinados realizan el cálculo de la nueva escala a la que se somete la matriz y cambian dicha escala en función de altura y anchura. Además realizan el control sobre el aumento máximo y mínimo que se puede hacer en la imagen. Todo este código queda representado en la *Figura 5.29*:

```
protected void maxZoomTo(int x, int y) {
    if (mMinScale != getScale() && (getScale() - mMinScale) > 0.1f) {
        // threshold 0.1f
        float scale = mMinScale / getScale();
        zoomTo(scale, x, y);
    } else {
        float scale = MAX_SCALE / getScale();
        zoomTo(scale, x, y);
    }
}

protected void zoomTo(float scale, int x, int y) {
    if (getScale() * scale < mMinScale) {
        return;
    }
    if (scale >= 1 && getScale() * scale > MAX_SCALE) {
        return;
    }
    mMatrix.postScale(scale, scale);
    // move to center
    mMatrix.postTranslate(-(getWidth() * scale - getWidth()) / 2,
            -(getHeight() * scale - getHeight()) / 2);

    // move x and y distance
    mMatrix.postTranslate(-(x - (getWidth() / 2)) * scale, 0);
    mMatrix.postTranslate(0, -(y - (getHeight() / 2)) * scale);
    setImageMatrix(mMatrix);
}
```

Figura 5.29 .- Métodos sobre el control del zoom máximo mínimo y la escala.

El método más importante de la clase es el *onTouchEvent*. Este método, redefinido de la clase *OnTouchListener*, actúa con una distinción de casos dependiendo de qué tipo de presión se produzca sobre la pantalla: desplazamiento, aproximación con dos dedos hacia dentro o hacia afuera y doble toque táctil de aumento o reducción. Dependiendo del evento producido se recalcularán coordenadas, escalas, anchuras y alturas de la nueva imagen que se mostrará o el zoom máximo que se podrá realizar. Todo el proceso hace llamadas a métodos descritos anteriormente y métodos propios del *framework* de *Android*. En la *Figura 5.30* se puede observar el código del método OnTouchEvent.

```
@Override
public boolean onTouchEvent(MotionEvent event) {
    int touchCount = event.getPointerCount();
    switch (event.getAction()) {
    case MotionEvent.ACTION_DOWN:
    case MotionEvent.ACTION_POINTER_1_DOWN:
    case MotionEvent.ACTION_POINTER_2_DOWN:
        if (touchCount >= 2) {
            float distance = distance(event.getX(0), event.getX(1),
                    event.getY(0), event.getY(1));
            mPrevDistance = distance;
            isScaling = true;
        } else {
            if (System.currentTimeMillis() <= mLastTime + DOUBLE_TAP_SECOND) {
                if (30 > Math.abs(mPrevMoveX - event.getX())
                        + Math.abs(mPrevMoveY - event.getY())) {
                    isDoubleTap = true;
                    mDoubleTapX = (int) event.getX();
                    mDoubleTapY = (int) event.getY();
                }
            }
            mLastTime = System.currentTimeMillis();
            mPrevMoveX = (int) event.getX();
            mPrevMoveY = (int) event.getY();
        }
        break;
    case MotionEvent.ACTION_MOVE:
        if (touchCount >= 2 && isScaling) {
            float dist = distance(event.getX(0), event.getX(1),
                    event.getY(0), event.getY(1));
            float scale = (dist - mPrevDistance) / dispDistance();
            mPrevDistance = dist;
            scale += 1;
            scale = scale * scale;
            zoomTo(scale, getWidth() / 2, getHeight() / 2);
            cutting();
        } else if (!isScaling) {
            int distanceX = mPrevMoveX - (int) event.getX();
            int distanceY = mPrevMoveY - (int) event.getY();
            mPrevMoveX = (int) event.getX();
            mPrevMoveY = (int) event.getY();
            mMatrix.postTranslate(-distanceX, -distanceY);
            cutting();
        }
        break;
    case MotionEvent.ACTION_UP:
    case MotionEvent.ACTION_POINTER_UP:
    case MotionEvent.ACTION_POINTER_2_UP:
        if (event.getPointerCount() <= 1) {
            isScaling = false;
            if (isDoubleTap) {
                if (30 > Math.abs(mDoubleTapX - event.getX())
                        + Math.abs(mDoubleTapY - event.getY())) {
                    maxZoomTo(mDoubleTapX, mDoubleTapY);
                    cutting();
                }
            }
        }
        isDoubleTap = false;
        break;

    }
    return true;
}
```

Figura 5.30 .- Método OnTouchEvent, el más importante de la clase.

Por último, comentar a nivel de código el método *cutting()*, que es el encargado de calcular todas las coordenadas en la matriz para pintar la imagen final, posterior al evento producido, ya sea aumento o reducción del zoom. Podemos observar su código en la *Figura 5.31*:

```
public void cutting() {
    int width = (int) (mIntrinsicWidth * getScale());
    int height = (int) (mIntrinsicHeight * getScale());
    if (getTranslateX() < -(width - getWidth())) {
        mMatrix.postTranslate(-(getTranslateX() + width - getWidth()), 0);
    }
    if (getTranslateX() > 0) {
        mMatrix.postTranslate(-getTranslateX(), 0);
    }
    if (getTranslateY() < -(height - getHeight())) {
        mMatrix.postTranslate(0, -(getTranslateY() + height - getHeight()));
    }
    if (getTranslateY() > 0) {
        mMatrix.postTranslate(0, -getTranslateY());
    }
    if (width < getWidth()) {
        mMatrix.postTranslate((getWidth() - width) / 2, 0);
    }
    if (height < getHeight()) {
        mMatrix.postTranslate(0, (getHeight() - height) / 2);
    }
    setImageMatrix(mMatrix);
}
```

Figura 5.31 .- Metodo cutting, recalcula coordenadas en la matriz.

Ejemplo práctico del uso con el mapa del Retiro

Un ejemplo práctico del uso de la clase descrita anteriormente es el del mapa del parque. Para poder realizar zoom sobre la imagen se ha definido previamente en el .xml de la pantalla, el *ImageView* con la particularidad que se expone a continuación: se trata de un *ImageView*, pero que aplica la clase *ScaleImageView* para poder realizar *zoom* sobre la imagen. La parte "*com.galeria_imagenes*" representa el nombre del paquete bajo el cual se localiza la clase *ScaleImageView*. Toda esta descripción del XML se plasma en la *Figura 5.32:*

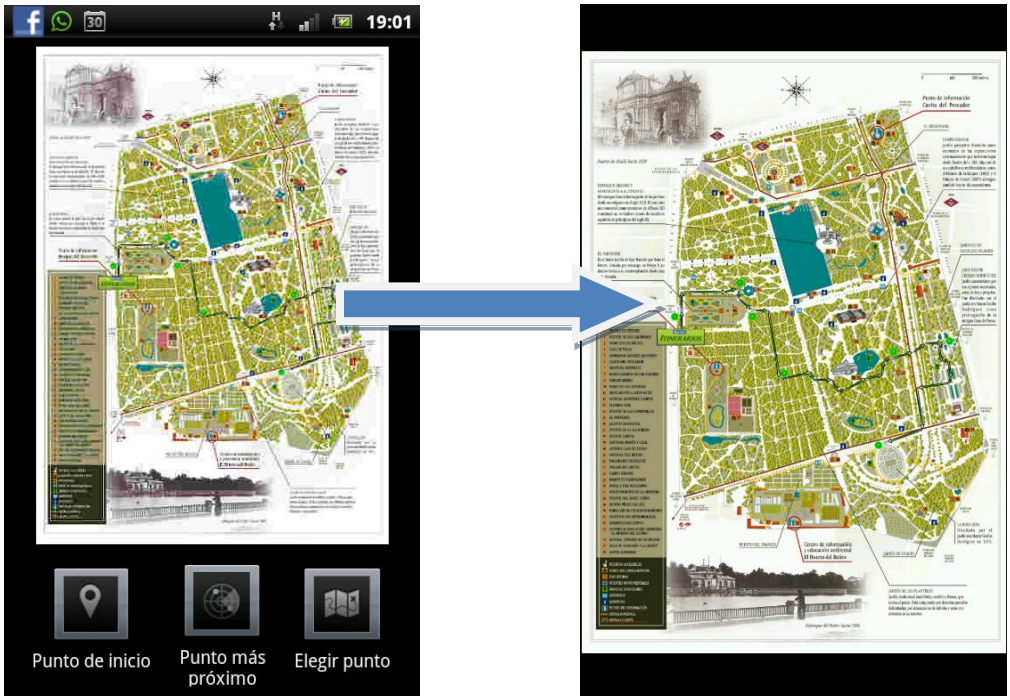
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@color/Black"
    android:orientation="vertical" >

      <com.galeria_imagenes.ScaleImageView
          android:id="@+id/imageView1"
          android:layout_width="fill_parent"
          android:layout_height="fill_parent"
          android:layout_margin="1dp"
          android:src="@drawable/mapa_retiro" />

</LinearLayout>
```

Figura 5.32 .- Archivo .xml del mapa del parque que aplica ScaleImageView.

Una vez en la pantalla de “Iniciar ruta”. Si se pulsa sobre el mapa, se despliega ese mismo mapa en pantalla completa como muestra la cadena de imágenes de la *Figura 5.33*.



Pantalla “Iniciar ruta”. ***Pantalla del mapa completo.***

Figura 5.33 .- Hacer zoom sobre el mapa del Buen Retiro.

Cuando el mapa se sitúa en pantalla completa se puede realizar *zoom*, como se muestra en la *Figura 5.34*. El resultado de ello se observa en la *Figura 5.35*. También se puede hacer zoom sobre el mapa con un doble *click* táctil, en cuyo caso se apreciará un zoom máximo dependiendo de la resolución de la imagen.

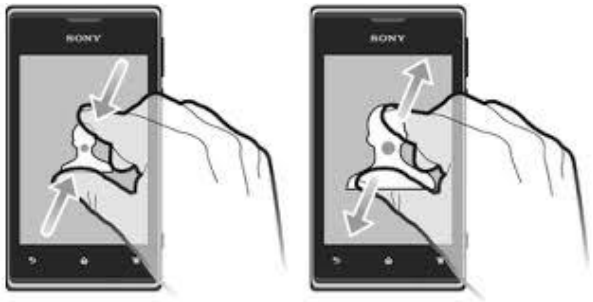


Figura 5.34 .- Para realizar aumento concreto con dos dedos.

Figura 5.35 .- Resultado de hacer zoom sobre el mapa anterior.

Librería Universal Image Loader

Para el proyecto *Itinerarios: Jardines del Buen Retiro* se hizo necesaria la muestra de imágenes incrustadas en la aplicación. Se necesitaba buscar algún recurso o una librería que permitiera realizar esto antes de ser programado por el equipo. Las necesidades de tal búsqueda no eran triviales, pues se necesitaba lo siguiente:

- Las imágenes se deben descargar de Internet y, si hay un error en la transferencia, sea cual fuere el motivo se debe informar de ello
- Pantalla previa que muestre un listado de imágenes que contengan icono de una imagen y una descripción para que el usuario elija la que desee
- Tras seleccionar una de las imágenes, se tendría que mostrar la imagen a pantalla completa y si la imagen está diseñada para que fuese horizontal (si se tiene la opción del giroscopio activada) se tendría que girar adecuadamente. Además, debe ofrecer zoom al usuario para que pueda consultar con mayor detalle los diferentes aspectos de la imagen
- Debe ofrecer la posibilidad de dejar guardada la imagen dentro del dispositivo almacenándola en la memoria SD. De esta manera, el usuario no necesita tener *Internet* en el parque del Retiro mientras interactúa con la aplicación para ver sus imágenes, pues previamente ha podido descargarlo desde algún lugar donde sí tuviese *Internet*. También debe garantizar que estas imágenes se borren si el usuario así lo desea

Todas estas características las podemos encontrar y adaptar de la librería de Android Universal Image Loader (c) desarrollada por Sergey Tarasevich y distribuida bajo la licencia Apache 2.0. El objetivo de este apartado es el uso de esta librería y la adaptación al proyecto *Itinerarios: Jardines del Buen Retiro*. Se puede encontrar más información del funcionamiento interno de esta librería, así como del término de la licencia en el apartado de la bibliografía [37].

Despliegue

Para el uso de esta librería es necesario seguir una serie de sencillos pasos para poder, desde la aplicación, hacer uso de sus funcionalidades. Para comenzar, hay que declarar en el fichero *AndroidManifest.xml* de la aplicación los siguientes permisos de uso:

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

- Se necesita conexión a Internet para descargar las imágenes desde cualquier servidor externo que ofrezca este recurso

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

- En este proyecto, es necesario escribir en la memoria SD interna del dispositivo y por ello se debe activar este permiso

Después de realizar estos cambios en el *AndroidManifest.xml*, se debe descargar el fichero *Android-Universal-Image-Loader-master.zip* de *Github* [37]. A continuación descomprimimos los siguientes archivos y los colocaremos en donde muestra la *Figura 5.36*:

- *\downloads\universal-image-loader-x.y.z.jar*
- *\downloads\universal-image-loader-x.y.z-javadoc.jar*: Se hace necesario consultar el Javadoc de esta librería porque ofrece muchas posibilidades y están muy bien detalladas.

Estos archivos se deben descomprimir dentro de la carpeta *libs* del proyecto *Android* a la altura del *AndroidManifest.xml* como se ilustra en la *Figura 5.36*.

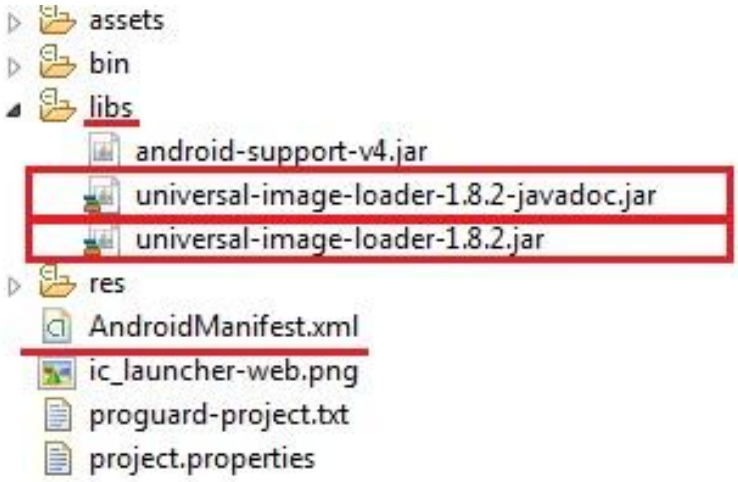


Figura 5.36 .- Lugar de descompresión de los archivos de la librería

Posteriormente, hay que construir la ruta de construcción a estas librerías externas de las que dependemos, para ello, hay que seguir una serie de pasos:

1. Seleccionar el proyecto maestro de *Android* y hacer click derecho sobre él
2. Seleccionar la opción *Build Path* y luego *Configure Build Path*
3. Seleccionar la pestaña *Libraries* y pulsar en *Add JARs*

4. Elegir los dos ficheros descomprimidos dentro de la carpeta libs: *universal-image-loader-x.y.z.jar* y universal-image-loader-x.y.z-javadoc.jar
5. Se debe observar que los .jar se han añadido en el panel de "*JARs and class folders on the build path*".
6. Pulsar OK para validar la operación

Con estos pasos ya podemos utilizar las clases que están internas en esta librería. La *Figura 5.37* y la *Figura 5.38* ilustran algunos de estos pasos.

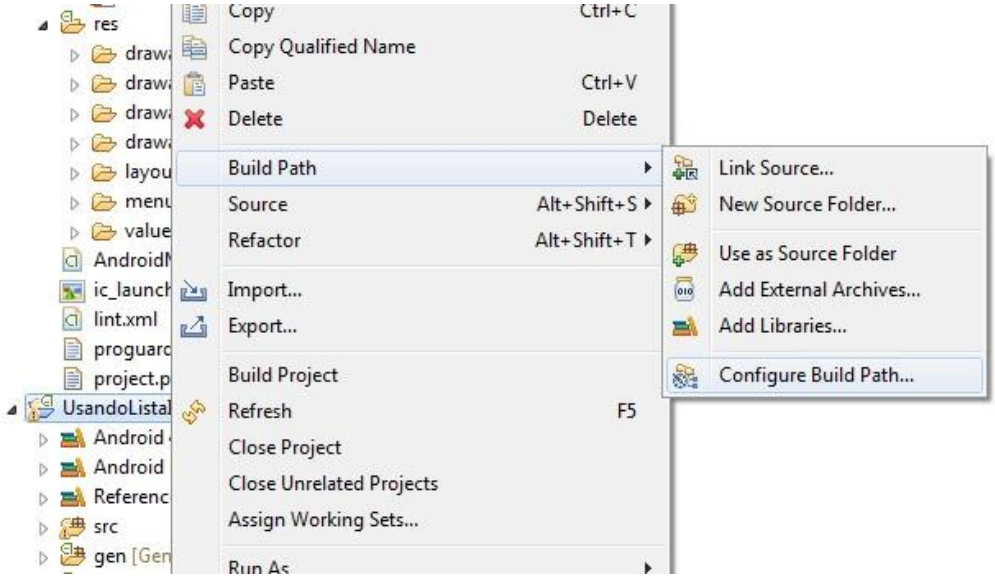


Figura 5.37 .- Paso 2

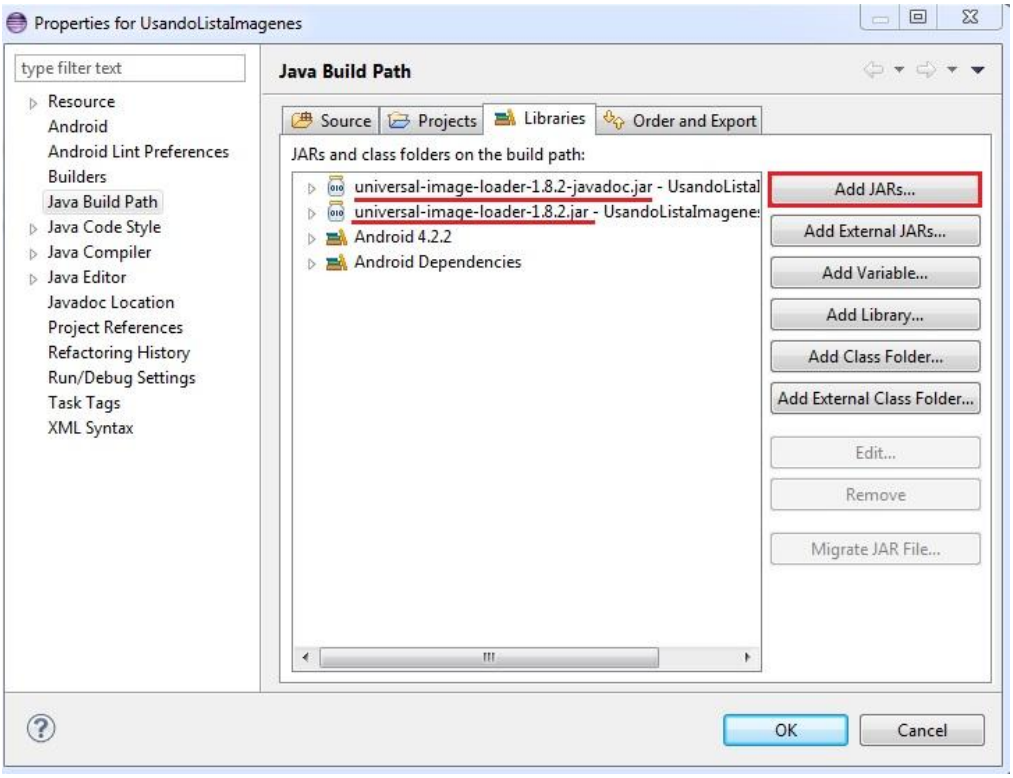


Figura 5.38 .- Paso 3 y Paso 5

Diagramas UML

Antes de particularizar y detallar las clases, se ilustra un diagrama UML de clases para ver cómo se relacionan entre sí las clases de la aplicación y las clases adaptadoras. Las clases que se muestran en la *Figura 5.39* son las involucradas en la galería de imágenes y **adaptadas** del ejemplo que se puede descargar de *Github* [37].

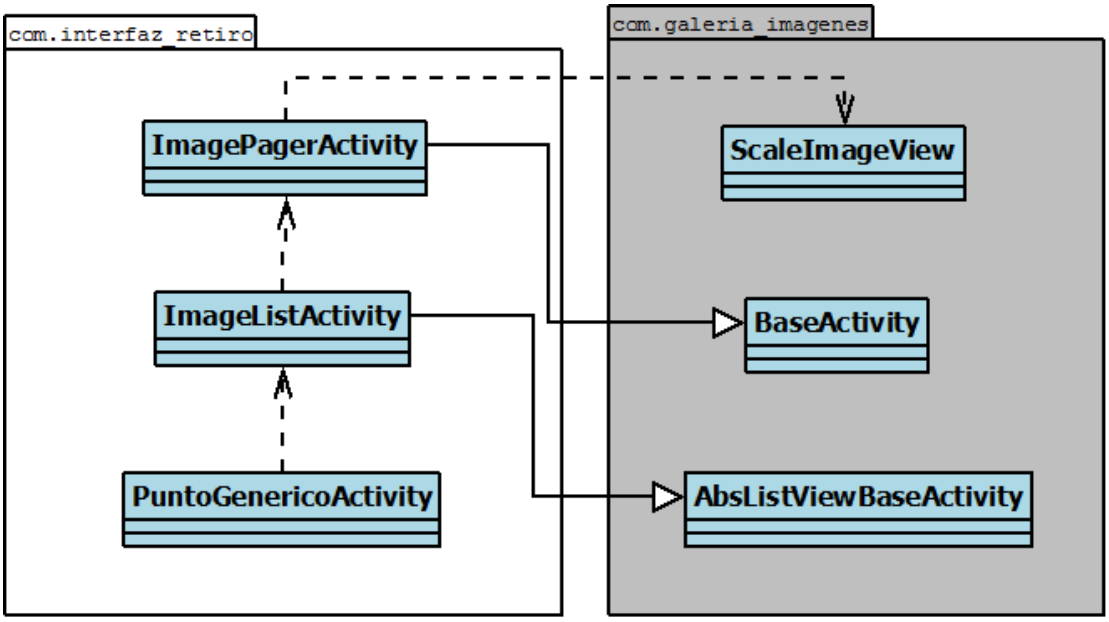


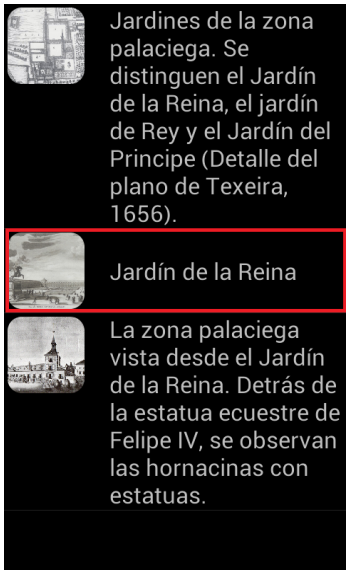
Figura 5.39 .- Diagrama UML de la Galería de Imágenes

Las clases contenidas en el paquete *com.interfaz_retiro*, en la parte izquierda de la *Figura 5.39*, son aquellas que son llamadas por la aplicación como actividades y, por tanto, tienen *layouts* asociados. Las clases contenidas en el paquete *com.galeria_imagenes*, en la parte derecha de la *Figura 5.39*, son las que se han adaptado al proyecto, procedentes del ejemplo de uso de la librería *Universal Image Loader*, entre otras. Se detallan a continuación las dependencias más importantes:

- *PuntoGenéricoActivity* usa la clase *ImageListActivy* debido a que crea el *Intent* de tal clase y le pasa los siguientes argumentos:
    - **String[] urls**: el lugar desde donde se tiene que descargar el recurso. Los recursos vienen ordenados según el orden del id en el que estén almacenados en la base de datos
    - **String[] nombres**: la descripción de cada foto que se va a descargar
- *ImageListActivity* hereda de *AbstListViewBaseActivity* y usa la clase *ImagePagerActivity*, ya que ésta se desplegará tras seleccionar una imagen de la lista, pasándole los siguientes argumentos:

    - **String[] urls**: igual que en el apartado anterior. Esto permite desplegar una imagen que todavía no se ha descargado y que, en segundo plano, continúe descargándose mientras se despliega la imagen en proceso de descarga.
    - **Int page**: indica el número de imagen que se tiene que desplegar según en qué elemento de la lista se haya pulsado
- *ImagePagerActivity* hereda de *BaseActivity* y usa *ScaleImageView* debido a que en lugar de ser un *ImageView* como venía por defecto en la librería, se utiliza la clase que implementa la funcionalidad de *zoom* de una imagen. Para realizar esto, únicamente hay que sustituir en el *layout* de esta clase el tipo de vista que se quiere desplegar, es decir, un *ScaleImageView*. También hay que hacer sus respectivos cambios en el código como se explicará más tarde.

Las clases adaptadas, (todas salvo *PuntoGenericoActivity* y *ScaleImageView*) utilizan por debajo clases de la librería *Universal Image Loader*. El cometido de este apartado es dar breves pinceladas de su adaptación para el uso en nuestra aplicación. A continuación, aparecerán tres ilustraciones de las actividades anteriormente descritas, siendo *la Figura 5.40* la que representa a *PuntoGenericoActivity*, la *Figura 5.41* es *ImageListActivity* y la *Figura 5.42* ilustra la imagen en forma de diapositivas de la actividad *ImagePagerActivity*. El flujo de transición de una pantalla a otra está determinado por el color **rojo**. Cada pantalla ilustrada en las figuras es una *Activity* en *Android*, desde la cual se puede interactuar a base de eventos.



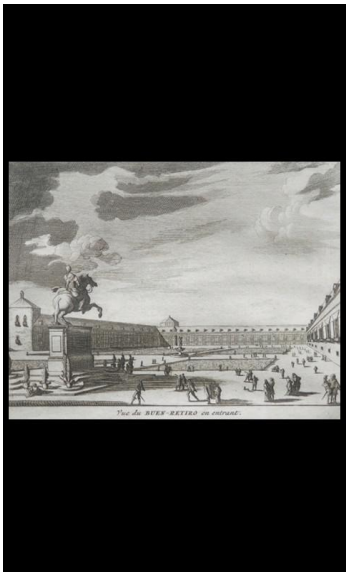


Figura 5.40 ***Figura 5.41*** ***Figura 5.42***

Antes de detallar las clases de adaptación, cabe decir que dentro de cada una de ellas se va a crear (si no se ha creado todavía) un objeto imageLoader, de la clase *ImageLoader* (interna de la librería) que cargará las imágenes del modo en el que se quiera inicializar. Cada una de las clases que adaptan la librería la inicializa de una manera u otra.

ImageListActivity

Esta clase es la encargada de representar (y descargar) las imágenes y mostrarlas en forma de lista, acompañadas de un pequeño icono en miniatura de lo que será su ilustración futura y de la descripción propia de la foto como se puede ver en la *Figura 5.41*. De esta clase, destacamos el método *onCreate* y la declaración de la clase interna *ItemAdapter* (que hereda de *BaseAdapter*) para explicar el funcionamiento principal de la actividad.

onCreate()

Ajusta el contenido a un *layout* que se define por un sólo *ListView* identificado como list. El procedimiento empieza recuperando los atributos (URI y descripción –nombres-) comentados que vienen en contexto desde el *Intent* que le ha llamado, es decir, la actividad *PuntoGenericoActivity* (*Figura 5.43*).

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

    page = 0;
    Bundle bundle = getIntent().getExtras();
    urls = bundle.getStringArray("urls");
    nombres = bundle.getStringArray("nombres");

    options = new DisplayImageOptions.Builder()
        .showStubImage(R.drawable.ic_stub)
        .showImageForEmptyUri(R.drawable.ic_empty)
        .showImageOnFail(R.drawable.ic_error)
        .cacheInMemory()
        .cacheOnDisc()
        .displayer(new RoundedBitmapDisplayer(20))
        .build();

    listView = (ListView) findViewById(android.R.id.list);
    ((ListView) listView).setAdapter(new ItemAdapter());
    listView.setOnItemClickListener(new OnItemClickListener() {
        public void onItemClick(AdapterView<?> parent, View view, int position, long id) {
            startImagePagerActivity(position);
        }
    });
}
```

Figura 5.43 .- Método onCreate(Bundle savedInstanceState) de ImageListActivity.java

Después de haber recogido las URIs y las descripciones, se ajusta las opciones del futuro *ImageLoader* (cargador de imágenes), con los siguientes valores:

- **stub:** muestra una imagen de *Android* mientras está descargando una imagen de Internet
- **forEmptyUri:** muestra un icono cuando la URI está vacía
- **onFail:** muestra un aspa rojo cuando el recurso no está disponible
- **cacheInMemory:** cachea en memoria la imagen pequeña de la foto descargada porque ocupa poco
- **cacheOnDisc:** guarda en la memoria SD el recurso descargado para poderlo consultar posteriormente sin necesidad de Internet

El método *onCreate()* posteriormente creará un objeto que hace de adaptador para averiguar qué ítem de la lista se ha pulsado, incorporando un oyente, para, posteriormente, dejar paso a la actividad *ImagePagerActivity* con la imagen específica seleccionada.

La clase ItemAdapter interna a ImageList

Esta clase, que hereda de *BaseAdapter*, tiene un oyente que actualiza las imágenes en caso de que terminen de descargarse. Además, calcula las imágenes que hay y procede a descargarlas en el método *getView*. También incorpora las descripciones asociadas a cada imagen, ya que contiene un propio *TextView* e *ImageView* incorporado dentro de cada elemento de la lista. Crea el objeto de *ImageLoader* y lo inicializa con la configuración guardada en *options*. Los cambios más significativos realizados en la plantilla del ejemplo de *Github* [37] vienen marcados en **rojo** en el código de la *Figura 5.44*.

```
class ItemAdapter extends BaseAdapter {

    private ImageLoadingListener animateFirstListener = new AnimateFirstDisplayListener();

    private class ViewHolder {
        public TextView text;
        public ImageView image;
    }

    public int getCount() {
        return urls.length;
    }

    public Object getItem(int position) {
        return position;
    }

    public long getItemId(int position) {
        return position;
    }

    public View getView(final int position, View convertView, ViewGroup parent) {
        View view = convertView;
        final ViewHolder holder;
        if (convertView == null) {
            view = getLayoutInflater().inflate(R.layout.item_list_image, parent, false);
            holder = new ViewHolder();
            holder.text = (TextView) view.findViewById(R.id.text);
            holder.image = (ImageView) view.findViewById(R.id.image);
            view.setTag(holder);
        } else {
            holder = (ViewHolder) view.getTag();
        }

        holder.text.setText(nombres[position]);
        ImageLoaderConfiguration config = ImageLoaderConfiguration.createDefault(ImageListActivity.this);

        imageLoader.init(config);
        imageLoader.displayImage(urls[position], holder.image, options, animateFirstListener);

        return view;
    }
}
```

Figura 5.44 .- Adaptación de la clase interna ItemAdapter de ImageListActivity

ImagePagerActivity

Esta clase hereda de la clase *BaseActivity* lo cual le permite, por herencia, borrar las imágenes almacenadas en la memoria SD y en la propia caché de la memoria. Pero además, se ha adaptado esta clase para que en lugar de un *ImageView* fuese un *ScaleImageView* que se explicó en el apartado de la clase *ScaleImageView*. Se recuerda que los pasos a seguir para sustituir un *ImageView* son de la siguiente manera: se coloca en el *layout* un *ScaleImageView* (sustituyendo al *ImageView*) según ilustra en la *Figura 5.45*:

```
<com.galeria_imagenes.ScaleImageView
    android:id="@+id/image"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:layout_gravity="center"
    android:layout_margin="5dp"
    android:adjustViewBounds="true" />
```

Figura 5.45 .- Modo de sustituir un ImageView por un ScaleImageView en un layout

Y posteriormente, dentro del código, donde se despliega la imagen a pantalla completa, se instanciará un *ScaleImageView*. Esto nos permitirá disponer de *zoom* en todas las imágenes que despleguemos a través de la actividad *ImagePagerActivity*. En la *Figura 5.46*, se muestra el código del método *instantiateItem(...)* que es aquel que realiza este paso.

Se debe comentar que existe gestión de errores en caso de que haya problemas con el servidor o que un recurso esté corrupto a la hora de ser descargado. Además, para esta actividad en exclusividad, se ha dejado la libertad al usuario de, con el giroscopio activado en su dispositivo, girarlo para ver las imágenes en horizontal y que se acople mejor a la pantalla (si fueron diseñadas para que así se vieran). Para que esta opción se habilite hay que excluir en el *AndroidManifest.xml* la propiedad *android:screenOrientation=”portrait”* de esta actividad, ya que en el resto de actividades se tomó la decisión de sólo permitir la inclinación vertical. Se ilustra un claro ejemplo en la *Figura 5.47*.

```java
@Override
public Object instantiateItem(ViewGroup view, int position) {
    View imageLayout = inflater.inflate(R.layout.item_pager_image, view, false);
    ScaleImageView scImageView = (ScaleImageView) imageLayout.findViewById(R.id.image);
    scImageView.setScaling(true);
    final ProgressBar spinner = (ProgressBar) imageLayout.findViewById(R.id.loading);

    imageLoader.displayImage(images[position], scImageView, options, new SimpleImageLoadingListener() {
        @Override
        public void onLoadingStarted(String imageUri, View view) {
            spinner.setVisibility(View.VISIBLE);
        }

        @Override
        public void onLoadingFailed(String imageUri, View view, FailReason failReason) {
            String message = null;
            switch (failReason.getType()) {
                case IO_ERROR:
                    message = "Error en el dispositivo";
                    break;
                case DECODING_ERROR:
                    message = "Error al procesar la imagen";
                    break;
                case NETWORK_DENIED:
                    message = "La descarga desde el servidor ha sido denegada";
                    break;
                case OUT_OF_MEMORY:
                    message = "Memoria insuficiente";
                    break;
                case UNKNOWN:
                    message = "Error desconocido";
                    break;
            }
            Toast.makeText(ImagePagerActivity.this, message, Toast.LENGTH_SHORT).show();

            spinner.setVisibility(View.GONE);
        }

        @Override
        public void onLoadingComplete(String imageUri, View view, Bitmap loadedImage) {
            spinner.setVisibility(View.GONE);
        }
    });

    ((ViewPager) view).addView(imageLayout, 0);
    return imageLayout;
}
```

Figura 5.46 .- Cómo instanciar una imagen con zoom en la clase ImagePagerActivity

```
<activity android:name=".ImagePagerActivity" />
<!-- android:screenOrientation="portrait" -->
```

Figura 5.47 .- Eliminación del modo retrato de una actividad en el Manifest (comentado)

La opción de desplazamiento en modo diapositiva viene dado por el propio *ImageLoader* dependiente de esta librería. Algunos ejemplos de lo aquí comentado se encuentran a continuación.

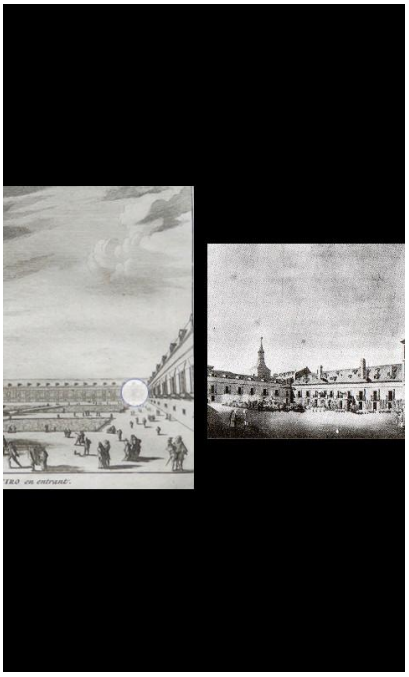


Figura 5.48

Figura 5.49

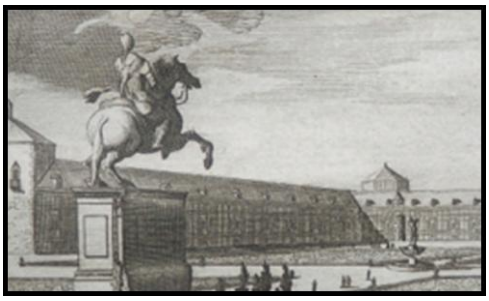


Figura 5.50

En la *Figura 5.48*, que está a la izquierda, podemos ver el evento de deslizamiento de la actividad *ImagePagerActivity* (en esta foto se está deslizando de derecha a izquierda), en el que aparecerá otra imagen como si fuera una diapositiva. En la pantalla de arriba a la derecha, de la *Figura 5.49* podemos observar la nueva imagen que llega como consecuencia del deslizamiento, además, adaptada en formato horizontal (procedente del modo vertical). Para terminar, se hace *zoom* en la foto como viene reflejada en la *Figura 5.50*.

Como se expuso al principio, si se da la opción de escribir en la memoria SD en la configuración del objeto *ImageLoader*, también garantiza la posibilidad de borrar esto que hemos descargado para posteriormente eliminarlo del dispositivo. Esto se desarrolla en la clase abstracta BaseActivity (que extiende la clase *ImagePagerActivity*) de una manera muy intuitiva. Cuando **pulsamos el menú** de una actividad *ImagePageractivity* hay dos opciones:

- Eliminar la memoria cacheada hasta el momento para liberar recursos
    - Para realizar esto se ejecuta el siguiente método del objeto *imageLoader*: ***imageLoader.clearMemoryCache()***
- Eliminar la memoria SD del dispositivo.
    - Esto se consigue Ejecuta el siguiente método del objeto imageLoader: ***imageLoader.clearDiscCache()***

Tras la ejecución de este último método, las imágenes volverán a ser descargadas de Internet. Seleccionando una de las dos opciones, o las dos si así se desea, se ejecutarán tales métodos. En la *Figura 5.51* se ofrece un ejemplo gráfico:

Figura 5.51 .- Pantalla que muestra dos opcionas para limpiar memoria

La clase MapFragment (Google Maps)

La clase *FragmentMap* ha sido creada por *Google* a finales del 2012. Vio la luz junto con la versión 2 de la API de *Google Maps*. En un principio, la intención de los desarrolladores era usar la versión 1, ya que al comenzar a fraguarse el proyecto era la vigente. Tras la salida de la nueva versión y un profundo estudio de campo, se llegó a la conclusión de que, pese a retroceder en el desarrollo, a la larga iba a ser la opción con más ventajas.

Una de las primeras cosas que el equipo tuvo que hacer, fue añadir las librerías pertinentes, puesto que la clase *Fragment* en general y *MapFragment* en particular están diseñadas para la versión 4 de *Android* . Como la versión pensada desde un principio fue la 2.3, el uso de la librería android support v4 se hace necesaria para la utilización de mapas, así como los servicios de *Google Play*. La *Figura 5.52* muestra cómo se crean dependencias con estas bibliotecas.

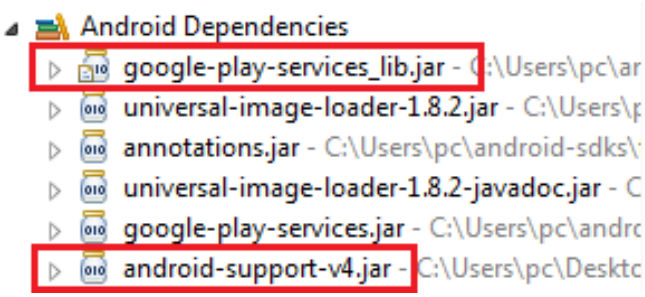


Figura 5.52 .- Uso de las librerías para MapFragment

Las ventajas que ofrece esta API frente a su predecesora son:

- Ahora los mapas se comportan como *Fragments*, lo que puede permitir la aparición de más de un mapa en una misma pantalla.
- Utilización de *mapas vectoriales*, lo que repercute en una mayor velocidad de carga y una mayor eficiencia en cuanto al uso de ancho de banda.
- Mejoras en el sistema de caché, lo que reducirá en gran medida las famosas áreas en blanco que tardan en cargar.
- Los mapas son ahora 3D, es decir, podremos mover nuestro punto de vista de forma que lo veamos en perspectiva.

Entrando en materia más técnica, los pasos necesarios para integrar esta librería al proyecto comienzan por la creación de una "*API key"* que será un código de desarrollador para que *Google* tenga controlado el número de accesos.

Para la obtención de la "*API key*" será necesario visitar la consola de API's de *Google*, en este enlace: https://code.google.com/apis/console/?pli=1#project:462708371760. Una vez dentro, tendremos que dar el nombre de nuestra aplicación (*Figura 5.53*).

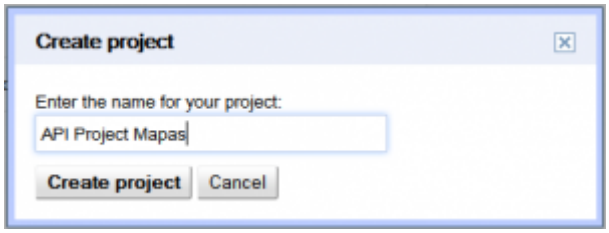


Figura 5.53 .-Dar nombre a nuestra aplicación a través de la consola de API's de Google [61].

Después, habrá que hacer *click* sobre la pestaña *services* que se encuentra en el panel de la izquierda (*Figura 5.54*).

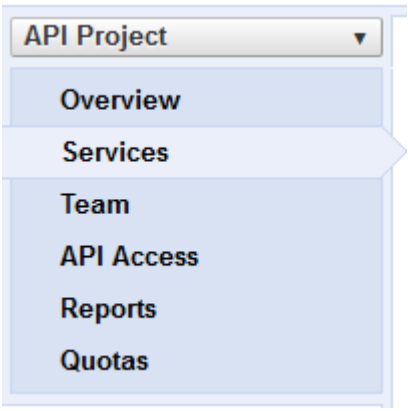


Figura 5.54 .-Pestaña de servicios. [61]

A continuación se activará el servicio de *Google Maps* (*Figura 5.55*).

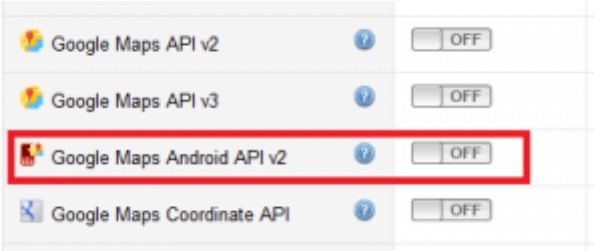


Figura 5.55 .- Servicio de Google Maps que hay que activar [61].

Posteriormente, marcando la pestaña de API Access, podremos solicitar una "*API key*" pulsando en el botón correspondiente (*Figura 5.56*):

Figura 5.56 .- Botón donde solicitar la "API key" [61].

Sólo queda proporcionarle a la consola el nombre del paquete en el que va a ser introducida la API y la huella digital con la que se firmará la aplicación (SHA1).

Para hallar la huella digital hay que averiguar dónde se encuentra el "certificado de pruebas de nuestro proyecto (En *Preferences*, dentro de Eclipse). Se puede apreciar con claridad en la *Figura 5.57*:

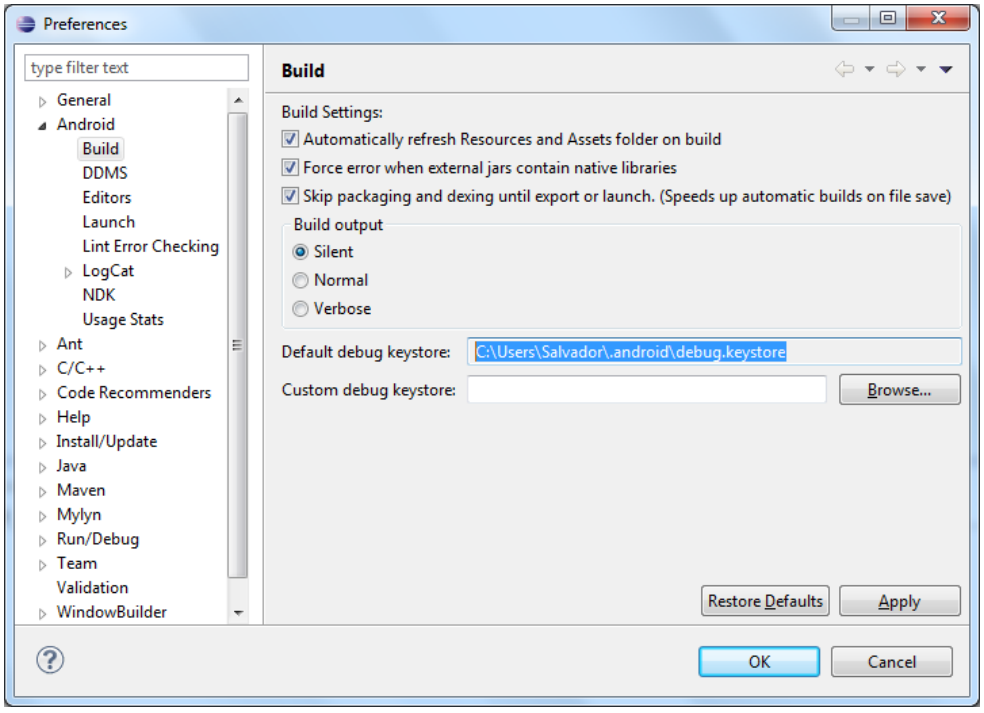


Figura 5.57 .- Obtención de la ubicación del certificado de pruebas a través de Eclipse[61].

Una vez hallado el directorio, se accederá a él a través del símbolo del sistema en Windows y habrá que introducir la siguiente directiva (Figura 5.58):

```
   C:\Users\Salvador\.android>"C:\Program
Files\Java\jdk1.7.0_07\bin\keytool.exe" -list -v -keystore debug.keystore
-alias androiddebugkey -storepass android -keypass android
```

Figura 5.58 .- Directivas necesarias para conocer la huella digital de la aplicación [61]

Donde C:\Users…\.android es la ruta en la que se encuentra el certificado de pruebas y C:\Program\Files\...\keytool.exe la ruta donde se encuentra la "*java virtual machine*". Introduciendo esta directiva, nos aparecerá la huella digital (Figura 5.59):

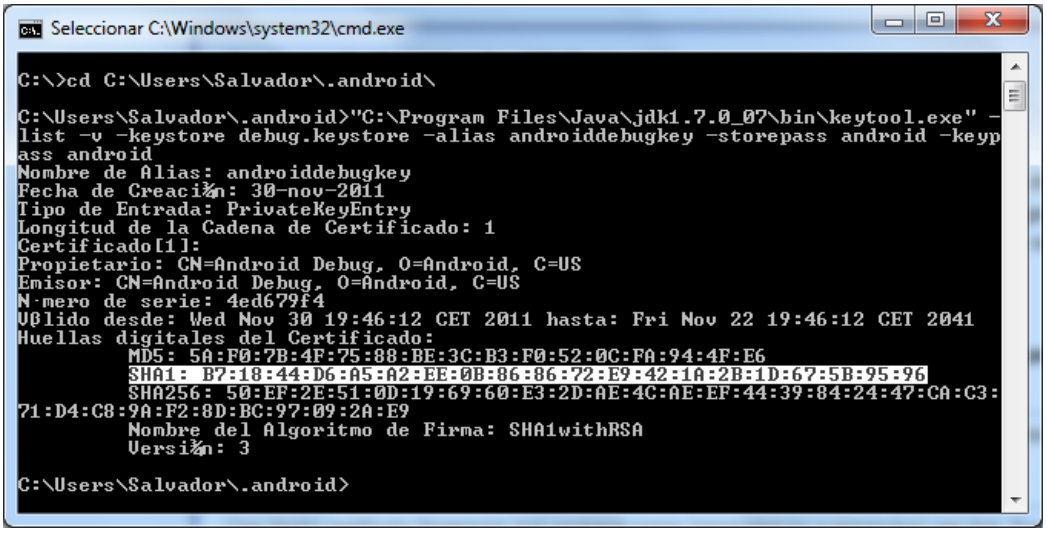


Figura 5.59 .- Huella digital de la aplicación [61].

Con esto ya se estaría en condiciones de solicitar la "*API key*" . Google la proporcionará y habrá que incluirla en el *AndroidManifest.xml* con la siguiente porción de código (*Figura 5.60*):

```
...
<application>
...
    <meta-data android:name="com.google.android.maps.v2.API_KEY"
               android:value="api_key"/>
...
</application>
```

Figura 5.60 .- Código necesario para incluir la "API key". api_key deberá sustituirse por el valor obtenido[61].

Una vez se tiene la "*API key"*, se han importado las librerías necesarias y se han aportado los permisos necesarios en el *Android Manifest*, hay que definir en el archivo xml de la pantalla en la que va a aparecer el MapFragment. Para ello se utiliza el código de la *Figura 5.61*:

```
<fragment
        android:id="@+id/map"
        android:layout_width="match_parent"
        android:layout_height="500dp"
        class="com.google.android.gms.maps.SupportMapFragment"/>
```

Figura 5.61 .- Código XML para la introducción de mapas

Con esto ya tendríamos el mapa incluido en la interfaz. Sólo queda darle funcionalidad. Para ello basta con modificar la clase asociada a ese archivo xml. Solo hay que poner la siguiente línea de código para tener acceso pleno al mapa:

```
mapa = ((SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map)).getMap();
```

Con eso ya se podrán utilizar todos los métodos que proporciona la API de *Google* sobre la variable mapa. Los métodos usados sobre el mapa en este proyecto son los siguientes:

- setMapType(): Selecciona la vista del mapa, en nuestro caso satélite.
- moveCamera(): Mueve la cámara del mapa a un punto dado.
- Clear(): Borra cualquier elemento dibujado sobre el mapa
- addMarker(): Añade un marcador de posición que podrá tener un texto asociado
- addPolyLine(): Dibuja sobre el mapa un conjunto de líneas con los puntos determinados previamente.

Además, la API provee al usuario de muchos otros métodos de distinto tipo para diferentes usos, como registrar el punto donde el usuario ha pulsado, dibujar formas geométricas variadas, registrar eventos, etc.

Por último cabe destacar el conjunto de operaciones que el usuario puede realizar con el mapa: Acercar, alejar, desplazar, girar, cambiar perspectiva, etc.

En la *Figura 5.62* se pueden apreciar algunos de los componentes anteriormente explicados, como los marcadores (en **rojo**) que indican una posición o las poli-líneas (en **azul**) que unen dos puntos.

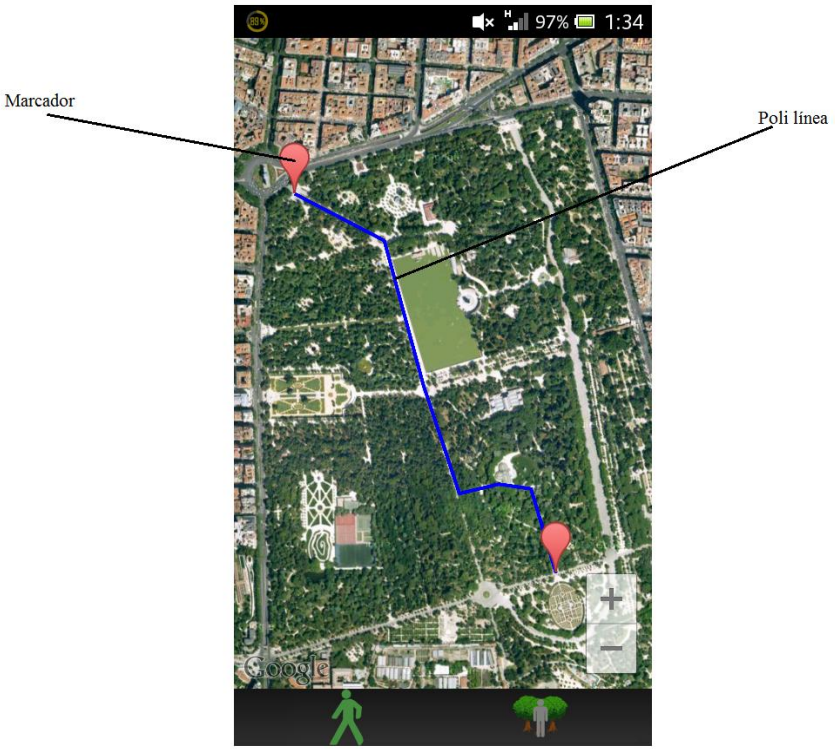


Figura 5.62 .- Ejemplo del uso de la clase MapFragment

Incidencias de riesgo del desarrollo de la aplicación

En este apartado queremos enunciar todos y cada uno de los problemas que han surgido a la hora de llevar a cabo este proyecto. Se enumerarán de manera breve:

- Cambio de personal por parte del Ayuntamiento de Madrid, lo que tiene como consecuencia
    - Cambio drástico de la temática del proyecto
    - Cambio de requisitos exigidos por el cliente
- Cambio de versión de la API de *Google Maps*
- Mal diseño de caminos en el mapa del Retiro por parte de *Google Maps*

Capítulo 6 - Técnicas algorítmicas

Debido al conocimiento tecnológico requerido en el desarrollo de una aplicación *Android* no es de extrañar que en mayor o menor medida hayamos empleado parte de los contenidos y técnicas adquiridos en la licenciatura de Ingeniería en Informática. Las técnicas más importantes y cruciales que hemos utilizado han sido principalmente aquellas relacionadas con la optimización de caminos mínimos en grafos, el tratamiento del almacenamiento de datos por servidores, el uso y aplicación de conceptos relacionados con la computación y el control de estados y la aplicación de conceptos relacionados con las redes. Se ha empleado la optimización de caminos mínimos en grafos para dar solución a un problema de geolocalización presente en el parque del Buen Retiro. A través del algoritmo de Dijkstra hemos obtenido una solución a dicho problema. Otra de las aplicaciones interesantes ha sido la máquina de estado que se ha desarrollado para el funcionamiento ordenado de la reproducción de la información locutada, en estados de reproducción, pausa, reanudación... Asimismo para el control de esos audios se han empleado conocimientos relacionados con las colas de mensajes. Cada una de las técnicas se describen detalladamente en los subapartados siguientes.

Algoritmo de camino óptimo: Dijkstra

La idea de utilizar este algoritmo vino cuando el proyecto ya se había empezado. Tras un tiempo de investigación sobre la API y el uso de *Google Maps*, el equipo descubrió, que sería imposible mostrar los caminos dentro del Retiro, dado que éstos aún se encuentran en fase beta y pueden dar soluciones insatisfactorias.

Esencialmente, al calcular la ruta desde un punto a otro, *Google Maps* tomaba algunos de los caminos del Retiro como autopistas, por lo que las rutas algunas veces carecían de sentido. Un ejemplo de este fallo que no se puede permitir se muestra en la *Figura 6.1*, en la que se ve cómo para llegar de un punto a otro cercano, *Google* nos traza una ruta fuera de lugar:

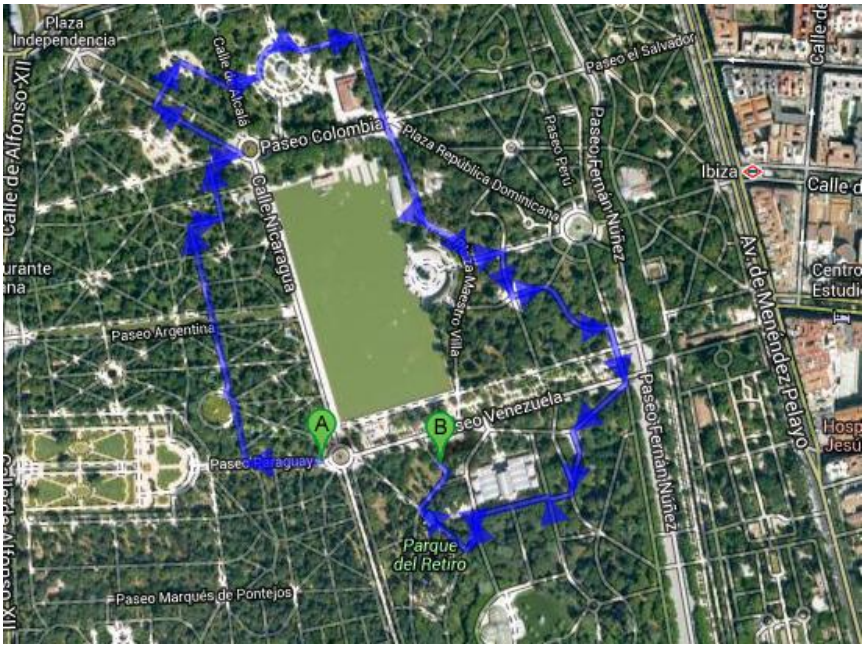


Figura 6.1 .- Ruta trazada por google maps sin sentido.

La primera solución que el equipo pensó fue trazar una línea recta entre origen y el destino, pero debido a su simpleza e inexactitud se consideró insuficiente, así que hubo que rediseñar ésta parte de la aplicación.

El *algoritmo de Dijkstra* resultó ser la solución más adecuada para lo que el equipo estaba buscando, ya que se tratará siempre del mismo grafo y lo que interesa es saber el camino mínimo desde cualquier punto de éste. El algoritmo nos facilitará el camino mínimo desde un punto a todos los demás dentro del grafo, por lo tanto bastaría con ejecutar dicho algoritmo una vez por cada punto y, de este modo, obtener el resultado buscado.

Para realizar esto, en primer lugar, el equipo tuvo que diseñar un grafo del retiro que cubriese la mayoría de posibilidades usando vértices tanto en las entradas como en las intersecciones más importantes (*Figura 6.2*). Tampoco era conveniente hacer un grafo demasiado grande ya que la eficiencia de la aplicación se vería resentida.

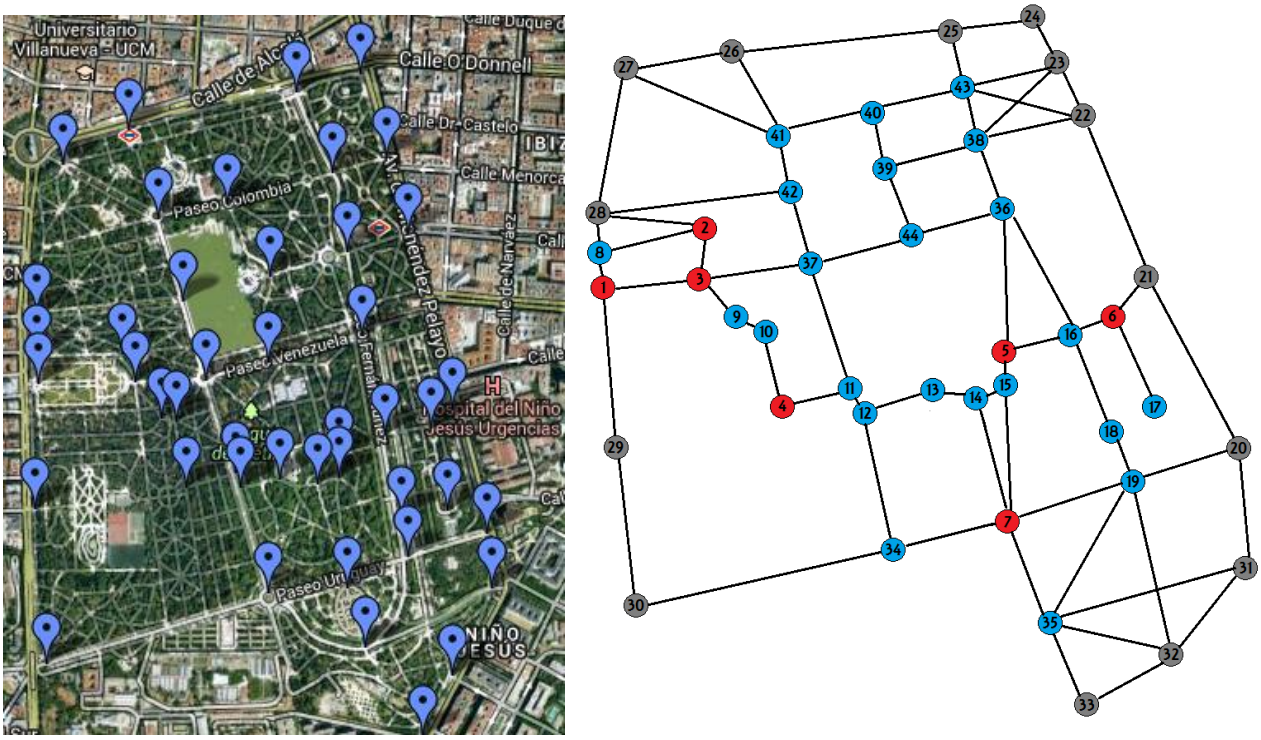


Figura 6.2 .- Conjunto de puntos seleccionados (izquierda) y grafo construido (derecha)

Una vez definido el grafo, el siguiente paso fue implementar una aplicación que escriba en un fichero el conjunto de todos los caminos mínimos posibles dentro del grafo. Para ello, pasaremos un fichero de texto al programa con el grafo expresado en el formato representado en la *Figura 6.3*. Cada fila representa un punto y las columnas son (de izquierda a derecha), nombre del punto, latitud, longitud y cada uno de los puntos adyacentes.

```
1 40.4153 -3.6886 3 8 29
2 40.4159 -3.6864 3 8 28
3 40.4153 -3.686 1 2 9 37
4 40.4132 -3.6847 10 11
5 40.4139 -3.6808 15 16 36
6 40.4144 -3.6783 16 17 21
7 40.4115 -3.6806 14 19 34 35
8 40.4159 -3.6886 1 2 28
9 40.4146 -3.6853 3 10
10 40.4145 -3.685 4 9
11 40.4135 -3.6835 4 12 37
12 40.4132 -3.6833 11 13 34
13 40.4132 -3.6833 12 14
14 40.4133 -3.6813 7 13 15
15 40.4135 -3.6807 5 14
16 40.4143 -3.6795 5 6 18 36
17 40.4127 -3.6779 6
18 40.4127 -3.6792 16 19
19 40.4118 -3.6789 7 18 20 35
20 40.4123 -3.6769 19 21 31
21 40.4148 -3.6777 6 20 22
22 40.4183 -3.679 21 23 38 43
23 40.4198 -3.6795 22 24 38 43
24 40.4215 -3.6802 23 25
```

Figura 6.3 .- Fichero que codifica el grafo.

Para la implementación se contará con una estructura de datos auxiliar extraída del paquete *java.lang*. Su nombre es *PriorityQueue* y se trata de un tipo especial de cola en el que al insertar los elementos, éstos lo harán en el orden que el programador determine. En este caso el orden será de menor a mayor teniendo en cuenta el valor de la arista.

El algoritmo no es complejo. Se trata de, partiendo de un punto de origen, hallar la distancia mínima a cada uno de los puntos. Para ello cada punto tendrá una distancia mínima dinámica que se irá actualizando a medida que el algoritmo sigue su curso. Inicialmente tomaremos esa distancia mínima como infinita para calcular el mínimo con éxito. Además, los vértices se irán marcando según se vayan evaluando, así que, al principio no hay ninguno marcado. Lo que hará el algoritmo es recorrer todos los vértices adyacentes al actual si ni están marcados (al empezar, será el nodo de origen) y actualizar, si es preciso, las distancias. Una vez se han recorrido todos los vértices adyacentes, el vértice se marca y pasa a evaluarse el más cercano.

Por último, el algoritmo implementado realizará los siguientes pasos:

1. Se crea una cola de prioridad vacía.
2. Se establece como punto de origen (actual) el que se da como argumento
3. Como es el punto inicial, la distancia a sí mismo será 0, luego actualizamos el campo *distMin* (distancia mínima a ese punto desde el origen) a dicho valor.
4. Se inicia un bucle en el que se examinarán todos los vértices (mientras no se hayan evaluado todos los vértices…).
5. Se toma el vértice adyacente que corresponde en cada momento (no es necesario hacerlo en orden).
6. Se evalúa la condición siguiente: Si el valor de la arista al vértice adyacente sumado al valor del campo *distMin* del vértice considerado como actual es menor que el campo *distMin* del vértice adyacente, entones se actualiza el valor *distMin* del mismo y se añade a la cola de prioridad. Esto significa que el camino hallado (o no) desde algún vértice ya marcado es más largo que el actual, por tanto no era la mejor solución. Un ejemplo de esto puede apreciarse en la *Figura 6.4*.

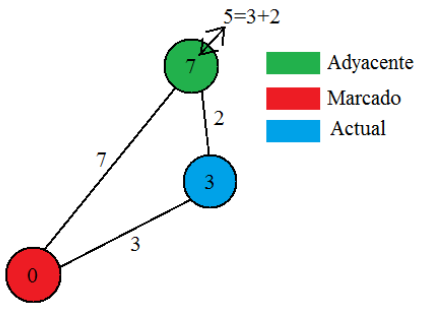


Figura 6.4 .- Muestra un ejemplo de actualización del valor distMin.

7. Volver al punto 4 y hacer lo mismo con cada uno de los vértices adyacentes.
8. Una vez se han evaluado todos los vértices adyacentes, el que es actual se marca, con lo que no volverá a ser evaluado.
9. Se extrae el primer elemento de la cola de prioridad que será el más cercano al origen detrás del que acaba de ser actual.
10. Se repite el proceso hasta marcar todos los vértices.

El código, teniendo la implementación del grafo que el equipo ha determinado que sería mejor, es el que se muestra en la *Figura 6.5*.

```
public void Dijkstra(int partida)
              {
               int marcados=0;
               PriorityQueue<verticeGrafo> cola=new PriorityQueue<verticeGrafo>();
               verticeGrafo actual= értices[partida-1];
               actual.setDistmin(0);
               while (marcados < vertices.length)
                     {
                      for(int i=0;i<actual.getAdyacentes().length;i++)
                            {
                             double
distancia=actual.getDistadyacentes().get(actual.getAdyacentes()[i]);

if(actual.getDistmin()+distancia<vertices[Integer.valueOf(actual.getAdyacentes()[i])-
1].getDistmin())
                                        {

vertices[Integer.valueOf(actual.getAdyacentes()[i])-
1].setDistmin(actual.getDistmin()+distancia);

 értices[Integer.valueOf(actual.getAdyacentes()[i])-
1].setPredecesor(actual.getNombre());

cola.add( értices[Integer.valueOf(actual.getAdyacentes()[i])-1]);
                                        }
                                  }
                      actual.marcado=true;
                       értices[Integer.valueOf(actual.getNombre())-1].marcado=true;
                      marcados++;
                      if(marcados< értices.length)
                             actual=cola.remove();
                     }
              }
```

Figura 6.5. Código de la implementación del algoritmo de Dijkstra.

Para clarificar cualquier duda que pudiese surgir sobre este algoritmo, en la *Figura 6.6* se muestra un ejemplo del funcionamiento del mismo.

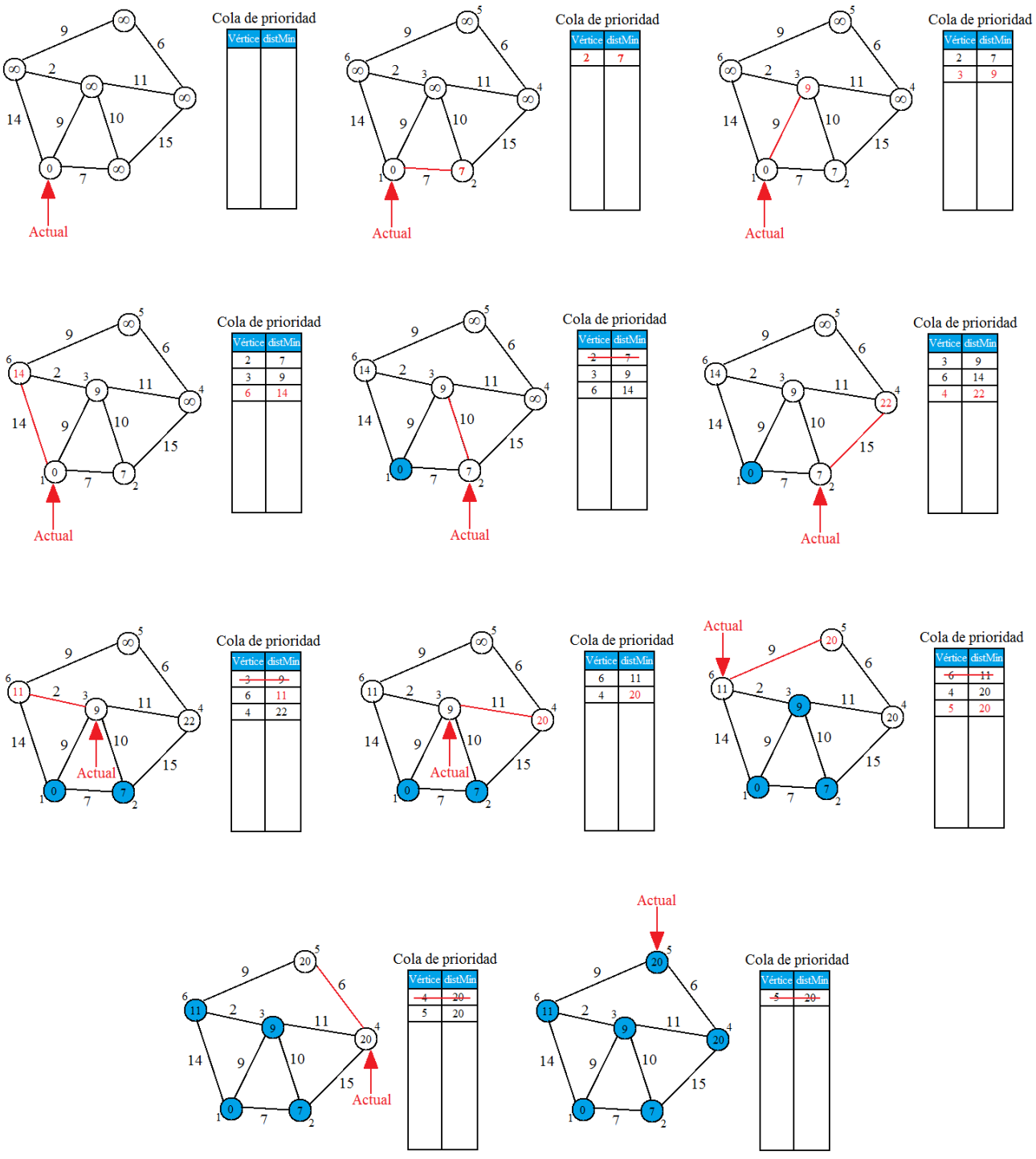


Figura 6.6. Ejemplo de ejecución del algoritmo sobre un grafo

Preselección de un servidor en base a la disponibilidad de un recurso

En ocasiones, es posible que un servidor deje de ofrecer servicios y se tenga que actuar y proponer una solución. Debido a que se posee diferentes servidores de *hosting* gratuito, lo más inmediato es la selección de otro servidor que contenga el recurso y descargarlo de ése. Si ningún servidor lo posee o todos están caídos, se intentará descargará del servidor por defecto.

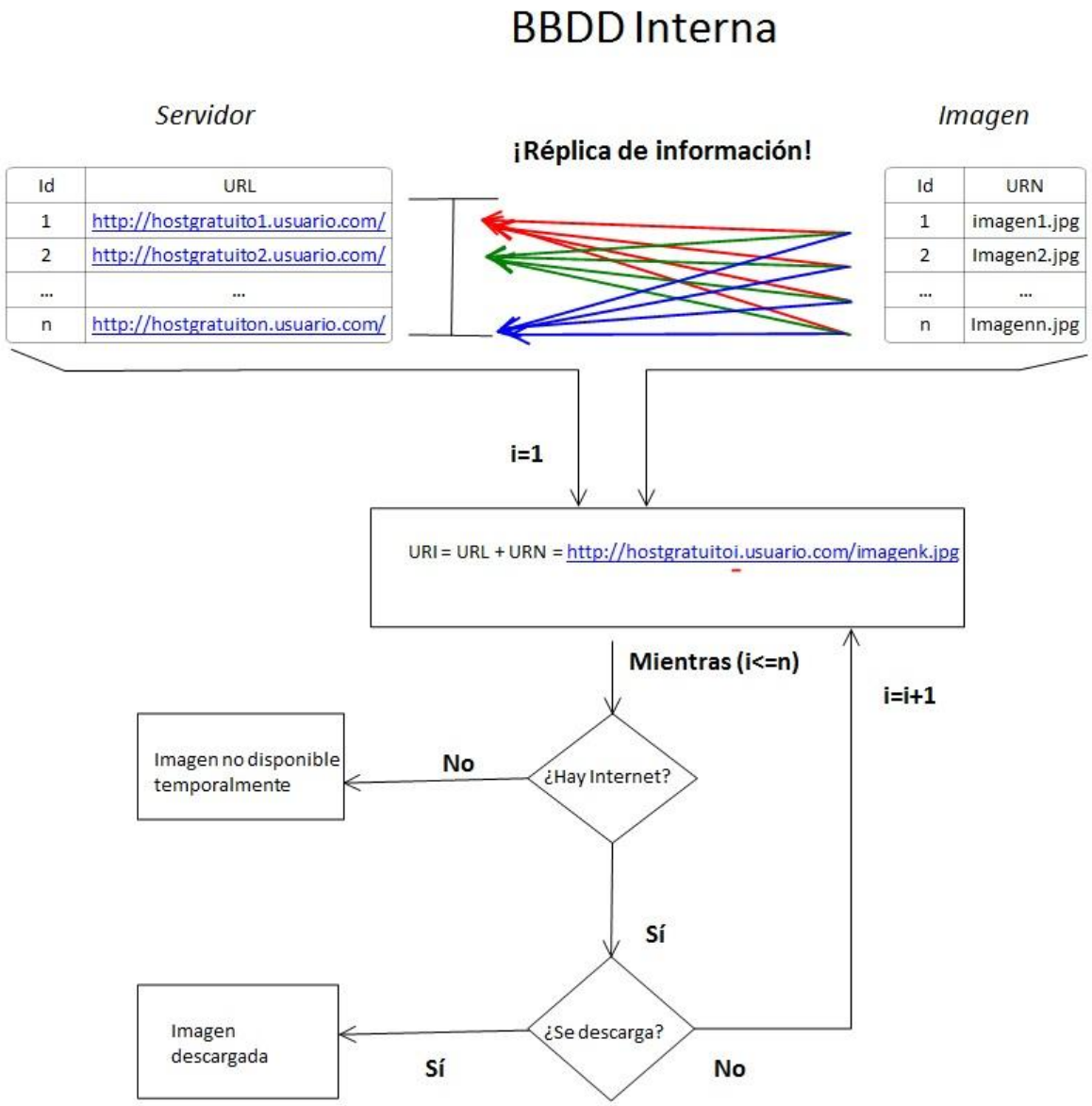


Figura 6.7 .- Algoritmo de selección de servidor con imágenes

Algoritmo de selección de un servidor

La idea principal es la réplica de los datos en cada servidor, su transferencia mediante FTP es muy sencilla y no requiere mucha configuración. Aprovechándose de este *hosting* gratuito (normalmente temporal), se puede acceder a los recursos bajo unas prestaciones medias que pueden dejar de funcionar. Esto deja unas serie de problemática que se ha estado resolviendo a lo largo del proyecto, pero sin una manutención o un alojamiento privado, podría hacer que no se pudiese descargar ningún recurso. Esta problemática se resuelve mediante el algoritmo de la *Figura 6.7*, del cual destacaremos algunas ventajas y desventajas:

- **Ventajas:**
    - El usuario no tiene conocimiento del servidor al que se accede
    - El usuario puede elegir si esperar a la elección del mejor servidor que le otorga el recurso o ir al primero de todos y no esperar
    - El *hosting* gratuito tiene coste cero y el mantenimiento suele ser normal
    - Si un servidor se cae, el usuario si espera a la preselección del recurso podrá disponer de él
- **Desventajas:**
    - Actualmente no hay un proceso que avise al usuario de esta precarga, aunque se hace por debajo. Esto es debido a:
        - El ocultamiento premeditado de los *hosts* a los que se accede
        - Idea para proyectos futuros de coste cero
    - Cuantos más servidores dejen de tener el recurso la selección más lenta será. Esto implica una 'repoblación' de servidores en la tabla de la base de datos según se vayan quedando inoperativos

Posicionamiento en el código

Las principales consultas a la base de datos SQLite, con el propósito de recuperar la información de la tabla de servidores y de las tablas de recursos, se realizan en la actividad principal del programa (*MainActivity.java*) y en cada punto de la aplicación (*PuntoGenericoActivity.java*). En sendas clases se tendrá que declarar una clase privada que herede de la clase *AsyncTask* propia del sistema operativo *Android*. Esto es necesario debido a que si se realiza una tarea que dependa de factores externos (como una consulta por internet) dentro del *thread* principal de la *Activity*,

se procederá a su bloqueo en desarrollo y lanzará una excepción la actividad afectada. Puede encontrar más información acerca de este problema aquí [56] y aquí [57]. La clase que extienda la clase *AsyncTask* se convertirá en una tarea asíncrona por sí sola. Puede encontrar más información [58] y, en resumen, tiene estas características:

- Cualquier clase que la extienda permitirá que sea tratado como otro hilo aparte justo cuando se cree el objeto
- Es ideal para tareas en segundo plano que duran pocos segundos
- Podemos controlar la clase que la extienda mediante los siguientes métodos:
    - *doInBackground(...)*: este método es el principal de esta clase. Es ideal para realizar toda la ejecución de la tarea
    - *onProgressUpdate (...*): este método se llama mientras esté en ejecución esta clase extendida de *AsyncTask*, que sirve de gran utilidad para cambiar porcentajes y tareas no muy exigentes
    - *onPreExecute(), onPostExecute()*: métodos que se ejecutan antes y después de la llamada a *doInBackground(...).*

La clase que extiende a *AsyncTask* está declarada en *MainActivity.java* y en *PuntoGenericoActivity.java* y, en este último caso, se ha llamado *SearchAvailableResource*. Cabe mencionar que el comportamiento en las dos clases que la declaran es exactamente el mismo. En la *Figura 6.8* se puede ver el código de la declaración e implementación de esta clase y algoritmo.

La declaración de la clase privada *SearchAvailableResource* en *MainActivity.java* es casi igual a la declarada en *PuntoGenericoActivity.java*, **sólo que no hay imágenes**. En la ilustración anterior, se puede observar que se pregunta por la disponibilidad del recurso al método estático *isAvailable(String ruta)* de la clase *Utils.java* del paquete *com.miscelanea*. Este método, a grandes rasgos hace lo siguiente:

```
private class SearchAvailableResource extends AsyncTask<String, Void, String> {
    protected String doInBackground(String... params) {
        // Lugar donde desarrollar la operación de larga duración
        // Recorremos todos los recursos de imágenes que tenemos
        for (int i=0; i < imagenes.size(); i++){
            //Cogemos el i-ésimo de la lista
            Imagen recurso = (Imagen) imagenes.get(i);
            int j = 0;
            //Si el recurso está disponible, pararemos.
            boolean disponible = false;
            //Mientras haya servidores y aún no tengamos el recurso disponible
            while (j<servers.size() && !disponible){
                // Cogemos el j-ésimo
                Servidor server = servers.get(j);
                // Formamos la ruta del recurso real para hacer la petición
                String ruta = server.getRuta_parcial()+recurso.getUrl();
                // La disponibilidad vendrá determinado por el método
                disponible = Utils.isAvailable(ruta);
                // Si está disponible asignamos y posteriormente saldremos del bucle
                if (disponible)
                    urlsImagen[i]=ruta;
                j++;
            }
        }

        // El sistema es análogo para audios, teniendo en cuenta que el número
        // de recursos no tiene por qué coincidir
        for (int i=0; i < audios.size(); i++){
            Locucion recurso = (Locucion) audios.get(i);
            int j = 0;
            boolean disponible = false;
            while (j<servers.size() && !disponible){
                Servidor server = servers.get(j);
                String ruta = server.getRuta_parcial()+recurso.getUrl();
                disponible = Utils.isAvailable(ruta);
                if (disponible)
                    urlsAudio[i]=ruta;
                j++;
            }
        }

        return "Fin";

    }
    // No necesario
    protected void onPostExecute(String result) {
    }
    // No necesario
    protected void onPreExecute() {
    }
    // No necesario
    protected void onProgressUpdate(Void... values) {
    }
}
```

Figura 6.8 .- Algoritmo de la clase privada declarada por PuntoGenericoActivity.java

1. Crea un objeto de la clase *HttpGet* con unos parámetros básicos
2. Ajusta un tiempo de *timeout* de conexión y de socket de 3 y 5 segundos respectivamente
3. Ajusta los parámetros de conexión y los tiempos de *timeout*
4. Se crea un cliente *http* mediante la clase *DefaultHttpClient* y se ejecuta la petición.
5. Se guarda la cabecera de la respuesta, y con ella el tamaño del cuerpo. Sólo se ha bajado la cabecera de la respuesta, no se ha realizado realmente la transferencia del fichero entero.
6. Si el status de la petición es 200 (la petición es correcta [59]) y el contenido de la respuesta es mayor que un mínimo establecido (10 KB, debido a que casi todos los ficheros que descargamos rondan el orden de 1 MB), el recurso se considera disponible para descarga.

En la *Figura 6.9* se muestra la ilustración de este método en lenguaje Java, donde se realiza esta consulta en la clase *Utils.java*

```
public static boolean isAvailable(String url)
{
	try
	{
		HttpGet request = new HttpGet(url);
		HttpParams httpParameters = new BasicHttpParams();
		int timeoutConnection = Constants.TIME_OUT_CONNECTION;
		int timeoutSocket = Constants.TIME_OUT_SOCKET;
		HttpConnectionParams.setConnectionTimeout(httpParameters, timeoutConnection);
		HttpConnectionParams.setSoTimeout(httpParameters, timeoutSocket);

		DefaultHttpClient httpClient = new DefaultHttpClient(httpParameters);
		httpClient.setKeepAliveStrategy(new ConnectionKeepAliveStrategy()
		{
		    public long getKeepAliveDuration(HttpResponse arg0,
		            HttpContext arg1) {
		        return 0;
		    }
		});
		HttpResponse response = httpClient.execute(request);
		//System.out.println(url+" con STATUS "+response.getStatusLine().getStatusCode());
		Header h = response.getFirstHeader("Content-Length");
		int length =0;
		try {
		    length = Integer.parseInt(h.getValue());
		}
		catch (Exception e){
		    e.printStackTrace();
		}
		//System.out.println("Su tamaño es:" +length);
		return ((response.getStatusLine().getStatusCode()) == 200 && (length > Constants.MINIMUM_FILE_SIZE));
	}
	catch (IOException e){
	    //System.out.println(url+" no disponible en estos momentos, TIMEOUT");
	}
	return false;
}
```

Figura 6.9 .- Algoritmo de validación de la disponibilidad de un recurso

Conclusiones

En este apartado enumeraremos las conclusiones que se han sacado al desarrollar este algoritmo.

- Si el recurso **no está** puede ser por varias razones:
    - El servidor está caído: con lo cual el servidor no puede dar una respuesta y saltará eñ *timeout* del socket, ya que no se puede crear un canal de comunicación.
    - El servidor al principio de la petición otorgaba servicio y conexión, pero en el tiempo de consulta se perdió la conexión y salta el *timeout* de conexión.
    - El servidor está disponible pero el recurso ha sido movido de sitio o eliminado: el modo de discernir este problema es el de comprobar el peso del fichero. Como se ha dicho antes, todos los recursos pesan cerca de 1MB.
- Si el recurso **está** disponible:
    - Si se deja el suficiente tiempo a la tarea asíncrona del objeto de la clase *SearchAvailableResource* se propagarán por la aplicación las rutas comprobadas y disponibles en ese tiempo. Este tiempo es deducible y es estimable a partir del:
        - Número de servidores (S) (penalizando si hay muchos servidores caídos)
        - Número de recursos (R), ya sean locuciones o imágenes
        - Tiempo de *timeout* (4 segundos de media)
    - El usuario debe esperar **en el caso peor** 4*S*R segundos en la actividad principal ó en el punto genérico para asegurarse que se han elegido bien los servidores a los que acceder al recurso
- Esta selección de servidores no se hace visible al usuario debido a que no se ajusta a los requisitos, y por tanto, no entraría dentro de lo que pide el cliente aunque, indudablemente, es una proposición en cuanto a mejora para el futuro.

Para dejar constancia de las clases que hacen uso de este algoritmo, se dejan a en la *Figura 6.10* las capturas de las interfaces visuales de las actividades implicadas.

Figura 6.10 .- Capturas de las actividades MainActivity.java y PuntoGenericoActivity.java

Audio

Una de las funcionalidades principales de *Itinerarios: Jardines del Buen Retiro* es la narración locutada de los aspectos más importantes del punto de una ruta. Estas locuciones, las cuales detallaremos en esta sección, están grabadas en formato *mp3* por personal cercano a los desarrolladores, con buenas referencias, gran entonación y un excelente filtrado de ruido. Con las locuciones se pretende dar una visión resumida de lo que componen los textos, así como hacer más agradable la interacción con la aplicación para personas invidentes. Estos ficheros de audio ocupan aproximadamente 1 MB y se han de bajar de *Internet*. A continuación se habla de los criterios de diseño para la realización del audio, de la máquina de estados en código que se ha desarrollado y de la adaptación a requisitos *a posteriori* de las peticiones del cliente en la explicación del controlador de mensajería.

Diseño

En este apartado se hablará principalmente de la clase *MediaPlayer* nativa de *Android* en el paquete *android.media*, y de las relaciones entre las clases y actividades más importantes que proporcionan el funcionamiento. Antes de entrar en detalle con éstas, a continuación, en la *Figura 6.11* se ilustra un ejemplo del funcionamiento del audio en un punto de la ruta dentro de la aplicación:

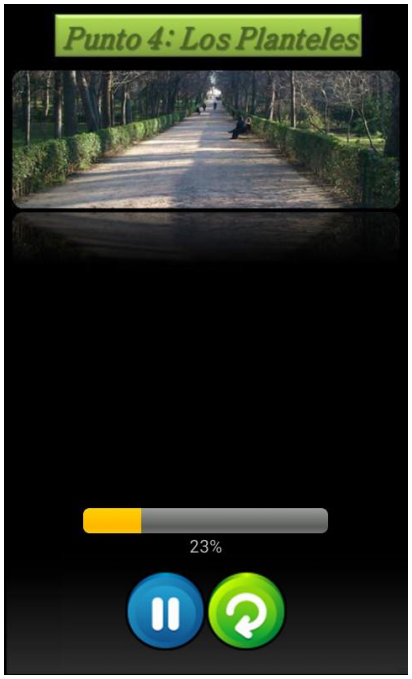


Figura 6.11 .- Pantalla de la reproducción de audio en un punto

De la imagen anterior, se pueden apreciar tres aspectos que se consideran importantes y que en otros apartados se explicarán con mayor detalle a nivel de código:

- **Botón de parar/continuar/iniciar (en azul):** pausa la reproducción si está en curso, la continúa si está pausada o la empieza si no se ha iniciado todavía
- **Botón de repetición (en verde):** repite la reproducción en curso. Esta acción es similar a iniciar si la reproducción todavía no ha sido iniciada.
- **Barra de progreso (en amarillo):** indica en cada momento el porcentaje reproducido de la aplicación
    - Esta barra de progreso se añadió posteriormente a petición del cliente, con el propósito de que el usuario final pudiese estar informado de la duración de la reproducción en curso

Tal y como se dijo en la sección de Casos de uso, es posible acceder al audio de dos maneras diferentes. La primera es desde la pantalla principal de la aplicación, donde se accederá al audio introductorio del Retiro. La segunda manera es accediendo desde cada punto de la ruta,

consiguiéndose así la posibilidad de tener distintos audios (según el punto desde el cual haya pulsado el usuario).

Las características de la reproducción de audio son internas al propio sistema operativo. La reproducción multimedia se lleva a cabo principalmente gracias a la clase MediaPlayer. El uso de esta clase dentro de las aplicaciones Android está muy extendido. En el apartado siguiente se detallarán más características para sacar provecho a un objeto de esta clase

La clase MediaPlayer

Esta clase está integrada en el paquete *android.media* de la API de *Android*. Un objeto de esta clase va a poder pasar por gran variedad de estados:

- Inicializando sus recursos (*initialized*)
- Preparando la reproducción (*preparing*)
- Preparado para reproducir (*prepared*)
- Reproduciendo (*started*)
- En pausa (*paused*)
- Parado (*stopped*)
- Reproducción completada (*playback completed*)
- Finalizado (*end*)
- Con errores (*error*)

Es necesario conocer en qué momento se pueden llamar a qué métodos como se ilustra en la *Figura 6.12*, ya que, por ejemplo, no se puede poner en reproducción un objeto de *MediaPlayer* que no haya sido preparado (en estado *prepared*) o no se puede pausar si ya se encuentra en estado pausado (en estado *paused*).

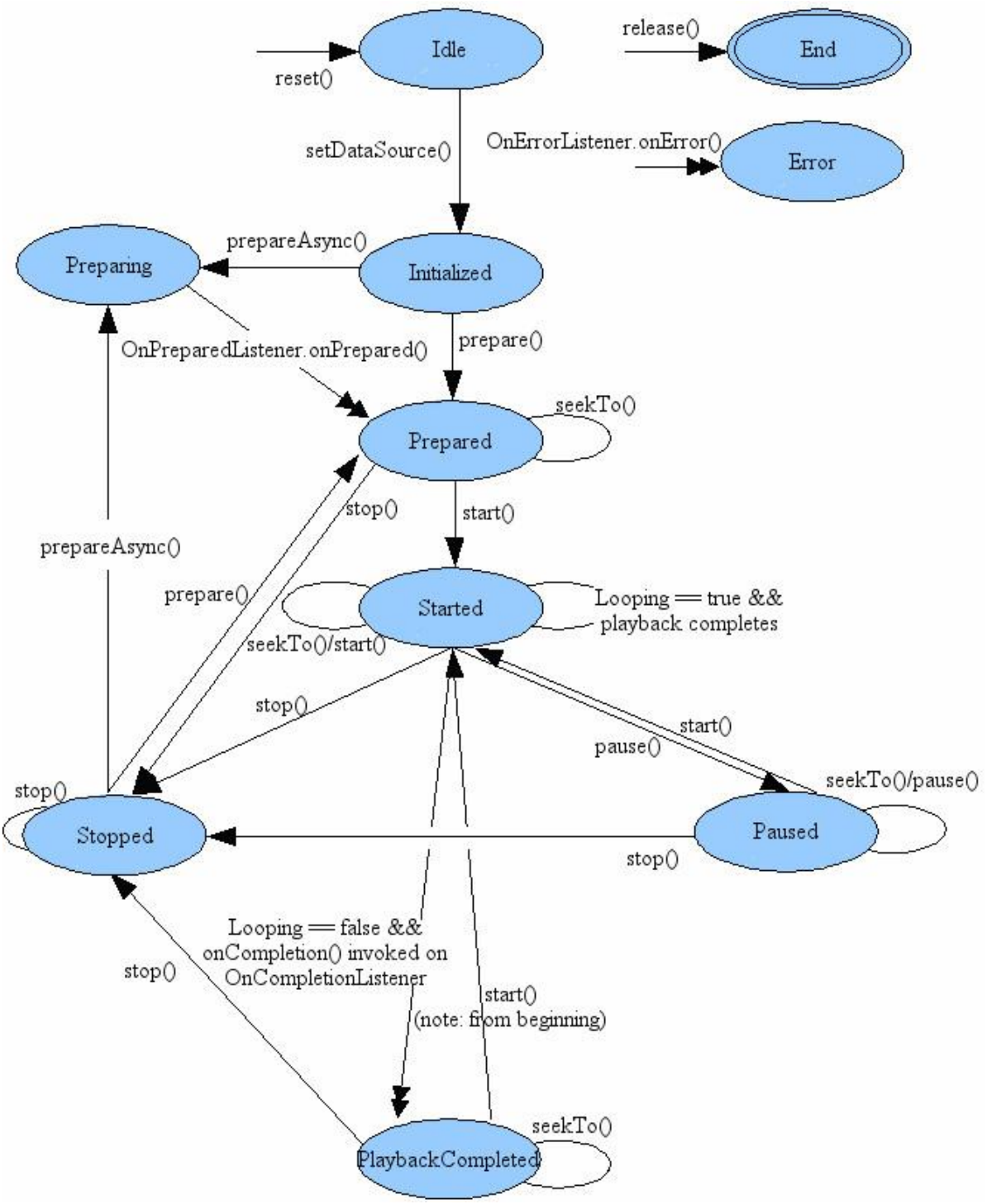


Figura 6.12 .- Diagrama de estados de la clase MediaPlayer[14]

Tras hablar de estados en esta clase *MediaPlayer* se hace evidente la necesidad de implementar una sencilla máquina de estados Moore para una actividad. En dicha actividad el

usuario puede interactuar iniciando, o pausando la reproducción e incluso repitiéndola en caso de que así sea necesario. Esta actividad (ilustrada en la *Figura 6.11*) se llama *AudioActivity* y se detallará en apartados posteriores.

La principal razón de uso de esta clase para la reproducción de sonido de la aplicación es porque soporta la reproducción directa de recursos de audio procedentes de Internet. Para que esto sea posible sólo necesitamos realizar dos pasos:

- Crear el objeto *mediaPlayer* de la clase *MediaPlayer* de la siguiente manera:
    - ***mediaPlayer = MediaPlayer.create(AudioActivity.this, uri);***
        - *mediaPlayer* es el objeto de la clase *MediaPlayer*
        - *create*(Context, Uri) es un método estático de la clase *MediaPlayer*
        - *AudioActivity.this* es el contexto, es decir, la activity desde donde se está invocando la creación
        - *uri* es el objeto de la clase Uri (que contiene una URI -como es lógico-) que contiene el acceso al recurso del servidor, por ejemplo: http://itinerariosretiro1.260mb.org/audio/introduccion.mp3
- Como consecuencia de descarga de Internet desde esta actividad se debe declarar en el fichero *AndroidManifest.xml* del proyecto el permiso de *Internet*:
    - ***<uses-permission android:name="android.permission.INTERNET" />***

Con únicamente estos dos pasos podemos descargar una locución en formato *mp3* dada una URI y empezar a reproducirla. Y pausarla si queremos, ya que una vez que se ha descargado, se puede interactuar con ella todo lo que se quiera.

Máquina de estados

La actividad *AudioActivity* es la encargada de cumplir la máquina de estados de la que hablábamos anteriormente, permitiendo al usuario interactuar con todas las posibilidades que hay gracias a los botones. En la *Figura 6.13* se ilustra la máquina de estados que se ha considerado para anticiparse a la opinión del cliente. Como se comentó en la base de datos, aquí también hay potencial sin explotar, pues no se estableció claramente el diseño hasta el 20 de abril, por lo que

fue necesario considerar todos los casos y dar respuesta a todos los requisitos impuestos **a posteriori**. Los posibles requisitos serían los que vienen a continuación:

- Locuciones internas de la aplicación (no necesaria descarga)
- Locuciones descargadas desde un servidor externo
- Posibilidad de que hubiera varias locuciones por punto

Finalmente, la tercera posibilidad no se pidió por parte del cliente aunque es soportado por la máquina de estados y se perfeccionó para que funcionase correctamente. El diagrama de la máquina estados se muestra a continuación, en la *Figura 6.12*.

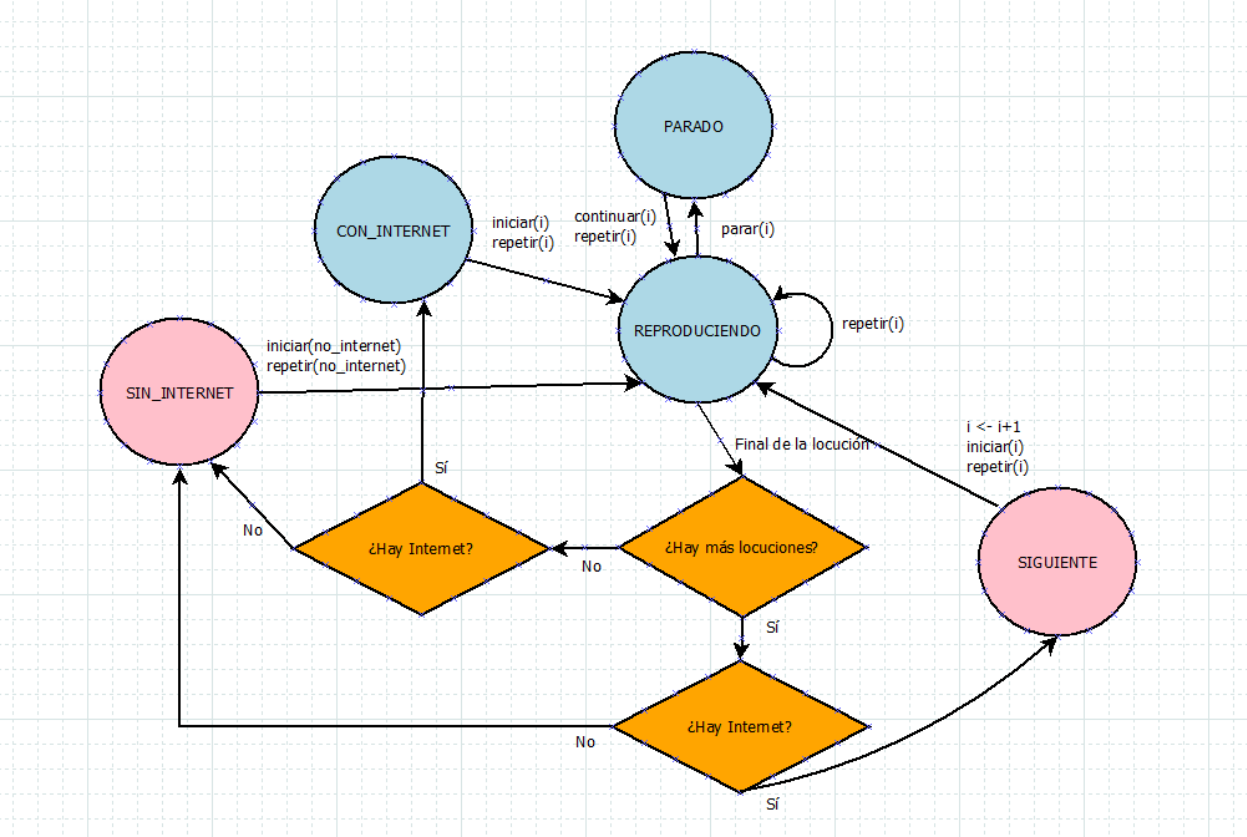


Figura 6.13 .- Lógica de la máquina de estados de AudioActivity

Diagrama de clases

La principal clase de esta sección es *AudioActivity* que es una actividad llamada a través del botón audio desde la interfaz del menú principal de la aplicación (*MainActivity*) y también desde la interfaz punto de la ruta (*PuntoGenericoActivity*). Como podemos observar en la *Figura 6.14* también dispone de una objeto que hereda de AsyncTask y que se encarga de controlar la barra de progreso mencionada anteriormente. Para terminar, tiene un gestor de mensajería para controlar el flujo de concurrencia entre los *threads* de esta aplicación, el cual se explicará en el siguiente apartado. En las clases básicas, como los clásicos botones (*ImageButton*) o el fondo (LinearLayout) se omite su herencia y métodos en la *Figura 6.14* por no ser tan relevantes, ya que se pueden consultar en la documentación de *Android[15]*. En este diagrama se quieren manifestar las relaciones importantes entre clases para hablar posteriormente de ellas.

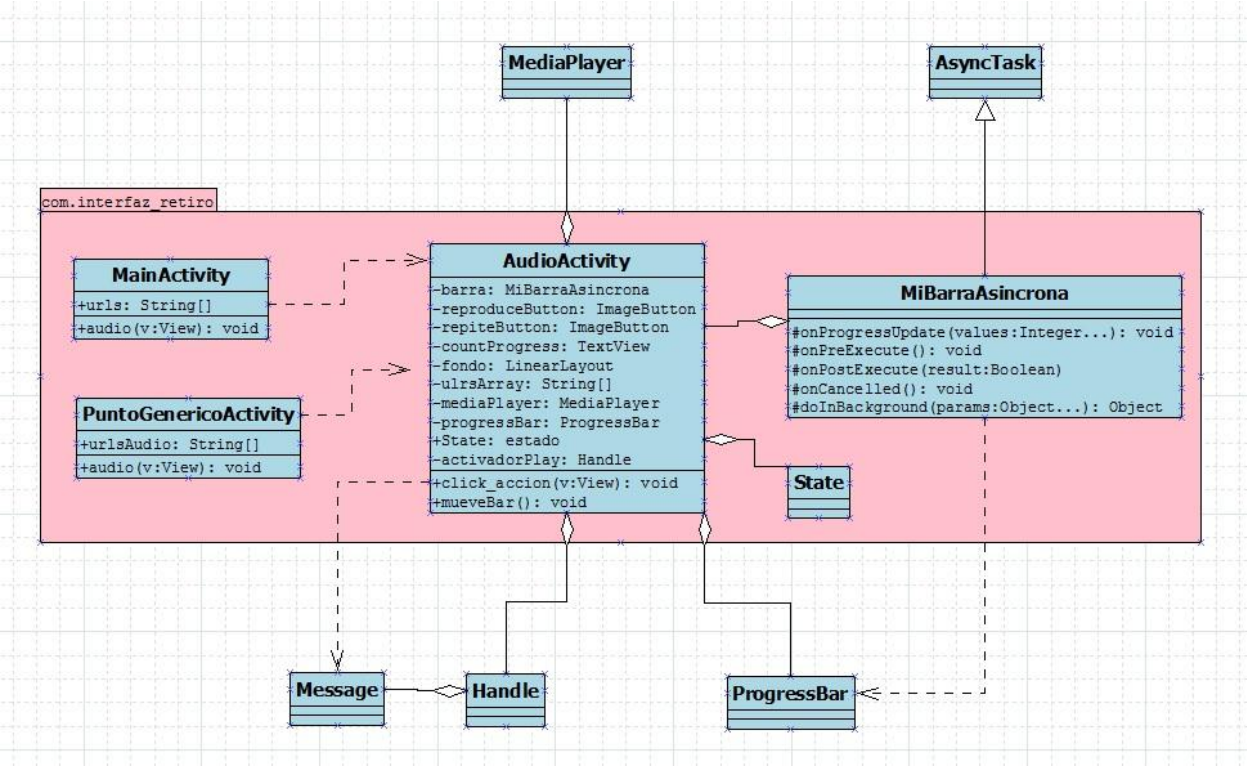


Figura 6.14 .- Diagrama de clases UML del audio de la aplicación

Inicio del audio

La manera de iniciar el audio es mediante el método *void audio(View v)* que será invocado por pulsar el botón de audio de las clases *MainActivity* y *PuntoGenericoActivity*. Antes de ello, dentro de cada una de estas dos clases, se ha debido calcular qué servidor debe prestar el recurso como se definió anteriormente en la sección de preselección de recursos de un servidor. Una vez que se tiene las URL de acceso a los recursos, se puede iniciar la actividad *AudioActivity* como un *Intent* pasándole por el contexto el array de Strings de los recursos de audio que tiene que descargarse y reproducir (que se corresponden con los objetos *urls* en *MainActivity* y *urlsAudio* en *PuntoGenericoActivity*).

Posteriormente, se invocará al método *onCreate* en el que se dejará un mensaje con retardo a esta propia actividad (lo suficientemente grande) para que inicie el *mediaPlayer* en este propio *Thread* y comience a descargarse el recurso y reproducirlo. **Esto es requisito del cliente**, ya que una aplicación móvil en la que se tenga que pulsar más de tres veces los botones para acceder a un recurso no suele triunfar. Por ello, fue implementado *a posteriori* y se introduce (aunque se explicará luego cómo se gestiona) la gestión de los mensajes. De momento, únicamente se detallará cómo se manda un mensaje al controlador de mensajes (el objeto *activadorPlay* de la clase *Handle*) en la *Figura 6.15*.

```
Message msg = new Message();
msg.what = 0;
activadorPlay.sendMessageDelayed(msg, Constants.DELAY_AUDIO);
```

Figura 6.15 .- Modo de mandar un mensaje al gestor de mensajes

Como se puede observar, se define el objeto *msg* con *msg.what* que es un identificador de mensaje (también se pueden pasar argumentos en el mensaje) y se le envía al gestor de mensajes con *Constants.DELAY_AUDIO* que es el retardo para que se programe la acción del mensaje dentro de un tiempo (aunque esto es opcional). Cuando llegue ese retardo, el gestor de mensajes ejecutará la acción que venga determinado por el identificador de mensaje. Esto es necesario

para no bloquear el *thread* principal de la actividad mientras está inicializando el audio y no se ha dibujado todavía la interfaz (el método **onCreate()** siempre tiene que terminar).

Reproducción del audio

Cuando se quiere empezar a reproducir el audio (aislándonos del funcionamiento del gestor de mensajes) se sigue el esquema proporcionado anteriormente de la máquina de estados de la *Figura 6.16*, y el inicio y la descarga es muy intuitiva debido a que se utilizan los métodos de la clase *MediaPlayer* como se ve en el siguiente ejemplo de la *Figura 6.16*.

```
// Tipica maquina de estados
switch (estado) {
case CON_INTERNET:
    if (Utils.isOnline(AudioActivity.this)) {
        reproduceButton.setImageResource(R.drawable.bt_audio_pause);
        try {
            mediaPlayer = MediaPlayer.create(AudioActivity.this,
                    Uri.parse(urls.get(siguiente)));

            mediaPlayer
                    .setOnPreparedListener(new OnPreparedListener() {

                        @SuppressWarnings("unchecked")
                        public void onPrepared(MediaPlayer mp) {
                            mp.start();
                            progressBar
                                    .setVisibility(ProgressBar.VISIBLE);
                            progressBar.setProgress(0);
                            progressBar.setMax(mp.getDuration());

                            barra = new MiBarraAsincrona();
                            barra.execute();
                            estado = State.REPRODUCIENDO;
                        }

                    });
```

Figura 6.16 .- Forma de descargar e iniciar el audio del objeto mediaPlayer

Se puede observar que estamos en el estado '*CON_INTERNET*', se pregunta si se tiene conexión a Internet y, en este caso, se procede a descargar el recurso procedente de la URL. Posteriormente, cuando el recurso ha sido descargado, se ejecuta el método *start()* y se iniciará la reproducción mientras la barra de progreso se irá actualizando a cada momento que el audio progresa con la locución.

Barra de progreso

Como se facilitó en el diagrama de clases de la *Figura 6.3.4,* tenemos dos conceptos diferentes de barra. Por un lado, tenemos el objeto de la clase *MiBarraAsincrona* que es una tarea parecida a la de *SearchAvailableResourse* del apartado 6.1 de esta sección y por otra parte, se tiene el objeto de la clase *ProgressBar,* que es una imagen en pantalla que representa una barra de progreso. Sólo se debe comentar lo siguiente:

- **objeto *MiBarraAsincrona***: como su propio nombre indica, es una tarea asíncrona que se ejecuta en paralelo al *thread* principal de la actividad. Se le encomienda preguntar por cuánta cantidad de audio se ha reproducido para, si se ha cambiado el estado del objeto mediaPlayer, ajustar el progreso de la barra del objeto *ProgressBar* y el texto que informa del porcentaje.
- **objeto *ProgressBar***: es una imagen colocada en la interfaz en el que se va coloreando la relación de audio reproducido por audio total de la locución actualmente reproducida o a ser reproducida.

Gestor de mensajería

Como se introdujo anteriormente, el gestor de mensajería sirve para realizar acciones a través de un objeto de la clase *Handler* que, por sí sola, almacena en una cola de prioridad según el tiempo de retardo una cola de órdenes (o métodos a ser ejecutados). La manera de llamar a este objeto ya la hemos ilustrado en la *Figura 6.15,* mientras que la inicialización se plasma a continuación en la *Figura 6.17*. El tratamiento de mensajes es muy sencillo, pues según el argumento del tipo de mensaje que le llega al gestor hará una cosa u otra. En la *Figura 6.17* también se puede ver que se llaman a métodos de la actividad principal o se ejecuta alguna acción singular sin mucha relevancia como modificar un *TextView* según el dato pasado por argumento.

Este desarrollo del gestor de mensajes se realizó para evitar dos asuntos problemáticos que sólo en la experiencia al tratarse de un problema real puede darse. Por un lado, se tenía que evitar que el *thread* principal quedase bloqueado en el método *onCreate* (mensaje de tipo 0) y por otro, la comunicación entre tareas asíncronas (*MiBarraAsincrona*) y tareas síncronas (el *thread* principal de *AudioActivity*) está vetada. Por ello, se necesita una manera de ‘sincronizar’

los mensajes asíncronos enviados al *thread* principal con el motivo de actualizar la propia barra de progreso que forma parte de la actividad *AudioActivity*.

```
activadorPlay = new Handler() {
    @Override
    public void handleMessage(Message msg) {
        switch (msg.what) {
        case 0:
            click_accion(null);
            break;
        case 1:
            countProgress.setText(msg.arg1 + "%");
            break;
        }
        super.handleMessage(msg);
    }
};
```

Figura 6.17 .- Tratamiento de mensajes del gestor de mensajes

Capítulo 7 - Operaciones de despliegue

Descarga, instalación y ejecución de la aplicación

La aplicación se puede descargar desde la página web: www.tecnologiaUCM.es [5]. Desde este enlace, que se muestra en la *Figura 7.1*; se debe pulsar en el apartado de "Proyectos" para descargarse el archivo Itinerarios.apk.

Figura 7.1 .- Descarga de .apk desde tecnologiaUCM.es.

Si se ha descargado el fichero desde un ordenador, se debe copiar en la memoria del teléfono para que desde allí se proceda a su instalación. En cambio, si el archivo se ha descargado a través del dispositivo se abre el instalador de aplicaciones.

En algunos dispositivos está configurada por defecto la prohibición para instalar aplicaciones de procedencia externa. Para ello hay que cambiar esta característica en los Ajustes de aplicaciones del dispositivo tal y como se muestra en la *Figura 7.2*.

Figura 7.2 .- Permitir aplicaciones de origen desconocido.

Al proceder a la instalación de la aplicación se mostrará una pantalla con el resumen de permisos de control que tendrá la aplicación sobre el terminal (ver *Figura 7.3*). Entre estos controles observamos el permiso de acceso a Internet, la geolocalización y el permiso para la modificación o eliminación de contenido de la tarjeta SD.

Figura 7.3 .- Inicio de la instalación.

Una vez instalada la aplicación podemos validar la instalación (botón "Listo") o directamente ejecutar la aplicación (botón "Abrir") como se observa en la *Figura 7.4*. Si decidimos validar la instalación, para abrirla hay que buscar el icono de *Itinerarios: Jardines del Buen Retiro* entre las aplicaciones instaladas en nuestro móvil. De una u otra manera, se ejecuta la aplicación como se muestra en la *Figura 7.5*.

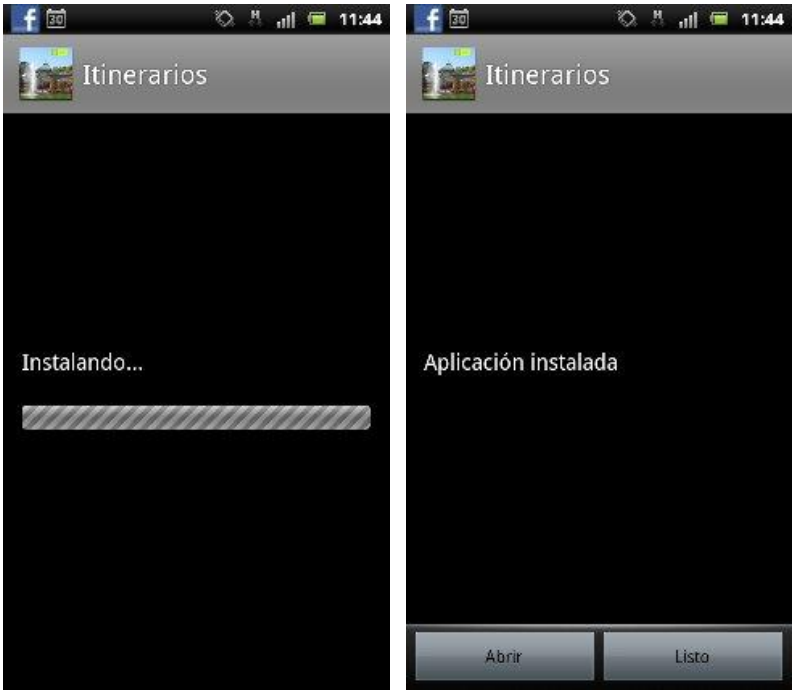


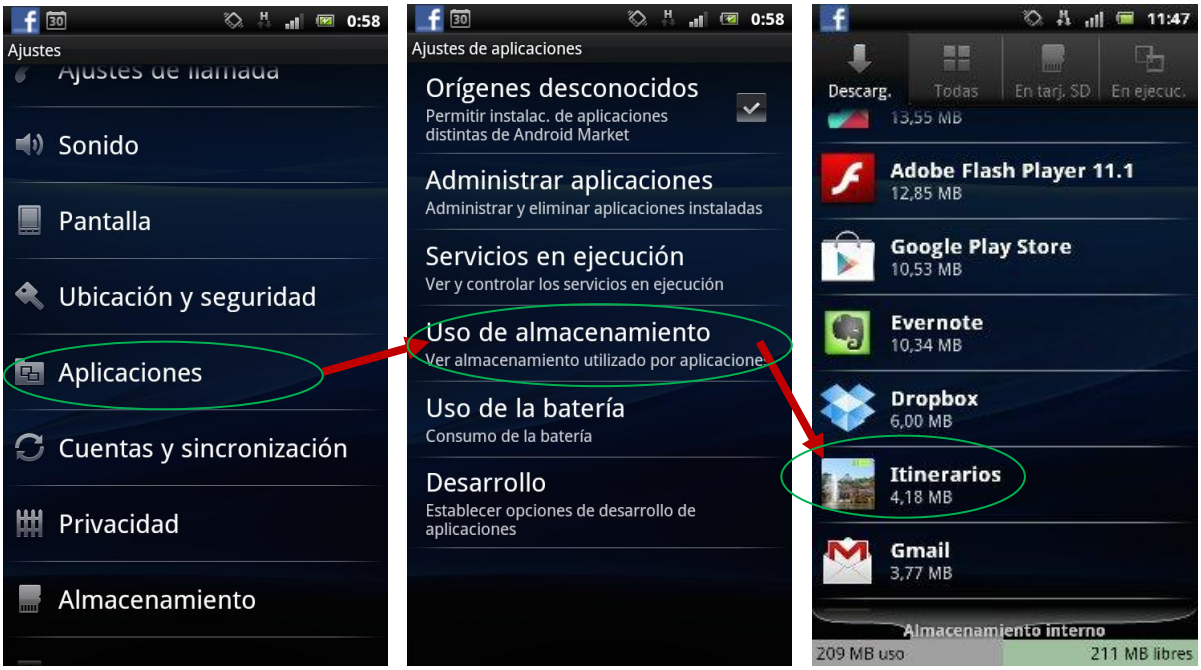
Figura 7.4 .- Proceso de instalación y finalización.

Figura 7.5 .- Icono y pantalla inicial de la aplicación.

Cuando interactuamos con el contenido multimedia de la aplicación (imágenes y locuciones) estos recursos se almacenan en la memoria del dispositivo. Para liberar este uso de almacenamiento en la memoria se puede: pulsar el botón de menú dentro de la aplicación eligiendo la opción que deseemos para liberar el espacio utilizado. La *Figura 7.6* muestra el menú que se despliega tras realizar esta acción.

Figura 7.6 .- Liberación de memoria desde la aplicación

Otra manera de limpiar la memoria es a través de los Ajustes del dispositivo. Sobre el menú aplicaciones de Ajustes, en el "Uso de Almacenamiento", se muestran las aplicaciones instaladas y su espacio ocupado, como podemos ver en la *Figura 7.7*.



.....Figura 7.7 .- Uso de Almacenamiento del dispositivo.

Al pulsar sobre *Itinerarios: Jardines del Buen Retiro*, además de poder ver las características de la aplicación; se podrá liberar el espacio ocupado por los recursos multimedia al presionar el botón “borrar datos” y/o “borrar caché”. La *Figura 7.8* muestra este hecho.

Figura 7.8 .- Resumen de la aplicación y liberar memoria a través de los ajustes.

Una opción futura para realizar la descarga de la aplicación es mediante el escaneo de códigos QR [6] situados en diversos paneles informativos en el parque. Esta tarea es una idea futura que está en proceso de implantación y que depende de la decisión del Ayuntamiento de Madrid para su colocación. De esta manera se contribuirá en gran medida a una mayor difusión. Para poder descargar la aplicación desde tecnologíaUCM.es se puede escanear el código QR de la *Figura 7.9*.

Figura 7.9 .- Código QR para la descarga de itinerarios .apk

Pruebas de campo

En este apartado se detalla las pruebas excepcionales en las que uno o varios componentes han ido a realizar labores de *testing* dentro del recinto del Parque del Retiro de Madrid. Al ser un proyecto dependiente del entorno en el que nos movemos por llevar incorporado un trazado de rutas, es necesario probarlo debidamente. Por cada prueba realizada, se indicará lo contenido a continuación:

- Asistentes
- Fecha
- Hora
- Lugar

- Duración
- Motivo de la prueba
- Errores descubiertos
- Conclusiones

Grabación del primer vídeo de la aplicación

Asistentes: Alberto Segovia

Fecha: 5 de mayo del 2013

Hora: 12:30 h

Lugar: por los Jardines del Buen Retiro, por los caminos de la ruta que sigue la aplicación Itinerarios. Especialmente, los que aparecen en el vídeo (del punto 4 al 3, desde el Jardín de los Planteles al Parterre).

Duración: dos horas y media, debido a una correcta grabación del vídeo.

Motivo de la prueba: la grabación de un primer vídeo que mostrase todas las habilidades del vídeo, empezando por el apartado de cómo usar, siguiendo por los textos introductorios, así como la galería de imágenes y el cómo ir a un lugar específico de la ruta. Posteriormente, se acaba el vídeo con la reproducción del audio del punto del Parterre.

Errores descubiertos: se descubrieron bastantes errores que no se pudieron probar anteriormente debido a que el equipo se centró en desarrollar. Por fortuna, fueron costosos pero posibles de solucionar. A continuación se enumeran.

- El muestreo de imágenes dentro de la galería de imágenes no estaba liberando memoria, por lo que cuando se abría una imagen pesada que necesitaba más recursos (como el plano), se consumía la máxima memoria por actividad permitida.
    - Ahora libera memoria una vez salimos de la galería
- El caso anterior, al abrirse el plano se colgaba la actividad de *Itinerarios: Jardines del Buen Retiro*.
- El audio al terminar su reproducción se quedaba sin iniciar, pero el mensaje de la barra de progreso se quedaba completo y no se actualizaba.

- El trazado de las rutas estaba mal. En lugar de ir por el punto i-ésimo más cercano, se iba por el punto (i-ésimo)-1 más cercano, lo cual dificultaba enormemente la grabación del video debido a que elegía otra ruta y daba un rodeo inexplicable para ir al objetivo.
- Gracias a otro *bug* descubierto mientras se caminaba por la ruta se pudo grabar el vídeo, existían dos puntos con exactamente las mismas coordenadas, por lo que en este caso sí se trazaba bien la ruta (se trata de dos puntos correspondientes a los Planteles).
- El audio en algunas ocasiones se colgó en dos ocasiones debido a la mala iniciación del mismo

Conclusiones:

- Cuando se hace un cambio y se puede probar, hay que hacerlo. Lo de la barra de progreso ha sido un despiste que no se debería haber producido
- Las tareas pesadas de minería de datos hay que revisarlas lo máximo posible antes de validarlas
- Hay que intentar probar la aplicación también con tarifa de datos, ya que el ritmo de descarga es más lento y pueden suceder nuevos eventos de errores
- La grabación de vídeo debe hacerse cerca del anochecer pues la luz del sol refleja la pantalla del dispositivo móvil

Presentación de la aplicación con la funcionalidad completa al personal del Retiro

Asistentes: Guadalupe Miñana, Miguel Bueno, Alberto Segovia, Víctor Torres y Alberto Díaz
Fecha: 14 de mayo del 2013
Hora: 16 h
Lugar: el huerto del Retiro, dentro del propio Parque del Retiro
Duración: hora y media, debido a reunión
Motivo de la prueba: fue una demostración de las capacidades de la aplicación a Alberto Díaz, uno de los principales clientes del personal del Retiro que ha puesto a disposición del equipo todos los textos y algunas imágenes de las que se pueden ver en la aplicación

Errores descubiertos: se puso de manifiesto sólo un *bug*, debido también a la naturaleza del problema porque la señal de GPS no alcanza a todos los hogares.

- Las peticiones al servicio de GPS del dispositivo móvil por parte de la aplicación eran excesivas, por lo que se tuvo que revisar los tiempos y hacer una prueba más exhaustiva
    - Actualmente, las peticiones al inicio de la ruta son instantáneas, mientras que una vez que ya se ha localizado al usuario, se tarda un minuto en volver a pedir servicios GPS. De esta manera, se ahorra el consumo de batería necesitado por la aplicación

Conclusiones

Hay que mencionar la existencia de una comisaría de policía cerca del lugar donde se estaban haciendo las pruebas, por lo que es posible que alterase el comportamiento de la prueba.

Por último, a la hora de descargarse un audio y tardar mucho la descarga (apenas llegaba conexión), se canceló exitosamente la descarga debido a un *timeout* de la aplicación, por lo que fue un éxito en un escenario extremadamente adverso. En el parque del Retiro hay siempre conexión a Internet por tarifa de datos salvo cuando se está cerca de la comisaría.

Capítulo 8 - Conclusiones y trabajo futuro

Itinerarios: Jardines del Buen Retiro es una aplicación pionera en las visitas guiadas del contexto del Retiro. Es la primera aplicación *Android* con **información oficial** proporcionada por personal del Ayuntamiento de Madrid y del parque del Buen Retiro.

Durante el desarrollo de la aplicación se han dado **soluciones a problemas reales**. Estos problemas se han resuelto, y de los más importantes, se explica su solución a continuación. Se hizo imposible el trazado de rutas eficientes a pie mediante *Google Maps* por los caminos del parque del Retiro, por lo que se solucionó mediante un grafo que abstrae los puntos e intersecciones más importantes del recinto y el cálculo del camino mínimo del punto origen al destino. La posibilidad de que un servidor de *hosting gratuito* dejase de estar operativo se ha solucionado mediante un algoritmo de precálculo del recurso a descargar que esté disponible en el servidor. Además, para solucionar la demanda de información locutada de calidad por parte del cliente se ha contado con la colaboración de recursos externos para afrontar dicha calidad. Para el problema planteado por el tamaño del instalador y, de que un usuario sin tarifa de datos no pierda información al interactuar con la aplicación, existe un compromiso entre el tamaño que ocupa la aplicación en el instalador y el contenido que se descarga, buscando siempre la ausencia de la dependencia de Internet para el máximo disfrute de la aplicación por parte de los usuarios.

En la labor de desarrollo de la aplicación hay que hacer un recordatorio de la utilidad de los contenidos que se han adquirido durante la carrera para la resolución de algoritmos mínimos, así como en la construcción de la base de datos y para la aplicación la aplicación de patrones de diseño e ingeniería del software. De igual manera, ha sido muy importante la teoría de las telecomunicaciones y de sistemas operativos aprendida en la carrera.

Cabe destacar que la aplicación ha sido presentada en diversos lugares de la ciudad de Madrid, como en la Catedral de la Innovación (http://www.lacatedralonline.es) el día 17 de mayo frente a representantes de empresas y a profesionales de la tecnología. El cartel promocional de dicho acto se observa en la *Figura 8.1*. Así mismo, se ha presentado la aplicación en el Ayuntamiento de Madrid el 23 de mayo de 2013 frente a:

(1) Responsables y técnicos municipales del Área de Medioambiente y Movilidad. Entre los cuales resultan implicados los siguientes órganos jurídicos:

(a) Dirección General de Áreas Urbanas, Coordinación y Educación Medioambiental
(b) Dirección General de Zonas Verdes, Limpieza y Residuos
(c) Dirección General de Control ambiental, Transportes y Aparcamientos
(d) Dirección General de Sostenibilidad

(2) Coordinadores de los Centros de Educación Ambiental de Retiro, Casa de Campo y Dehesa de la Villa
(3) Responsables y técnicos municipales del IAM, entre los cuales:
(a) Subdirección de Sistemas de Información sectoriales
(b) Subdirección General de la Unidad de Apoyo

Con el objetivo de publicitar y facilitar la descarga de la aplicación se ha adquirido el compromiso del Ayuntamiento de Madrid para la colocación de códigos QR en diversos paneles del parque del Buen Retiro.

Uno de los mayores atractivos futuros para *Itinerarios: Jardines del Buen Retiro* es la adaptación de la aplicación. Dadas sus características y funcionalidades, cumple con una buena base para el desarrollo de futuros proyectos relacionados con el mundo de las rutas turísticas y visitas auto-guiadas. De esta manera, se podría realizar otra aplicación prácticamente con el mismo código. Además, cabe destacar que la aplicación es totalmente ampliable y que soporta más rutas que la de 'Jardines del Buen Retiro', pudiéndose crear nuevas rutas auto-guiadas dentro de este entorno o del que se necesite.

A título personal manifestar que se ha aprendido a programar en *Android* desde cero. Todo lo aprendido ha sido a base de leer documentación y tutoriales proporcionados por nuestras directoras de proyecto y a través de recursos *online*. Todo ello nos ha llevado a conseguir la satisfacción del cliente, pues según dijo Rafa, el encargado del proyecto por parte del Ayuntamiento: "este proyecto ha superado las expectativas notablemente". Por parte del personal del Retiro, quedaron completamente encantados con el desarrollo realizado. Esto es un prestigio, un honor por nuestra parte y son hechos que nos han motivado como equipo.

Finalmente, en cuanto a nuestro futuro próximo, hemos adquirido la visión de un proyecto totalmente real, con personas que son realmente clientes, con entregas en las que puede jugar el tiempo en contra y con muchas presentaciones, reuniones y cambios de requisitos. Por

estos hechos el proyecto posee mayor mérito y satisfacción; y ha sido una experiencia muy enriquecedora.

Figura 8.1 .- Cartel de promocional de la Catedral de la Innovación

Bibliografía

[1] R. Elmasri, *Fundamentos de sistemas de bases de datos*, 5ª ed., último [sic] reimp. Madrid: Pearson, 2008.

[2] N. Martí Oliet, *Estructuras de datos y métodos algorítmicos: ejercicios resueltos*, Reimp, corr. Madrid: Pearson Prentice-Hall, 2012.

[3] «Ayuntamiento de Madrid - Medio Ambiente». [En línea]. Disponible en: http://www.madrid.es/portales/munimadrid/es/Inicio/Ayuntamiento/Medio-Ambiente?vgnextfmt=default&vgnextchannel=4b3a171c30036010VgnVCM100000dc0ca8c0RCRD.

[4] «Actividades ambientales en el Retiro». [En línea]. Disponible en: http://www.actividadesambientalesretiro.com/2012/02/el-retiro-en-el-objetivo-de-nuestra.html.

- Folletos y mapa (Jardines del Buen Retiro) http://www.madrid.es/UnidadesDescentralizadas/AreasUrbanas_EducacionAmbiental/ContenidosBasicos/Publicaciones/Retiro.%20Desplegable/RetiroDesplegable_Plano.pdf
- Folletos e información(Senda botanica del retiro) http://www.madrid.es/UnidadesDescentralizadas/Agenda21/ContenidosBasicos/Publicaciones/SendaBot%C3%A1nicaRetiro11.pdf

[5] «Grupo de Tecnologías Móviles - Universidad Complutense de Madrid». [En línea]. Disponible en: http://www.tecnologiaUCM.es.

[6] «The QR Code Generator», *The QR Code GEnerator*. [En línea]. Disponible en: http://www.the-qrcode-generator.com/.

Estado del arte:

[7] «Guía de Madrid - minube - Aplicaciones de Android en Google Play». [En línea]. Disponible en: https://play.google.com/store/apps/details?id=com.guides.minube.madrid&feature=search_result#?t=W251bGwsMSwyLDEsImNvbS5ndWlkZXMubWludWJlLm1hZHJpZCJd.

[8] «Audio Guías de Sevilla - Aplicaciones de Android en Google Play». [En línea]. Disponible en: https://play.google.com/store/apps/details?id=audio.guias.sevilla&feature=search_result#?t=W251bGwsMSwyLDEsImF1ZGlvLmd1aWFzLnNldmlsbGEiXQ.

[9] «Guía de Córdoba (España) - Aplicaciones de Android en Google Play». [En línea]. Disponible en: https://play.google.com/store/apps/details?id=laguiadetuciudad.com.guia&feature=search_result#?t=W251bGwsMSwxLDEsImxhZ3VpYWRldHVjaXVkYWQuY29tLmd1aWEiXQ.

[10] «Parcs i Jardins - Itineraris». [En línea]. Disponible en: http://mediambient.itineraris.bcn.cat/es.

Cursos y manuales:

[11] «Curso Android: Todo lo que necesitas para empezar», *Maestros del Web*. [En línea]. Disponible en: http://www.maestrosdelweb.com/editorial/curso-android/.

[12] «::: Fundación General de la Universidad Complutense de Madrid :::» [En línea]. Disponible en: http://pendientedemigracion.ucm.es/info/fgu/formacion/escuela_verano/cursos/b18.php.

[13] «Curso Programación Android - Indice de Contenidos | sgoliver.net blog». [En línea]. Disponible en: http://www.sgoliver.net/blog/?page_id=3011.

[14] «Android Developers». [En línea]. Disponible en: http://developer.android.com/index.html.

Recursos Software:

[15] «Java SE Downloads». [En línea]. Disponible en: http://www.oracle.com/technetwork/java/javase/downloads/jdk7-downloads-1880260.html?ssSourceSiteId=otnes.

[16] «Eclipse Classic 4.2.2 | Eclipse Packages». [En línea]. Disponible en: http://www.eclipse.org/downloads/packages/eclipse-classic-422/junosr2.

[17] «ADT Plugin | Android Developers». [En línea]. Disponible en: http://developer.android.com/tools/sdk/eclipse-adt.html.

[18] «Managing Virtual Devices | Android Developers». [En línea]. Disponible en: http://developer.android.com/tools/devices/index.html.

[19] «SQLite Administrator - International Milestone Beta». [En línea]. Disponible en: http://sqliteadmin.orbmu2k.de/.

[20] «TortoiseSVN». [En línea]. Disponible en: http://tortoisesvn.net/.

[21] «Welcome to The Apache Software Foundation!» [En línea]. Disponible en: http://www.apache.org/.

[22] «FileZilla - The free FTP solution». [En línea]. Disponible en: https://filezilla-project.org/.

[23] «Dropbox». [En línea]. Disponible en: https://www.dropbox.com/.

[24] «Microsoft Office - Office.com». [En línea]. Disponible en: http://office.microsoft.com/es-es/.

[25] «Gmail: correo electrónico de Google». [En línea]. Disponible en: https://accounts.google.com/ServiceLogin?service=mail&passive=true&rm=false&continue=http://mail.google.com/mail/&scc=1&ltmpl=default&ltmplcache=2.

Servidores gratuitos:

[26] «Free Web Hosting with PHP, MySQL and cPanel». [En línea]. Disponible en: http://www.site11.com/.

[27] «VZ PLANET :: Hosting Gratuito, 1Gb sin publicidad Host en Español, gratis FTP, PHP, MySQL». [En línea]. Disponible en: http://vzpla.net/.

[28] «Welcome to zz.mu - Managed by Hostinger». [En línea]. Disponible en: http://zz.mu/.

[29] «Hosting gratuito, hosting gratis, alojamiento web gratuito, free host». [En línea]. Disponible en: http://260mb.org/.

Recursos hardware:

[30] «PORTÁTIL ESENCIAL 15,6" R540I - INFORMACIÓN GENERAL | SAMSUNG». [En

línea]. Disponible en: http://www.samsung.com/es/consumer/pc-peripherals-printer/notebook/discontinued/NP-R540-JA02ES.
[31] «PORTÁTIL ESENCIAL 15,6" RV511I - INFORMACIÓN GENERAL | SAMSUNG». [En línea]. Disponible en: http://www.samsung.com/es/consumer/pc-peripherals-printer/notebook/new-essential/NP-RV511-A06ES.
[32] «Acer Support: Aspire 5920G Series Specifications». [En línea]. Disponible en: http://panam.acer.com/acerpanam/notebook/0000/Acer/Aspire5920G/Aspire5920Gsp2.shtml.
[33] «Samsung GALAXY S II». [En línea]. Disponible en: http://www.samsung.com/global/microsite/galaxys2/html/specification.html.
[34] «Especificaciones de Xperia U | Pantalla táctil de 3,5 - Sony Smartphones (Spain)». [En línea]. Disponible en: http://www.sonymobile.com/es/products/phones/xperia-u/specifications/.
[35] «Especificaciones de Xperia neo V | Pantalla táctil de 3,7 - Sony Smartphones (Spain)». [En línea]. Disponible en: http://www.sonymobile.com/es/products/phones/xperia-neo-v/specifications/.
[36] «Tablet PC ICOO iCou8GT- 263606». [En línea]. Disponible en: http://www.ontablets.es/productos/tablet-pc-icoo-icou8gt.

Tratamiento de imagenes:
[37] «nostra13/Android-Universal-Image-Loader · GitHub». [En línea]. Disponible en: https://github.com/nostra13/Android-Universal-Image-Loader.
[38] «Image Converter Software. Compress & Convert JPEG TIFF GIF BMP JPG PNG & More». [En línea]. Disponible en: http://www.nchsoftware.com/imageconverter/index.html.
[39] «GIMP - The GNU Image Manipulation Program». [En línea]. Disponible en: http://www.gimp.org/.
[40] «Download Photoshop CC by joining Creative Cloud today | Adobe Photoshop CC». [En línea]. Disponible en: http://www.adobe.com/products/photoshop.html.
[41] «ImageView | Android Developers». [En línea]. Disponible en: http://developer.android.com/reference/android/widget/ImageView.html.
[42] «ScaleImageView.java - scale-imageview-android - Add pinch-in and pinch-out To Android ImageVIew - Google Project Hosting». [En línea]. Disponible en: https://code.google.com/p/scale-imageview-android/source/browse/src/com/matabii/dev/scaleimageview/ScaleImageView.java?r=d7af83f339bd373a335d45616a99381969fde185.
[43] «scale-imageview-android», *GitHub*. [En línea]. Disponible en: https://github.com/matabii/scale-imageview-android.

Mapas y geolocalización:
[44] «API de Google Maps para Android: biblioteca externa - API de Google Maps para Android — Google Developers». [En línea]. Disponible en: https://developers.google.com/maps/documentation/android/?hl=es#apikey.
[45] «Setup Google Play Services SDK | Android Developers». [En línea]. Disponible en: http://developer.android.com/google/play-services/setup.html.
[46] «Google Maps». [En línea]. Disponible en: https://maps.google.es/.

Documentacion y trabajo grupal:

[47] «Home - MiKTeX Project Page». [En línea]. Disponible en: http://miktex.org/.
[48] «LyX | LyX – The Document Processor». [En línea]. Disponible en: http://www.lyx.org/.
[49] «Generador de PDF, editar PDF | Adobe Acrobat XI». [En línea]. Disponible en: http://www.adobe.com/es/products/acrobat.html.
[50] «Dia draws your structured diagrams: Free Windows, Mac OS X and Linux version of the popular open source program». [En línea]. Disponible en: http://dia-installer.de/.
[51] «Diccionarios de Español, Ingles, Francés, Portugues - WordReference.com». [En línea]. Disponible en: http://www.wordreference.com/es/.
[52] «Google Drive». [En línea]. Disponible en: https://accounts.google.com/ServiceLogin?service=wise&passive=1209600&continue=https://drive.google.com/?usp%3Dchrome_app%26usp%3Dchrome_app%23&followup=https://drive.google.com/?usp%3Dchrome_app%26usp%3Dchrome_app&ltmpl=drive.
[53] «WhatsApp :: Home», *WhatsApp.com*. [En línea]. Disponible en: http://www.whatsapp.com/.

Asuntos específicos:

[54] «2013: ¿La misma cantidad de móviles que de personas?» [En línea]. Disponible en: http://www.bolpress.com/art.php?Cod=2013042804.
[55] S. V.-I. und IT-Service, «Webseite Steinberg Media Technologies GmbH», *http://www.steinberg.net/*. [En línea]. Disponible en: http://www.steinberg.net/en/products/cubase/why_cubase.html.
[56] «NetworkOnMainThreadException | Android Developers». [En línea]. Disponible en: http://developer.android.com/reference/android/os/NetworkOnMainThreadException.html.
[57] «Keeping Your App Responsive | Android Developers». [En línea]. Disponible en: http://developer.android.com/training/articles/perf-anr.html.
[58] «AsyncTask | Android Developers». [En línea]. Disponible en: http://developer.android.com/reference/android/os/AsyncTask.html.
[59] «Anexo:Códigos de estado HTTP», *Wikipedia, la enciclopedia libre*.
[60] «Algoritmos de Dijkstra», *Wikipedia, la enciclopedia libre*. http://es.wikipedia.es/wiki/Algoritmo_de_Dijkstra
[61] «Mapas en Android (Google Maps Android API v2 I)», *sgoliver.net blog*. http://www.sgoliver.net/blog/?p=3244
[62] «Centro de Información y Educación Ambiental del Retiro», *blog Actividades ambientasles en el Retiro*. http://www.actividadesambientalesretiro.com

Anexo A - Historial de Revisiones de este documento

Fecha	Versión	Descripción	Autor
25/10/2012	0.0	Maquetación de la plantilla según el criterio establecido	Alberto Segovia
29/10/2012	0.1	Primera versión del primer capítulo de la memoria	Alberto Segovia
30/10/2012	0.2	Correcciones de algunas erratas tras la revisión de Victoria y tabla de revisiones añadida	Alberto Segovia
22/11/2012	0.3	Adaptación de algunas frases, añadido algún requisito funcional.	Alberto Segovia
06/02/2013	0.4	Aporte y adaptación de ideas y corrección de erratas.	Víctor Torres
08/02/2013	0.5	Adición de apartados especificados por las directoras del proyecto, eliminación de algunos y redacción de otros tantos. Repasado el capítulo 1, la introducción del estado del arte y el reporte de incidencias. Se realiza el resumen en castellano y en inglés.	Alberto Segovia
10/02/2013	0.6	Contraste de la información contenida en el documento con el cambio de requisitos del cliente. Ahora debería ser coherente	Alberto Segovia
12/02/2013	0.7	Se cambia el nombre a algunos apartados, dejando Especificación de Requisitos y Casos de Uso	Alberto Segovia
12/05/2013	0.71	Se añaden apartados a rellenar por cada uno de los miembros	Alberto Segovia
24/05/2013	0.8	Se añade la parte de recursos *software*	Alberto Segovia
26/05/2013	0.9	Se adjunta la parte de recursos *hardware*	Alberto Segovia

29/05/2013	0.10	Se adjunta la parte de diseño de bases de datos (no está completa)	Alberto Segovia
03/06/2013	0.11	Adjuntado diagrama UML al diseño de bases de datos y revisado su contenido	Alberto Segovia
05/06/2013	0.12	Insertado en requisitos, herramientas y recursos el capítulo de material utilizado. Revisada la parte de base de datos y herramientas y recursos	Alberto Segovia
05/06/2013	0.13	Se añade el apartado de Pruebas de Campo	Alberto Segovia
05/06/2013	0.14	Se añaden los apartados Introducción y Estado del Arte	Víctor Torres
06/06/2013	0.15	Se añaden los apartados *ScaleImageView* y Descarga instalación y ejecución de la aplicación	Víctor Torres
06/06/2013	0.16	Se añade la parte de la selección de servidores para acceder a un recurso	Alberto Segovia
06/06/2013	0.17	Se añaden Casos de uso; Diseño, Arquitectura del sistema;	Miguel Bueno
06/06/2013	0.18	Bibliografía	Víctor Torres
07/06/2013	0.19	Clase *MapFragment*	Miguel Bueno
07/06/2013	0.20	*Dijkstra*	Miguel Bueno
09/06/2013	0.21	Añadidas referencias (parciales) a la bibliografía	Alberto Segovia
09/06/2013	0.22	Añadidos cambios en bibliografía	Víctor Torres
09/06/2013	0.23	Añadido Anexo B, Historial de bugs y mejoras	Víctor Torres
09/06/2013	0.24	Añadida el apartado de librería externa *Universal Image Loader*	Alberto Segovia
10/06/2013	0.25	Cambios en figuras, correcciones y referencias a bibliografía	Víctor Torres
11/06/2013	1.0	Se añade el apartado del audio y se versiona a 1.0 debido a que todos los apartados de la memoria oficiales están integrados	Alberto Segovia
12/06/2013	1.1	Se corrige y completa el capítulo 3 de	Alberto Segovia

		Especificación de Requisitos y Herramientas	
12/06/2013	1.2	Se corrigen diversos capítulos de la memoria	Víctor Torres
12/06/2013	1.3	Se corrige el diseño de la base de datos, la librería Universal Image Loader y el algoritmo de preselección de servidores	Alberto Segovia
13/06/2013	1.4	Varias correcciones, inserción de diagramas de casos de uso y ampliación de la sección referente al algoritmo de Dijkstra	Miguel Bueno
13/06/2013	1.5	Inserción del manual de usuario	Alberto Segovia
13/06/2013	1.6	Redacción y corrección del capítulo Conclusiones y trabajo futuro	Alberto Segovia y Víctor Torres
13/06/2013	2.0	Numeradas las figuras con (Capítulo.Numero) referenciadas todas las figuras, formatos de párrafo adecuados y Actas concatenadas al final	Alberto Segovia
15/06/2013	2.1	Revisión de formato de las imágenes, cambio del título de la portada y tamaño de fuente de las capturas de pantallas	Alberto Segovia
16/06/2013	2.2	Añadido el resumen del inglés después del cambio del resumen en castellano.	Alberto Segovia
16/06/2013	2.21	Revisión completa de la memoria	Alberto Segovia
16/06/2013	2.22	Añadido el prólogo y revisada la introducción	El equipo

Anexo B - Manual de usuario

En este anexo se describe un sencillo manual de usuario dedicado especialmente a aquellas personas que no tengan mucha experiencia interactuando con aplicaciones *Android*. Si se desea obtener más información de cómo realizar ciertas acciones más avanzadas, se debe consultar el apartado "cómo usar" interno de la aplicación. Como se verá más tarde, se puede ir desde la pantalla principal.

Requisitos del usuario

Este manual supone que la aplicación ya está instalada en el terminal en el que se va a ejecutar. Siguiendo todos los pasos del presente manual, se llegará a las pantallas que aquí recogemos, siempre y cuando se tengan las opciones adecuadas activadas. Sin estas opciones habilitadas habrá algún paso que no se pueda realizar, así que para un correcto funcionamiento de *Itinerarios: Jardines del Buen Retiro* debe activarlas. Estas opciones se enumeran a continuación:

- GPS y estar al aire libre: para geolocalizarnos y poder encontrar el punto más cercano, así como la ruta.
- Conexión a Internet, ya sea tarifa de datos de la teleoperadora o WI-FI.
- La opción de giroscopio activada (o más comunmente llamada "Rotar automáticamente") También es vital que sepa cuál es el botón atrás de su dispositivo móvil porque desde

Todas las pantallas se puede volver a la anterior si así se desea. En la *Figura B.1* se adjunta un ejemplo del botón atrás en un Samsung Galaxy S III, destacándose en **rojo**. Si tiene alguna duda, consulte el manual de instrucciones de su dispositivo móvil para saber cuál es el botón de atrás.

Inicio de la aplicación

Para iniciar la aplicación nos situaremos en el icono "Itinerarios" y pulsaremos en él como se muestra en la *Figura B.2*. De esta manera se llegará al Menú Principal del que se destaca una serie de botones en la *Figura B.3* que se comentarán a continuación:

- **Botón de textos** (**rojo**): mostrará la pantalla del texto introductorio como se verá reflejado en la *Figura B.4*.

- **Botón de persona andando** (**morado**): irá a la pantalla de selección del modo de ruta donde también está el Plano del Retiro como se puede observar más adelante en la *Figura B.10*
- **Botón del altavoz** (**marrón**): aparecerá la pantalla del audio empezando a descargarse y reproducirse como se puede observar más adelante en la *Figura B.7*
- **Botón del interrogante** (**azul**): muestra una pantalla que despliega toda la información necesaria para comprender la ejecución de la aplicación como se ve en la *Figura B.5*
- **Botón de la i** (**verde**): muestra la información de los creadores de esta aplicación como se puede observar en la *Figura B.6*.

Figura B.1

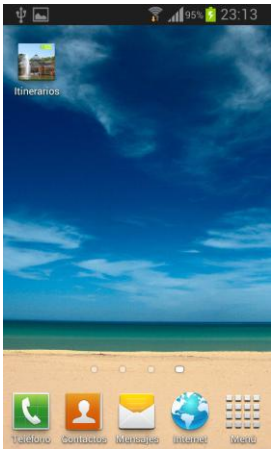


Figura B.2

Figura B.3

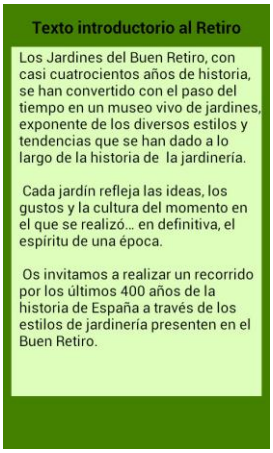


Figura B.4

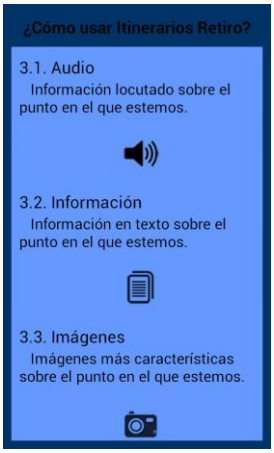


Figura B.5

Figura B.6

Audio

Como se dijo antes, en la *Figura B.7* se procede a la descarga del audio y durante unos segundos la aplicación estará esperando hasta que se termine. Después, como vemos en la *Figura B.8*, empezará la locución, por lo que si no se escucha nada, se aconseja subir el sonido del dispositivo. Más tarde podremos **pausar** la reproducción (o repetir) para llegar a la *Figura B.9*, en la que también tendremos dos opciones, **continuar** la reproducción o **repetirla** por si se ha perdido algún detalle mientras la escuchaba.

Figura B.7

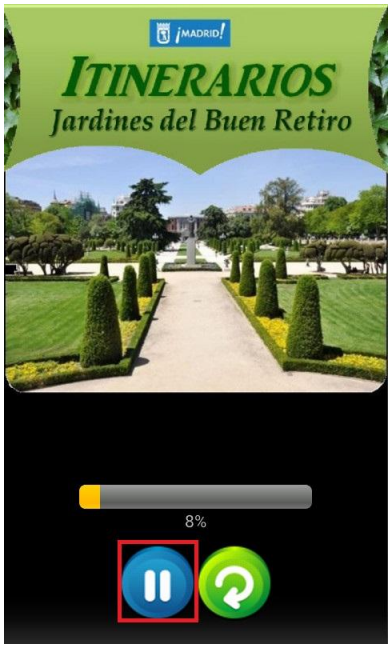


Figura B.8

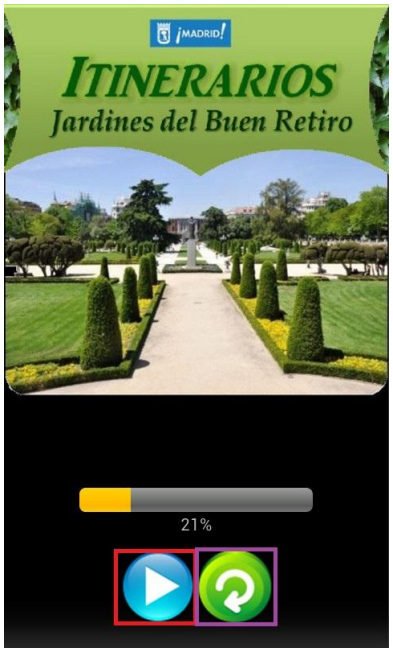


Figura B.9

Selección de modo de ruta

En esta pantalla (*Figura B.10*) podemos elegir el modo de ruta que se desee, existiendo estas distintas vertientes:

- **Botón de punto de inicio** (en **rojo**): si queremos empezar la ruta por el principio. De esta manera, llegaremos a la *Figura B.11*, en la que nos encontraremos en un punto. Para ir al siguiente punto se debe pulsar la flecha hacia la derecha que hay en **rojo** para pasar a la pantalla de la *Figura B.12*. Para volver, al primer punto del que procedemos hay que pulsar la flecha hacia la izquierda que viene rodeado en **azul** en la *Figura B.12*.

- **Botón de ir al sitio más cercano** (en **azul**): si no se quiere empezar de cero el itinerario y se desea visitar el punto de la ruta que quede más cerca, se ha de pulsar esta opción. Aparecerá un mensaje emergente como el de la *Figura B.13*, en el que habrá que esperar hasta que se geolocalice el punto más cercano. Después de esta breve espera, se llegará al punto más cercano como se ve en la *Figura B.14*.
- **Botón de elección del punto** (en **morado**): si se desea elegir un punto específico de la ruta seleccione esta opción. A continuación, se desplegará una lista de puntos como se puede observar en la *Figura B.15*, donde seleccionaremos el punto 5 de la ruta y accederemos a él, como se muestra en la *Figura B.19*.
- **Botón del plano del Retiro** (en **verde**): también se puede consultar el plano del Retiro pulsando en el botón del propio mapa. A continuación, se desplegará la *Figura B.16*, desde la cual podremos hacer *zoom* (por ejemplo, pulsando doblemente en la pantalla) como se ve en la *Figura B.17*. Además, también se puede alejar el zoom volviendo a pulsar doblemente en la pantalla, como se ve en la *Figura B.18*.

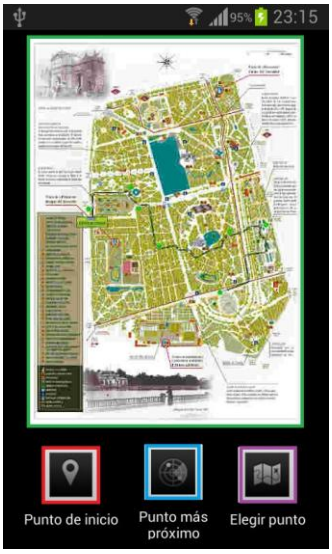


Figura B.10

Figura B.11

Figura B.12

Figura B.13

Figura B.14

Figura B.15

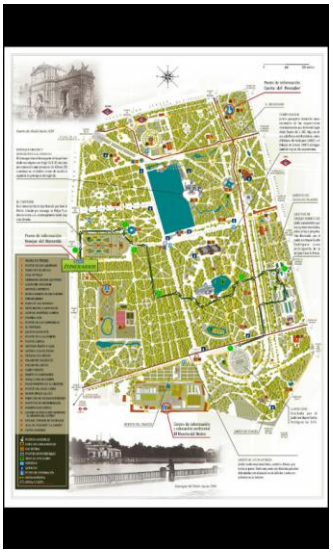


Figura B.16

Figura B.17

Figura B.18

Punto de una ruta

Es la pantalla más importante de la aplicación y se muestra en la *Figura B.19*. En cada punto de la ruta se cuentan los detalles de la zona que representa. Tiene varias opciones, como anterior y siguiente que ya se han comentado, que se enumeran a continuación:

- **Botón de audio** (en **rojo**): de manera muy parecida a como se ha explicado en el apartado de Audio, se puede escuchar la locución dedicada a este punto de la ruta como se puede ver en la *Figura B.20*.
- **Botón de la i de información** (en **azul**): dispone de una cantidad de texto muy extensa que permite saber todo acerca de este punto como se ve en la *Figura B.21*
- **Botón de interrogante** (en **morado**): despliega un texto con las curiosidades de este punto de la ruta, como se puede observar en la *Figura B.22*.
- **Botón de la cámara de fotos** (en **verde**): cada punto de la ruta tiene una cantidad de imágenes asociadas. Pulsando en este botón se abre la lista de imágenes que tiene este punto de la ruta. Si no se han descargado todavía, lo hará como se ve en la *Figura B.23* (con un icono de *Android*). Una vez descargadas, la imagen real en tamaño reducido como se observa en la *Figura B.24*. Si en esta pantalla pulsamos en la imagen marcada en rojo, se nos desplegará la imagen a su tamaño real que nos permite visualizar el dispositivo como aparece parcialmente en la *Figura B.25*. Además, esta galería de imágenes permite que se puedan deslizar las fotos contenidas como si fueran diapositivas. En la *Figura B.25* y *B.26* aparece la secuencia de cómo deslizar la imagen para pasar a la imagen de la derecha (mantener pulsado y mover el dedo hacia la izquierda, como el **recuadro rojo**) que es representada por la *Figura B.27*. Esta imagen parece que no es apropiada verla en vertical, así que gire su dispositivo para verla en horizontal como se observa en la *Figura B.28*. Como se ha aprendido cuando describimos el plano, también se puede hacer zoom para ver a todo detalle la foto, como en la *Figura B.29*
- **Botón de cómo llegar** (en **marrón**): si no sabemos dónde se encuentra este punto de la ruta dentro del Retiro, podemos localizar el punto y obtener la ruta aproximadamente óptima hacia el punto. Tras pulsar este botón se desplegará una pantalla de *Google Maps* en el que se tendrá que esperar a que llegue la señal de GPS, como se observa en la *Figura B.30*. Posteriormente, se verá el camino trazado en la *Figura B.31* (la localización de la captura fue desde fuera del Retiro). Como se puede apreciar en la *Figura B.32*,

podemos pulsar en un marcador para obtener una descripción mejor, así como también realizar dicha acción en el otro marcador como aparece en la *Figura B.33*. Para terminar, cabe destacar que este mapa también tiene *zoom* y se pueden usar los botones de la esquina inferior derecha para acercar (y alejar) como en la *Figura B.34*.

Figura B.19

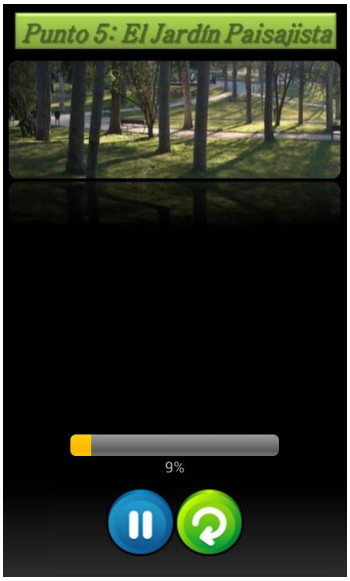


Figura B.20

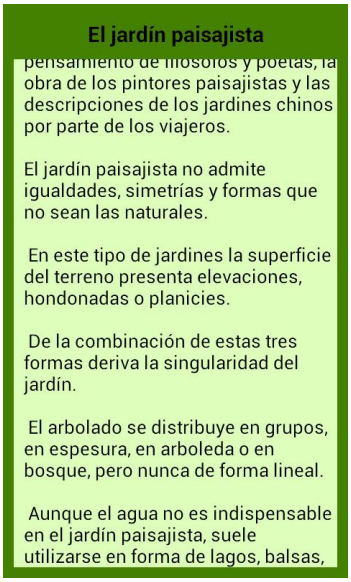


Figura B.21

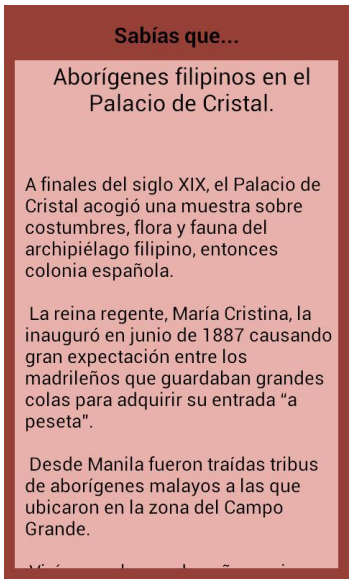


Figura B.22

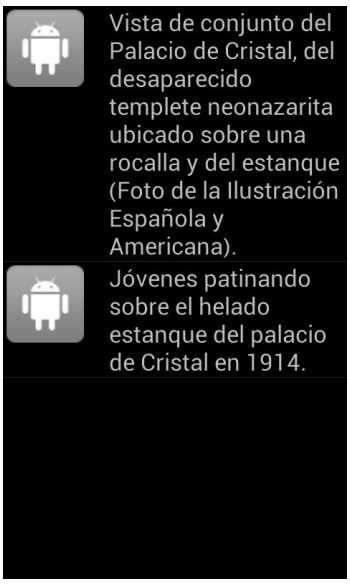


Figura B.23

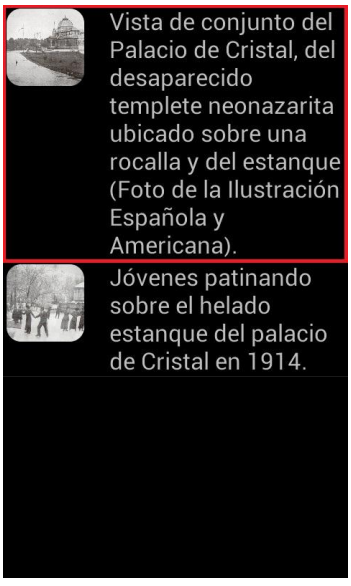


Figura B.24

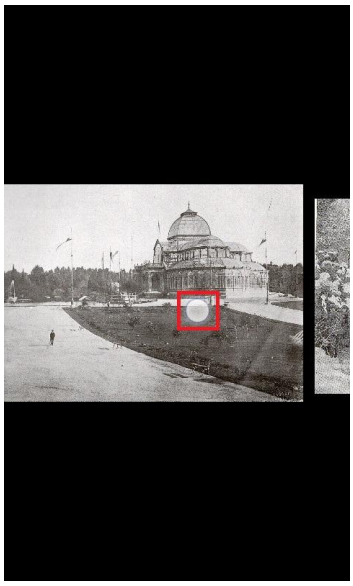


Figura B.25

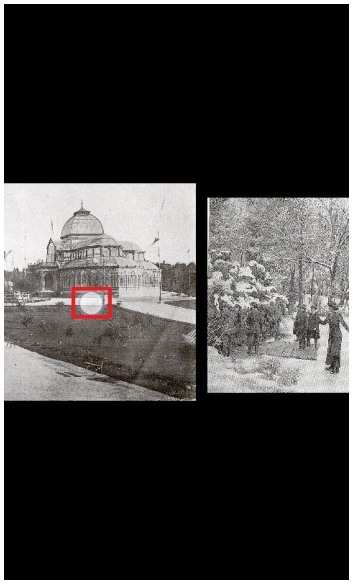


Figura B.26

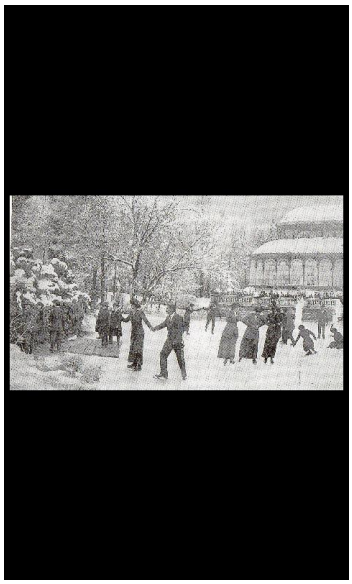


Figura B.27

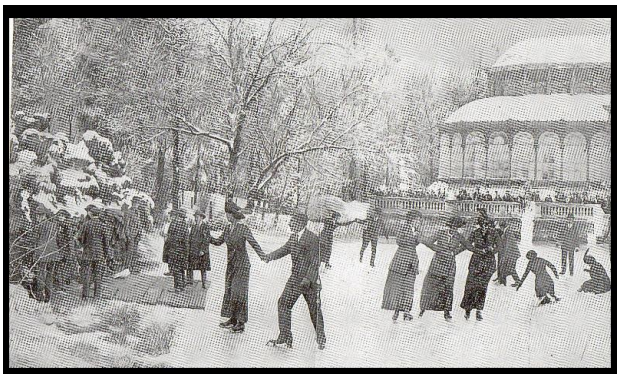


Figura B.28

Figura B.29

Figura B.30

Figura B.31

Figura B.32

Figura B.33

Figura B.34

Anexo C - Historial de bugs y mejoras realizadas según versión

ID	Fecha	Descubridor	Descripción del bug	Autor
1	15/03/2013	Alberto	Los textview del audio no se actualizan	Alberto
2	20/03/2013	El cliente	El botón de Saber+ debería ser un 'Sabías que' o un 'Curiosidades'	Víctor
3	27/03/2013	Alberto	Error de la aplicación al limpiar la caché del ImageList	Alberto
4	27/03/2013	Alberto	El Scroll de la pantalla de Información- “Saber+” elimina parte del texto	Víctor
5	27/03/2013	Alberto	Hay imágenes que ocupan cerca de 4 MB	Alberto
6	30/03/2013	Alberto	El mapa a veces no accede al servicio de Google	Miguel y Alberto
7	03/04/2013	Alberto	La descripción de las imágenes sale como 'Itemi-ésimo'	Alberto
8	04/04/2013	Guadalupe	Activar el audio directamente al pulsar sobre el botón Audio del menú principal o de un punto específico	Alberto
9	04/04/2013	Guadalupe	Cambiar el nombre de “Parada” en la lista por “Punto”	Alberto
10	04/04/2013	Guadalupe	Poner barra de progreso al audio en reproducción	Alberto
11	04/04/2013	Guadalupe	En la activity del audio cambiar el nombre de Repetir por Reiniciar	Alberto

12	04/04/2013	Guadalupe	Al pulsar el botón ATRAS/VOLVER del móvil dentro de cada punto, ha de regresar a la lista de puntos	Alberto
13	04/04/2013	Guadalupe	Al pulsar el botón Anterior dentro de cada punto, volverá al punto anterior. Desde el punto1 VOLVERA A LA LISTA	Alberto
14	04/04/2013	Guadalupe	En la activity de cada punto cambiar el nombre de Más Cosas por Sabias que	Víctor
15	04/04/2013	Guadalupe	Poner titulo en las imágenes y un año descriptivo en el titulo	Alberto
16	04/04/2013	Guadalupe	[MEJORA]Añadir comentarios explicativos en el activity de cada imagen	Se ha dejado igual
17	04/04/2013	Guadalupe	Linkar bien cada ítem de la lista para que cargue la imagen correspondiente	Alberto
18	04/04/2013	Guadalupe	Cambiar el diseño del color tanto del título como del contenido de los textos de cada punto, de introducción y de ¿Cómo usar?. Poner colores más vistosos, pero no agresivos. Cambiar también en ese activity el fondo de pantalla para que sea más vistoso. QUITAR el botón que hace la función de volver a la pantalla anterior	Víctor

19	**04/04/2013**	**Guadalupe**	**Al iniciar ruta, cambiar la imagen del Retiro para que se cargue una imagen más actual que nos pasaron/pasaran en el Retiro.** **Hacer que se cargue una imagen de alto de mayor resolución en la imagen sobre la cual podemos hacer zoom.**	**Víctor**
19 BIS	**02/04/2013**	**Víctor**	**Ambas imágenes podrán llevar dibujada el recorrido del Itinerario en color Rojo, por ejemplo**	**Víctor**
20	**04/04/2013**	**Guadalupe**	**[MEJORA]Hacer que el camino de la ruta esté bastante bien detallado (+ puntos en intersecciones de la ruta)**	**Miguel**
21	**04/04/2013**	**Victoria / Guadalupe**	**Cuando se le da al play en el audio no debería poner el icono de Play, si no el de pausa**	**Alberto**
22	**06/04/2013**	**Alberto**	**La segunda foto del punto 4 no carga**	**Alberto**
23	**06/04/2013**	**Alberto**	**[MEJORA]Las imágenes de la galería no tienen zoom ni márgenes**	**Víctor**
24	**07/04/2013**	**Alberto**	**NullPointerException al reiniciar un audio en curso**	**Alberto**

25	11/04/2013	Víctor	Poner texto de uso de aplicación en lugar de loremipsum	Alberto
26	12/04/2013	Víctor	En la lista de puntos la imagen del Punto 2 hay que cambiar una imagen de la fuente de las campanillas	Víctor
27	12/04/2013	Víctor	Cambiar imagen del botón sabias que	Víctor / Miguel
28	15/04/2013	Víctor	Revisar márgenes activities Imágenes	Víctor
29	15/04/2013	Víctor	Revisar márgenes activities audios y cambiar disposición botones	Miguel
30	17/04/2013	Alberto	Las imágenes o sonidos no alojados en un servidor sueltan excepciones	Alberto
31	17/04/2013	Alberto	[MEJORA]Se deben formar las rutas de url con todos los servidores. Esta mejora es dependiente del bug #30.	Alberto
32	21/04/2013	Alberto / Guadalupe	Texto 'Saber más' cambiar a 'Información', ya que Saber más y sabías qué puede dar lugar a confusión	Víctor
33	21/04/2013	Alberto	No sé por qué la apk de Miguel traza bien la ruta, pero sin embargo si exporto yo, no la traza.	Alberto

34	25/04/2013	Víctor	En las activities de información distribuir mejor los textos con párrafos con puntos y aparte etc.	Víctor
35	28/04/2013	Víctor	IMPORTANTE. En la pantalla donde tenemos el mapa con los tres botones: Punto de inicio, Punto Más próximo y Elegir punto; EL BOTÓN PUNTO MAS PROXIMO NO TIENEAUN AGREGADA SU FUNCIONALIDAD...	Alberto
36	05/05/2013	Alberto	Expedición: Me he encontrado con un fallo a la hora de trazar el camino por dentro del Retiro. Las rutas iban por otro lado. Las pruebas están en los vídeos del Dropbox	Alberto
37	05/05/2013	Alberto	Había dos puntos con las mismas coordenadas	Alberto
v.1.0 --- 08/05/2013				
38	01/05/2013	Alberto	Desactivar el zoom por defecto al ver una foto. Si se activa, que únicamente sea en esa foto...	Alberto
39	07/05/2013	Alberto	Hay algunas cosas de menús que sobran y que no tienen ningún efecto y por tanto hay que quitarlas	Alberto

40	**09/05/2013**	**Alberto**	**El sonido de repente al terminar se queda con el 100% de porcentaje**	**Alberto**
41	**08/05/2013**	**Alberto**	**El proyecto del parseador no estaba escribiendo en el fichero el separador entre puntos y aristas al leer del fichero de texto**	**Alberto**
42	**08/05/2013**	**Alberto**	**[MEJORA] Ahora hay un marcador de la puerta de Retiro más cercana sí y solo sí el usuario está por dentro del Retiro.**	**Alberto**
43	**08/05/2013**	**Alberto**	**[MEJORA] Bajado el tiempo de petición al GPS a la hora de calcular la posición para realizar la geolocalización del usuario la primera vez. Luego se pone que actualice cada sesenta segundos para un tener un compromiso con la duración de la batería.**	**Alberto**
v.1.1 --- 11/05/2013				
44	**11/05/2013**	**Alberto y Víctor**	**[MEJORA] Minimizar el tamaño de la aplicación quitando imágenes que no se usen**	**Alberto**
45	**11/05/2013**	**Alberto**	**Hacer la ruta donde el plano y si se puede que se vea bien**	**Alberto**
46	**11/05/2013**	**Alberto**	**Arreglado un bug a la hora de poner la foto en el plano. Se quedaba sin memoria 'para aplicaciones' si se miraban todas las imágenes del retiro y posteriormente se volvía al plano y no se abría.**	**Alberto**

47	11/05/2013	Alberto	En cada punto que haya una opción de geolocalizar o de utilizar internet se ha quitado el modo pantalla completa porque las personas mayores pueden no saber cómo activar el GPS o la tarifa de datos si no está visible la barra del estado.	Alberto
48	11/05/2013	Alberto	[MEJORA] Optimizado el coste de la apk. Sólo ocupa 3MB!	Alberto
49	12/05/2013	Alberto	Sabías que... también con puntos y aparte	Alberto
v.1.2	12/05/2013			
50	11/05/2013	Alberto	Hacer un cómo usar más explicativo; menos técnico y mas entendible por personas de cualquier tipo	Víctor
51	07/05/2013	Alberto, Miguel y Víctor	Algunas opciones de los layouts para formato tablet están fallando	Víctor
52	14/05/2013	Reunión 14/05/2013	El logotipo de la pantalla principal debe tener el nombre de 'jardines del Retiro' más grande	Miguel
53	14/05/2013	Reunión 14/05/2013	Revisar contenido de lo texto aportados por Alberto, modificar formato y añadir los iconos de los botones del mapa en el apartado ¿Cómo usar?	Víctor

54	14/05/2013	Reunión 14/05/2013	Arreglar que el GPS no pida tantas peticiones al dispositivo del usuario, si no primero una para la geolocalización y más tarde cada 60 segundos para actualizar su posición	Alberto
55	14/05/2013	Reunión 14/05/2013	Dar un mensaje mejor que el Out of memory del ViewPager...	Alberto
56	14/05/2013	Alberto	Probar la mejora de GPS en el dispositivo de Víctor	Miguel
57	24/05/2013	Alberto	Ocultar errores de depuración al usuario (que no eran errores, simplemente informes)	Alberto
v.1.3 --- 24/05/2013				
58	14/05/2013	Reunión 14/05/2013	Modificar el título de los árboles en el apartado Sabías qué... Estamos a la espera de Alberto Díaz.	En espera
59	14/05/2013	Alberto	Probar la mejora de GPS en el dispositivo de Miguel	Miguel
60	24/05/2013	Miguel	Ajustar tiempos de time_out en la selección de servidores	Alberto
61	24/05/2013	Alberto	Convenio de nombres de las clases y distribución de paquetes	Alberto
v.1.31 --- 24/05/2013				

Anexo D - Conjunto de Actas de Reunión del grupo

En este anexo se informa de todas las notas concluidas tras las reuniones, así como de decisiones de diseño acontecidas en las mismas. Dejamos constancia de los temas a tratar, del desarrollo y de las ideas que han surgido durante la reunión. Todas las actas están hechas en LaTeX y empiezan a partir de la página siguiente.

Datos de la reunión

- Lugar: El parque del Buen Retiro de Madrid
- Fecha: 1 de marzo de 2012
- Secretario: Alberto Segovia

Lista de asistentes

- Miguel Bueno
- Victor Torres
- Alberto Segovia
- Guadalupe Miñana
- Victoria López
- Carlos, personal del ayuntamiento y más personas

Orden del día

- Toma de contacto con el personal del ayuntamiento y del retiro. Explicación del proyecto, toma de requisitos y análisis posterior para el desarrollo de la aplicación Android *Itinerarios*

Desarrollo de la reunión

Importante explicación recogida en los apuntes, en los que se explica, más o menos, qué hay, qué es lo que se necesita y qué es lo que se debe desarrollar

- La aplicación se debe desarrollar en recorrido por puntos de interes
  - Las rutas ya estan establecidas
  - El público general que visita el Retiro es más de fines de semana
  - Existen distintas rutas con visitas específicas: la idea es que se visite desde el punto de vista paisajística y medioambiental con la excusa de ser guiada
- La senda botánica ya tiene 80 paneles en el Retiro, por lo que meter más paneles para informar al visitante sería sobresaturar el paisaje
  - La historia del arte del jardín no es la historia de la jardinería ni es parte del oficio de jardinero. La gente suele confundirse así que hay que hacer mella en ello
- La idea que debe ser persistente es la división de dos rutas:
  - Jardines y árboles singulares
  - Ruta monumental, estanque, palacio de Velázquez...
- Hay que recordar siempre que el Retiro es el sitio más emblemático de Madrid con más de 400 años de historia, por lo que el arte del jardín y los monumentos adquieren más riqueza
- No se tiene la obligación de autolimitarnos a un sólo itinerario, esto podría ayudarnos a crear nuestro propio itinerario dentro de la aplicación
- Las imágenes de los folletos serán necesarios para la aplicación, así como los textos.
- La mayor parte de los jardines son iguales

  - Se mantienen así desde la antigüedad

- Se comentó un aspecto adicional sobre el enfoque moderno y el antiguo de algún lugar. Ofrecer la imagen antigua en alguna posición donde se pueda capturar un código QR.
- Información para personas invidentes, textos leídos por audio. Grabar un pequeño texto informativo sobre el lugar.
- Recordar que Sistemas Informáticos son en total 45 créditos entre los 3 componentes del grupo, y por tanto, se debe 'complicar' tantos créditos como sean necesarios para ajustarse a ellos.
- Búsqueda de cobertura:
  - Un requisito es que el dispositivo móvil disponga de tarifa de datos (3G) y precise de un sistema de localización GPS.
    - Dentro de 10 años todos tendremos uno, en vistas de cómo avanza la tecnología.
  - Opción 1: Si no funciona Internet, recoger los códigos QR y descargárselos.
  - Opción 2: Descargar la aplicación antes de venir. Esto saturaría la aplicación.
- El contenido audiovisual y las locuciones de texto tendria un *Copyright* totalmente cerrado.
- Añadir distintos aspectos a la aplicación en forma de extras:
  - Información adicional como talleres
  - Música relajante de fondo mientras se visita el Retiro para ambientar a la persona que pasea.
  - Historia del jardin adicional que solamente el Retiro dispone.
- La senda botánica contiene 80 paneles dividido en 7 tramos.
  - Son espacios concretos, pero existe un problema y es que ni se paran a mirarlos.
  - Cada tipo de jardin representa estilos diferentes o un motivo por el que están ahí.
- No habría ningún problema en la conservación del código QR (se aplicaría un material anti-graffiti).
- Varios itinerarios: existen otros grupos que hacen lo mismo y nos pueden ayudar porque se enfrentan a los mismos problemas.
- Las imágenes que circulan en la red del Retiro no son propiedad del Retiro, por lo que existe el problema de que no podemos usarlas.
  - A pesar de ello, existe un banco de fotos.
- La idea es la de enfocar el Retiro como un museo vivo de los jardines del mundo.
  - Con el código QR se puede hacer publicidad o informar al turista acerca de la aplicación.
- Problemas del QR (resuelto con Geolocalización): ¿dónde colocar el QR? ¿problemas del usuario si no encuentra el QR?
  - La aplicación debería tener detalles como el de la finalización con unos versos (Carlos tiene ideas originales de este estilo)
- Idea del proyecto:
  - Publicitar el Retiro y su paisaje
  - Acabar el proyecto en febrero
  - Más tarde adecuar la memoria.
- Información de contacto (Retiro):
  - E-mail: inforetiro@madrid.es
  - Teléfono: 91 127 39 88

Datos de la reunión

- Lugar: Sala de grados de la primera planta de la FdI
- Fecha: 5 de marzo de 2012
- Secretario: Alberto Segovia

Lista de asistentes

- Grupo de Tecnología-UCM y proyectos dirigidos en él.

Orden del día

- Estructuración de Sistemas Informáticos por parte de Victoria López

Desarrollo de la reunión

Anotaciones útiles a la hora de estructurarnos para realizar el proyecto, entre ellas:

- Pensar en un nombre más llamativo (actualmente Itinerarios del Retiro o Itinerarios 'a secas')
- Se trata de estudiar el potencial de cada grupo de desarrolladores.
- Durante los cuatrimestres se desarrolla la aplicación y la memoria
  - En el primer cuatrimestre se desarrolla un prototipo
  - En el segundo cuatrimestre se corrige el código, se elabora la memoria y se desarrolla la aplicación, la cual ya no deberia ser un prototipo.
- Las reuniones individuales son más específicas que las grupales
- Elaboración de un Power-Point acerca de:
  - ¿Qué es lo que vamos a hacer?
  - ¿Qué herramientas voy a usar?
  - ¿Qué conocimientos tengo?
- Nos deben decir las directoras sobre lo que tenemos que hablar
- Debemos detallar el aspecto de la aplicación
- Si algo ya funciona, ¿por qué no copiamos la idea?
- Se debe desarrollar un documento antes de estar desarrollada la aplicación, con un descriptor del mismo.
- Hay que recordar que desarrollamos para empresas o particulares.
- Universia se ha fijado en nuestro proyecto para adaptar los recorridos al Campus Universitario, debemos hablar con ellos en un... ¿futuro?
- El resumen ejecutivo trata de vender la idea más desarrollada a un cliente
- Otro documento será la especificación de Requisitos, donde se debe presentar bien la idea con fecha límite del 31 de enero para la posibilidad de entrar en los premios de innovación. Otras opciones son los premios de emprendedor, lo que estaría bien para nuestros bolsillos cumplir con el plazo.
- El último documento a redactar sería la memoria y las plantillas nos las dejarán por Dropbox.

Datos de la reunión

- Lugar: el parque del Buen Retiro
- Fecha: 30 de marzo de 2012
- Secretario: Alberto Segovia

Lista de asistentes

- Alberto Segovia
- Victor Torres
- Guadalupe Miñana
- Victoria López
- Carlos y más personal.

Orden del día

- Recorrido por el parque del Buen Retiro con Carlos como guía con el motivo de ver y seguir la historia del Retiro, así como futuras ideas. El recorrido empieza en el punto 0 de la Figura 1 y acaba en el 7. Iré escribiendo basándome en los puntos de ruta. También hay anotaciones en el propio mapa de mayor interés

Desarrollo de la reunión

Itinerario cronológico, linea histórica por jardines de interés:

- Punto de reunión 0:
  - Plano antiguo para ver el Retiro y notar la evolución de este a lo largo de los años y el contraste entre lo presente y el pasado.
  - Se observa como se respeta la traza del camino.
- Punto de reunión 1:
  - Hacer notar la existencia de dos robles.
  - Ideas generales, tipologías y definición del jardín:
    - Jardín: Espacio creado concebido como síntesis natural entre naturaleza y arte. El jardín está inspirado en Egipto, Roma y Babilonia (y sus famosos jardines colgantes). Está dirigido a las altas esferas, vinculado a las ciudades. Se trata de un espacio propio de la aristocracia que se ha permitido otorgárselo al pueblo. La idea por aquel entonces era que la clase baja se dedicara a visitar menos la taberna y paseara más por estos ambientes más higiénicos.
    - El jardín en occidente: existen dos tendencias, la clásica y la paisajista
    - Jardín clásico: Hay láminas correspondientes a cada modelo del siglo XVIII renacentista. Era el Siglo de las Luces del Renacimiento italiano y del Barroco francés. La tendencia francés hereda de la italiana y se clasifica por formas muy racionales, matemáticas y geométricas
    - Topiaria: Arte de esculpir vegetales.
    - Jardín paisajista: Nace en Inglaterra y se caracteriza en que la persistencia de la mano del hombre **quede oculta.**

En el arte clásico francés siempre hay un eje central (esto se verá en el jardín francés del Parterre).

- En el Retiro aparecen estas dos tendencias (clásica y paisajista), especialmente en el terreno paisajista destacan cuatro tendencias: el jardín chino, el jardín japonés (y árabe), el jardín zen y el jardín islámico.
  - Jardín chino: tendencia naturalista que servirá de gran influencia para el jardín inglés paisajista. Es pintoresco, rústico y muy naturalista
  - Jardín japonés y árabe: son jardines simbólicos que suelen tener algún sentido religioso
  - Jardín zen: no es un fragmento de la naturaleza sino que es un pequeño jardín de bolsillo
  - Jardín islámico: evoca el paraíso del Corán en la tierra.

- Parada entre el punto de reunión 1 y el Parterre:
  - Esto no es un jardín, es más bien un paseo. El paseo de la República Argentina. Hay trece de cien estatuas que se utilizaron para repoblar el Palacio de Oriente.

- Punto de reunión 2:
  - El ahuehuete es un árbol con cerca de 400 años de historia de vida y está situado en el jardín francés en frente del Casón del Retiro, que era parte del conjunto palaciego del Buen Retiro. Este jardín estaba reservado para el disfrute del Rey. La Iglesia de los Jerónimos fue clave para la existencia del Retiro, pues:
    - Felipe II creó un cuartel para retirarse
    - Felipe IV lo utiliza para crear aquí su residencia
    - El Conde-Duque de Olivares quería que se divirtiera el Rey, ya que así el podría meter mano en los asuntos importantes del reino
  - El estanque es del siglo XVII
  - Los cipreses se han retocado a lo jardín japonés
  - Es importante destacar el Museo del Ejército de Felipe IV
  - El jardín francés:
    - Lo mandó realizar Felipe IV
    - Carlos III vincula
      - El Museo del Prado
      - La Real Fábrica de Porcelana
      - El Botánico
      - El Observatorio Astronómico
    - La guerra de la Independencia destruyó la parte sur del Retiro, teniendo un polvorín en todo el centro del Retiro. A partir de esta guerra, Fernando IV comienza a restaurarlo. Continuará Isabel II consiguió hacerlo del todo
    - En la primera República española, Amadeo de Saboya cedió el parque del Retiro al pueblo de Madrid siendo el primer parque público de España
      - En la plaza del ángel caído estaba la fábrica de porcelanas
      - La plaza del ángel caído es la única plaza en la cuál no hay ningún banco por la sencilla razón de la prohibición de quedarse sentado mirando al demonio
    - Los jardines de la reina: Son renacentistas y racionalistas, ya que se dispone de un control total de la naturaleza
      - Destaca una gran influencia flamenca debido al colorido de las flores
    - Felipe IV subió los impuestos para la restauración del parque del Retiro y ante ello Quevedo escribió irónicamente sátiras en contra de sus acciones controvertidas y no muy aplaudidas.

- Punto de reunión 3:
  - Destaca un eje central típico francés.

- Se quiere expresar un sentimiento solemne con el poder absoluto.

- Sentada tras el punto de reunión 3 (punto azul)
  - El jardín del Parterre:
    - El concepto solemne representa al absolutismo y es muy francés.
    - En Italia está más a la altura del hombre.
    - El parterre es adornado con pequeños arbustos recortados que no entorpezcan la visión aunque a posteriori se colocaran árboles altos que sí impiden la visión. Estos tres cedros traicionan la estética francés
    - Se realiza una visión del jardín como algo que debe ser contemplado y también pueda servir para contemplar otros lugares.
    - La topiaria viene de tupida y está caracterizada por ser realizada sobre hojas perennes (para que sea persistente más o menos con el tiempo).
    - Los cipreses no se saben cuándo se pusieron, pero aproximadamente tienen cien años y siempre se han recortado. Existen unos cuantos cipreses a los cuáles se les ha practicado la topiaria.
    - Salieri dice que el arte ceda ante la naturaleza. Se debe ir naturalizando hacia los confines del mundo.
    - Los edificios se hacen a imagen y semejanza del estilo arquitectónico del paisaje francés.

- Punto de reunión 4:
  - El jardín de los planteles:
    - El recuadro del jardín está más bajo del camino para el buen aprovechamiento del agua.
      - Hay muchísimos castaños de indias, ambiente propicio para insectos y pájaros
    - Con el paso del tiempo se ha ido geometrizando fusionándose así con el Neo Clasicismo.
    - Se dice que es el puente estético entre lo geométrico y lo paisajista

- Punto de reunión 6 (sí, el 5 nos lo saltamos por falta de tiempo):
  - El jardín chino o anglo-chino:
    - Anteriormente había aquí una ría de patinaje, existen grabados de gente patinando. Lo que implica que siempre hubo agua.

- Parada antes de la Rosaleda:
  - Eugenio de ?!?, un paisajista inglés critica al jardín clásico y su lema era que la gente debía vivir feliz con la naturaleza
  - Rousseau reclama un retorno a la naturaleza
  - El jardín inglés estaba fundamentalmente orientado a la burguesía
  - El suelo es ondulado, y todo es curvilíneo.
    - La aparición del césped surgió en esta época.
    - Los árboles de hoja caduca se pusieron de moda
    - Se crearon lagos y ríos artificiales, haciendo de los jardines un lugar que evoquen poesía.

- Duración del recorrido: 3 horas

Figura 1: Plano del Buen Retiro con el recorrido marcado

Datos de la reunión

- Lugar: Aula 13
- Fecha: 19/09/2012
- Secretario: Alberto Segovia

Lista de asistentes

- Alberto Segovia
- Victor Torres
- Guadalupe Miñana
- Victoria López
- Compañeros de otros proyectos de Sistemas Informáticos

Orden del día

- Charla general a todos los grupos de Sistemas Informáticos por parte de Victoria y de Guadalupe.

Desarrollo de la reunión

Apuntes y notas de interés tomados ese día:

- Grupo Tecnología G-Tec de la UCM trata de transferir la tecnología de la universidad a la empresa.
  - Como un puene a la empresa de los universitarios
  - Partiendo de ideas innovadoras y personas emprendedoras
- Nuestro proyecto será una Carta de Presentación para una empresa.
- Wayra de Telefónica podría estar interesado en nuestros proyectos.
- OTRI-COIE regula el trabajo de prácticas en empresa
- La idea de este proyecto es utilizar unos contundentes planes de Marketing.
  - Normalmente, los que venden un producto no saben ni cómo se diseña.
  - La perspectiva del desarrollador está en un segundo plano.
    - Se intenta que esta perspectiva pase a tener un primer plano dejando que éste desarrolle la idea del proyecto.
- Nuestro proyecto finalmente será con Guadalupe y con Victoria (hay otros proyectos que han tenido que compaginarse con otros profesores)
  - Han comentado que Univerde está interesado en adoptar el proyecto a la Ciudad Universitaria
- Normativa:
  - Memoria
  - Prototipo
  - Desarrollo

  Todo esto tiene que estar en febrero.

  - Presentación
  - Tribunal

  Plazo sobre marzo a junio

- Partes a determinar en la presentación:
  - Definición del problema
  - Documentación del problema
  - Estado del arte
    - Novedad e innovación de nuestro proyecto
  - Se base en cinco transparencias:
    - Atractivo
    - Que se entienda
    - Diferenciador
- Acerca de la utilidad de las reuniones generales:
  - Será mensual
  - Para llevar un control
  - Preguntar dudas y ofrecer soluciones
- Una posible división de tareas sería la siguiente:
  - Interfaz
  - Base de datos
  - Desarrollo de algoritmos específicos
- Documentación:
  - Dossier
  - Memoria
  - Resumen ejecutivo
  - Plan de marketing

Datos de la reunión

- Lugar: Mesas de la primera planta de la facultad
- Fecha: 04/10/2012
- Secretario: Alberto Segovia

Lista de asistentes

- Alberto Segovia
- Victor Torres
- Miguel Bueno

Orden del día

- Toma de contacto
- Decisiones con criterio de diseño bastante generales
- Esbozos de cómo sería la aplicación
- Dificultades de lo que quiere el 'cliente'

Desarrollo de la reunión

A lo largo de la reunión nos surgieron bastantes dudas sobre muchas cosas que se detallarán al final, pero empezaremos por las decisiones tomadas:

- Mejor versión de Android, nos hemos decantado por la versión 2.3 (Gingerbread) por ser la que más es usada por el 80 % de los móviles Android y poder desarrollar con las siguientes tecnologías:
  - Decodificación de audio
  - Google Maps
  - GPS
- Los códigos BIDI son cerrados mientras que los QR son abiertos, por tanto hemos seleccionado este último código como propuesto por razones de facilitar el desarrollo
- Posibilidad de añadir tres versiones a la aplicación (necesitamos debate con las directoras):
  - Tarifa de datos incluiría el uso de utilizar Google Maps
    - Hemos comprobado que hasta los caminos están bien mapeados, por lo que esta es la mejor opción, ya que muchos móviles disponen de 3G
  - Mediante el mapeado de coordenadas, únicamente con la tecnología GPS podríamos utilizar las coordenadas relativas a la tierra, fijar una franja con el Retiro como delimitación y no permitir que la aplicación funcione si no se está dentro de tales delimitaciones.
    - No se sabría por dónde decir cómo ir al siguiente punto de una ruta (no sería correcto hacer nadar al usuario en el estanque, por poner un ejemplo)
  - Mediante códigos QR se podría esclarecer el punto en el que estamos.
    - ¿El ayuntamiento los pondría? ¿y de ser así, cuándo? ¿nos daría tiempo a testear que todo va bien? Esto es un riesgo del proyecto.

Según el tipo de versión elegida, la aplicación sería más o menos pesada, ya que si no tiene internet el móvil en algún lugar se tendría que almacenar el contenido audiovisual. Si no hay internet, todo se podría meter en una base de datos sin ningún problema, aunque el peso de la aplicación podría ser del orden de decenas de *MB*.

- En caso de elegir la tarifa de datos, ¿el ayuntamiento de Madrid nos proporcionaría un servidor para acceder a consultar los datos o nos tendríamos que buscar nosotros uno gratis (si los hay) y con limitaciones? Esto es otro riesgo del proyecto.
- Ocurrencias fructíferas de la reunión: incluisión de fotos panorámicas a la hora de mostrar el contenido.
- ¿Carlos estaría dispuesto a poner la voz al texto de los puntos de la ruta?
    - Las personas prefieren oir a alguien con una voz agradable y muy unido con el tema, ya que si sabe de lo que habla, le da una interpretación bastante mejor de lo que podríamos hacer nosotros tres
- ¿Qué texto sería correcto poner en el móvil? Suponemos que Carlos puede ayudarnos

Datos de la reunión

- Lugar: Sala de Reuniones de la planta primera de la facultad
- Fecha: 9 de octubre del 2012
- Secretario: Alberto Segovia

Lista de asistentes

- Miguel Bueno
- Alberto Segovia
- Víctor Torres
- Victoria López
- Guadalupe Miñana

Orden del día

El objetivo de esta reunión era contar a nuestras directoras del proyecto diferentes decisiones, así como preguntar acerca de dudas:

- Utilización de la versión 2.3 (Gingerbread) de Android
- Opiniones de la presentación
- Lugar de alojamiento de los datos de la aplicación, ¿servidor virtual, del ayuntamiento?
- Decisión de cambio de BIDI por QR
- Utilidad/Inutilidad de los códigos QR
- Carga pesada de la aplicación, soluciones.
- Fotos panorámicas
- Selección de textos en la aplicación

Desarrollo de la reunión

- Versión 2.3 (Gingerbread) de Android: estan de acuerdo porque el usuario más propicio a usar nuestra aplicación serían personas más bien mayores que no cambian demasiado de móvil. La idea de que abarque más gente también parece muy aceptable.
- Acerca de la presentación fueron muy claras:
  - Falta bastante del estado del arte, hay que ponerse a documentar y a investigar:
    - Qué es lo que hay
    - Qué es lo que ofrecen
    - Qué es lo que no convence
    - Qué es lo que ofrecería nuestra aplicación
    - Qué es lo que se puede *adaptar* a nuestra aplicación de otras ya existentes
      - Si algo funciona y se utiliza, ¿por qué cambiarlo y marear al usuario?
  - Reparto de trabajo
    - No hace falta que fuera muy fiel, seguramente todos haremos un poco de todo, pero para ir empezando haciendo algo.

- Cambiar *Limitaciones* por *Requisitos*
- Cambiar una *visita guiada* por *desarrollo de una herramienta que realice una visita guiada*
- Definir funcionalidades
  - Moverse con la geolocalización
  - Información del entorno
- Reparto de tareas: tratar de utilizar los conocimientos adquiridos de la asignatura Ingeniería del Software
- Prácticas de campo: toda acción y comprobación que se realice en cualquier sitio, debe quedar documentada en algun lado.
  - Como en las transparencias
  - Y supongo que como todas estas actas que se están realizando

- Acerca del servidor y de la base de datos
  - El ayuntamiento de Madrid nos está dando largas, por lo que Victoria va a proponer que nos lo puedan proporcionar tras firmar un convenio.
  - Mientras tanto, Victoria propone que utilicemos una base de datos virtual (y supongo que utilizando un host gratuito y con limitaciones)
    - SQLite se manifiesta como la mejor opción, ya que se utilizó para proyectos como Recycla.me y el proyecto de Ingeniería del Software (grupo A) de la yincana de códigos QR
      - Estos últimos no han tenido muy buenas palabras acerca de la base de datos, no se sabe si porque en realidad lo era o porque el encargado de usarla no tenía muchas luces.
    - La base de datos debe ser:
      - Ágil, no se puede permitir el exceso de carga y descarga durante la aplicación. Durante el transcurso de diferentes puntos de ruta y mientras va paseando de un punto a otro, podría ir descargando los textos, fotos y audios del siguiente punto sin que el usuario se entere, para mayor anticipación de los acontecimientos (se me acaba de ocurrir)
      - Siguiendo el modelo relacional. No es totalmente necesario que esté fuertemente normalizada, pero tampoco se debe hacer una chapuza
      - Debe borrar lo que descarga de alguna manera, ya que puede saturar el dispositivo móvil
  - Victoria nos propuso una solución híbrida en cuanto a carga de base de datos y documentos propios de la aplicación:
    - Los archivos más comunmente utilizados deben estar en la aplicación para que ésta sea rápida y, por supuesto, no pese demasiado.

- La idea de las fotos panorámicas les gustó, además, se pueden sacar de Google Street View sin ningún problema, ya que son abiertas.
  - Las imágenes de Carlos deben servirnos para compararlo todo con lo que había en el pasado.

- Selección de textos importantes:
  - Debe haber distintos niveles de *tostón*. No todo el mundo se lee todo lo que pone y probablemente la mayoría querría un breve resumen locutado. Pero debemos satisfacer los niveles de todos los usuarios. En principio quedaría así:
    - Resumen del punto, con distintas características (por ejemplo en el punto del palacio de cristal):
      - Resumen de algo (palacio de cristal, vegetación del estanque del palacio de cristal, vegetación de esta zona)...
      - Locutación de este resumen
      - Saber más (y se iría al texto grande y genérico, para que los más avezados en el tema puedan quedarse satisfechos)

        - ◇ Creo que no se hará audio de este texto tan grande, dependerá de Carlos.
        - ◇ Sección curiosidades de este punto

- Utilidad de QR:

    - Es un apoyo a la herramienta que sirve para publicitar la aplicación y para que el usuario tenga la noción de la existencia de que ya hay algo que realice visitas guiadas. Además, también sirve para capturar puntos, aunque hasta que no se pongan los QR con la geolocalización es más que suficiente.

Asignación de trabajo

- Tras la reunión y en consecuencia con los que se nos da bien hacer, la asignación de tareas fue, en primera instancia (porque no hay diseño, sólo requisitos), la siguiente:

    - Miguel Bueno: interfaz
    - Alberto Segovia: bases de datos
    - Victor Torres: desarrollo de algoritmos específicos

- Hay que hacer constar que, a pesar de lo citado anteriormente, las tareas se rotarán o se fusionarán en caso de ser necesarias

Datos de la reunión

- Lugar: Ayuntamiento de Madrid, C/Montalbán 1
- Fecha: 11/10/2012
- Secretario: Alberto Segovia

Lista de asistentes

- Alberto Segovia
- Guadalupe Miñana
- Victoria López
- Belén (Recyclame)
- Sergio (mapa de recursos de interés)
- Héctor (camino al cole)
- Carlos (escuelas temáticas)
- Marisol: Jefe de departamento menars@madrid.es
- Nines: toribiocan@madrid.es 915884613
- Rafa: ruizlr@madrid.es 636017593 915884617

Orden del día

- Reunión en general con todos los grupos que trabajan con el ayuntamiento
- Decisiones propias de nuestro proyecto, como la comunicación del uso de servidores, utilización de QR en lugar de BIDI y actualización del personal del ayuntamiento

Desarrollo de la reunión

A lo largo de la reunión muy general, se intenta definir un poco todos y cada uno de los proyectos. Como el nuestro estaba ya bastante definido, no se ha dedicado mucho a él y sólo ha habido que matizar ciertas cuestiones relevantes como las siguientes:

- Servidor: Se ha firmado por fín el convenio y citarán a Victoria a partir del lunes con una persona que lleva toda la informática del ayuntamiento
    - Victoria nos ha dicho (nuevamente) que usemos SQLite para uniformizar la manera de compatibilizar las bases de datos de todos los proyectos para ahorrarnos problemas por nuestra cuenta.
- Idea de otros proyectos: Posibilidad de subir fotos del Retiro por redes sociales. Integrar Facebook y Twitter a la herramienta, ya que es una manera bastante eficaz de publicitar una aplicación. Además, esto haría que los ciudadanos colaborasen con fotografías para la aplicación (esto parece más complejo a simple vista)
- Usabilidad para invidentes: Hay que cerciorarse de que los smartphone para invidentes pueden utilizar nuestra aplicación de manera correcta. Si no, no vamos a saber cómo un invidente puede pulsar en el audio locutado, por ejemplo.

- QR: Están de acuerdo en utilizar QR en lugar de BIDI (no sabían la sutil diferencia que hay). Eso sí, las pegatina de los QR podrían tardar (lo ha afirmado Rafa), por lo que debemos tenerlo en cuenta.
- Carlos no sigue trabajando en el Retiro, ya que parece ser que el contrato no se ha prorrogado y a partir del 15 de noviembre sabremos qué personal nuevo aparecerá en el proyecto.
  - Este es un riesgo muy fastidioso, pues Carlos es aficionado a la fotografía y nos iba a servir también para el audio locutado
  - Cabe la posibilidad de que ofrezcan a Carlos entrar en el equipo
- Rafa, nuestro usuario final que llevará nuestro proyecto, ha intervenido mucho en el enfoque de las rutas del Retiro
  - En principio sólo debería haber dos: itinerario de jardines e itinerario de estatuas e historia.
  - El itinerario irá en orden cronológico.
    - He propuesto una fusión de los dos, pero me ha dicho Rafa que sería demasiado. De todas maneras, se puede pensar la manera más relajadamente y proponérselo.
  - Una futura senda sería la senda botánica.
- Nines nos ha advertido de que debemos obtener audio libre o abierto para ser utilizado en la aplicación. Con el proyecto de Ingeniería del Software no tuvimos demasiado problema con la música.

Datos de la reunión

- Lugar: Aula 7, Facultad de Informática
- Fecha: 31/10/2012
- Secretario: Alberto Segovia

Lista de asistentes

- Miguel Bueno
- Alberto Segovia
- Víctor Torres
- Guadalupe Miñana
- Victoria López
- Inmaculada Pardines
- Proyecto Recyclame adultos
- Proyecto Mapadrid
- Proyecto Camino al cole
- Proyecto Escuelas temáticas

Orden del día

- Reunión en general con todos los grupos que trabajan con el ayuntamiento

Desarrollo de la reunión

Apuntes para mejorar la presentación.

- Requisitos: se nos ha olvidado el espacio necesario para la aplicación
- Prueba y testeo: poner móviles con los que se va a testear, p. ej. el mío es un Samsung Galaxy S II
- Traspa de preguntas: es necesario ponerla al final
- Definir la idea del principio, (poner el estado del arte tras hablar de las características). Hay que venderla mejor
    - ¿Seguimos un proceso unificado? ¿basado en iteraciones?
- Hay que quitar las cosas que están en rojo de la presentación, sobretodo aquellas que pintan mal las aplicaciones ajenas
- Nunca se debe reguntar a la audiencia ni decir que estas nervioso
- Esclarecer cuál es el objetivo del proyecto

Apuntes sobre las faltas de ortografía.

- Los puntos al final de frase en las transparencias sobran
- No poner un espacio antes de que llegue un paréntesis es una falta de ortografia

Apuntes sobre la entrega final de las transparencias y presentación de proyecto.

- Definición del proyecto. Estado del Arte
- Parte técnica (bases de datos, interfaz, algoritmia...)
- Muestra. En nuestro caso será un video simulado, ya que no tenemos el Retiro en la sala de Grados

Apuntes sobre la planificación.

Es mejor planificar por meses que por semanas. A la hora de hablar con el personal del ayuntamiento se tiene que sealir de la reunión sabiendo como es la forma de la base de datos.

- A finales de noviembre debe estar ya hecho el diseño de la base de datos
- En enero debe estar ya hecho un diseño del prototipo con una apk utilizable
- Los meses dedicados a las pruebas son febrero y marzo
- Entre marzo y junio nos tenemos que dedicar a documentar y a limar los detalles últimos de la aplicación
- A principios de julio es la presentación

Datos de la reunión

- Lugar: Ayuntamiento de Madrid, C/Montalbán 1
- Fecha: 23/11/2012
- Secretario: Víctor Torres

Lista de asistentes

- Nuria: ntaibo@sma.com.es
- Alberto: adiez@sma.com.es
- Alicia
- Nines: toribiocan@madrid.es 915884613
- Maria Angeles
- Rafa: ruizlr@madrid.es 636017593 915884617
- Victoria López
- Guadalupe Miñana
- Miguel Bueno
- Víctor Torres

Orden del día

- Reunión general con el Ayuntamiento.
- Especificación de conceptos concretos de la aplicación a desarrollar.

Desarrollo de la reunión

Organizado bajo los siguientes puntos:

- Se aclara la necesidad de división de la temática del retiro en dos partes: sendas botánicas y sendas monumental; centrándonos en la parte botánica.
- Se habla también sobre la publicidad y difusión de la aplicación mediante códigos QR se intentará la colocación de dichos códigos en paneles en las principales puertas del retiro.
- Alberto comentó posibles problemas de cobertura de datos en distintas áreas del Retiro, por lo que se tendrá que llevar a cabo una prueba de campo para la comprobación de la existencia de dichos puntos y de su información en la memoria o requisitos de la aplicación.
- Alberto y Nuria realizarán una comprobación y revisión de las distintas rutas y puntos del recorrido del Retiro para delimitar a la perfección lo que se quiere mostrar al usuario a través de la aplicación. Por ejemplo: En el parterre del jardín francés... ¿Nos interesa mostrar y comentar la presencia del ahuehuete?
- Realizamos unas peticiones para la realización de la aplicación:
    - Imágenes de cualquier calidad, a ser preferible 800x600. Sobre todo pedimos imágenes de gran antigüedad para una parte de los puntos del recorrido en los cuales mostraremos una comparativa de la evolución del parque.

  - Locuciones. Pensamos en quién podría hacerlas. A priori serán caseras y con material propio. Se intentará contactar con Carlos (antiguo responsable del Retiro). Sobre todo se llega al consenso de que las locuciones han de ser de calidad y con un mismo narrador.
  - Logotipos del ayuntamiento de Madrid, Retiro... para su posterior implantación en la aplicación o incluso como icono de la misma se más adelante se sugiere.

- Centrarse en la senda botánica y dejar de lado otros tipos de detalles del parque o ciudad de Madrid (apartado de la aplicación que calificamos como extras)
- Acuerdo de próxima fecha de reunión para el Martes 14 de Diciembre a las 15:30.
- Realizamos un intercambio de contactos con Nuria y Alberto.
- Rafa nos recomendó ver un documental de La 2 sobre el Retiro para integrarnos más con la temática del proyecto. Dicho documental es del programa La mitad invisible y se encuentra en la web de la televisión.

Datos de la reunión

- Lugar: El huerto del Retiro: Centro de información y educación medioambiental
- Fecha: 04/12/2012
- Secretario: Victor Torres (apuntador) y Alberto Segovia (maquetación y explicación de funcionalidades)
- Duración: Dos horas

Lista de asistentes

- Miguel Bueno
- Alberto Segovia
- Víctor Torres
- Guadalupe Miñana
- Alberto (personal del Retiro)

Orden del día

- Establecer claramente el diseño de la aplicación:
  - Comunicación del flujo principal de funcionalidad de la aplicación
  - Posibles riesgos que tuviera la aplicación
  - Decisión de la interfaz y la posible saturación por la incorporación de muchos botones
  - Acotar los límites del proyecto: centrarnos en la funcionalidad de la aplicación.
- Establecer la documentación requerida para la aplicación:
  - Edición de textos que nos proporcionan
  - Formato de los mismos

Desarrollo de la reunión

En general se estuvo debatiendo en la funcionalidad de la aplicación. Como se sabe, estas cosas son lentas y entre la opinión de cada uno, al final llegamos a un consenso que teníamos pre-establecido ya los programadores de Itinerarios. Hay que destacar que nuestra visión suele ser más bien la de informáticos y pensamos con tal mente, por lo que la reunión con un usuario que no está realmente especializado con la tecnología, fue una buena decisión justo antes de elaborar una decisión del diseño final del proyecto. Dicho esto, lo que se esclareció en la reunión viene a continuación.

La ruta ha cambiado, en comparación con la anterior:

- Existen siete puntos de interés y se empieza en el Parterre.
  - Aunque esto no es tan relevante dentro del proyecto, sí está bien delimitar los puntos antes de ponernos a programar.
- Finalmente se termina en la Rosaleda, como en la ruta anterior.
- Se ha pedido mencionar edificios brevemente en los audios o en las imágenes para guiar al lector e informarle de qué es por si tiene más curiosidad

    - Nos han denegado totalmente la petición. En la senda de paisajes, **no se comentará la existencia de edificios o monumentos representativos**

- Seguramente no hagamos la senda escultural-monumental, ya que se pide que nos centremos en la primera que es sólo jardines

    - Pese a esta insistencia, tendremos en cuenta distintas sendas, ya que la aplicación puede ser extensible hasta límites insospechados, siempre que se tenga un buen servidor. Tener en cuenta estas pequeñas cosas desde el principio, nos simplificará mucho la vida si posteriormente quieren otra ruta contratándonos después de la entrega del proyecto. Somos futuros o casi ingenieros, y debemos prever estas cosas.

- Mostrar las fotos más representativas del paisaje en el punto de interes en el que estemos.

    - No las tendremos que elegir porque somos unos ignorantes

Acerca de las fotos:

- Nos las proporcionarían ellos
- Habrá cerca desde dos hasta diez por cada punto de interés de la ruta.
- Las fotos importantes vendran acompañadas de una descripción y un posible audio

    - Las fotos de contraste entre lo antiguo y lo actual necesitan un audio o un texto representativo en el que se explique cómo ha cambiado todo y servirá para ilustrar al ciudadano de los más finos detalles

- Las fotos que no son importantes, no tendran fotos, ni audio asociados
- Todas las fotos de la categoría `Imágenes` irán en modo galería.

Nuevas decisiones de diseño:

- Habrá un audio para las fotos más representativas
- El breve resumen que habíamos pensado será aún más breve, a modo de título representativo
- Habrá un audio de nivel medio, en el que se explique bastantes cosas
- Sigue habiendo un texto extra largo que estará presente siempre si se quiere saber más
- En cada punto que estemos habrá una imagen, un título y unos botones asociados a la funcionalidad
- Ahora se podrá ir al punto anterior
- Posibilidad de incluir en el botón de menú un apartado para ir al punto anterior
- Se informarán de los consejos en un botón de como usar la aplicación
- Se añade un apartado de introducción al Retiro en modo texto y en modo audio que parte del menú principal de la aplicación

Flujo más habitual de la aplicación:

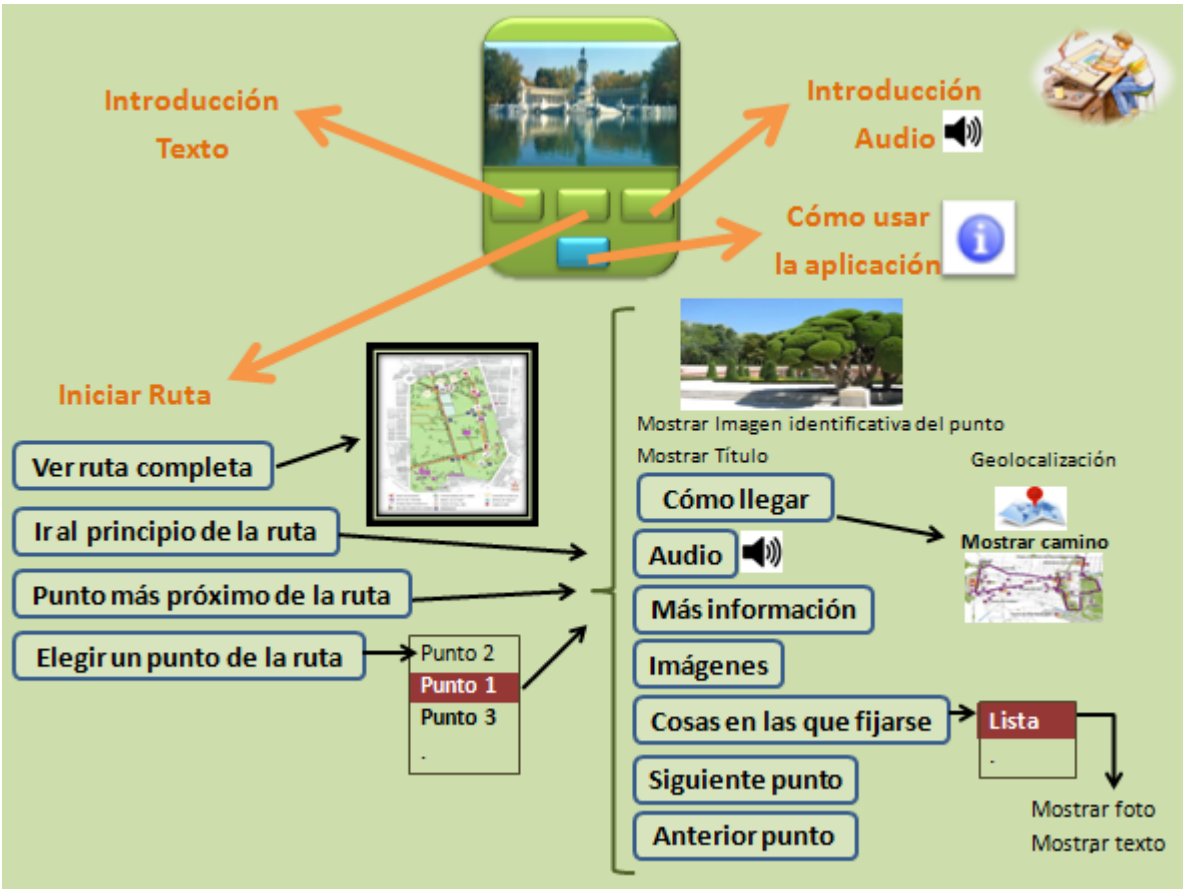


Figura 1: Diagrama funcional del uso habitual de la aplicación

Tal como observamos en la Figura 1, el uso de la aplicación es bastante sencillo. De todas maneras, describiremos más ampliamente lo acotado en el dibujo y nos centraremos en los dos temas principales, el menú principal y cada punto de interés de la ruta.

- **Menú principal**: Será lo que aparezca nada más iniciar la aplicación. Tiene los siguientes botones (tendrán alguno más, pero no son relevantes a este nivel).
  - Introducción texto: tras pulsar este botón, se desplegará de la información del Retiro, así como un texto introductorio para ir entrando en materia
  - Introducción audio: es lo mismo que el texto, pero de forma sonora y narrado por una persona
  - Iniciar ruta: esta decisión desplegará otros cuatro posibilidades.
    - Ver ruta completa: ofrecerá una imagen del Retiro y la ruta que estamos siguiendo actualmente, así como la ubicación de los diferentes puntos
    - Ir al principio de la ruta: abrirá la información correspondiente al primer punto, en este caso específico, dado la nueva ruta: el desaparecido Casón del Retiro
    - Punto más próximo de la ruta: la aplicación señalará el punto más cercano desde el lugar donde se encuentra el usuario, utilizando la tecnología GPS del dispositivo
    - Elegir un punto más cercano: si el usuario quiere volver a usar la aplicación porque un día se le hizo tarde, es posible que quiera retomar el camino de la ruta y seguir lo que empezó
    - Estos tres últimos puntos llevan directamente al punto de interés señalado que tiene un apartado propio
  - Cómo usar la aplicación: en este apartado se darán consejos de uso de la aplicación, así como restricciones, modos de ahorrar energía, cuándo y como activar el GPS, qué hacer en ciertas situaciones (mensajes propios de la aplicación, como activar el GPS al geolocalizarse y cosas de este estilo)

- **Punto de interés**: el grueso de la aplicación que debe ser homogéneo en todos los puntos de la ruta de jardines. Debe contener más o menos la misma información en todos los puntos de interés. Existirán siete puntos de interés asociados a la ruta de jardines. Pasamos a detallar todos sus puntos.

  - Imagen: será totalmente propia del punto en el que estamos para que identifique el punto con el lugar donde debería estar el usuario (no tiene por qué estar presente en el punto, se puede pasear por el Retiro de manera virtual sin tener que salir de casa, aunque la experiencia será menos grata)
  - Breve título: necesario para identificar un punto y además decir el tipo de paisaje que es, y el siglo en el que está basado.
  - Cómo llegar: un botón grande, de otro color, para que de manera resaltada se ofrezca al ciudadano la manera de llegar al punto de interés. Utilizando las APIs de Google Maps, se trazará la ruta óptima de donde se encuentre el usuario (habría que usar GPS) y la localización del punto de interés al que se quiera llegar.
  - Audio: narrará un texto de tamaño medio para que el usuario ni se entretenga leyendo en el dispositivo y únicamente se dedique al placer de mirar el paisaje mientras se interesa por la historia del lugar
  - Más información: desplegara un texto de tamaño largo, para que los más curiosos quieran leer toda la historia del punto y limar todos los pequeños detalles
  - Imágenes: desplegará una galería de imágenes, con un breve título, pero con ninguna información asociada
  - Cosas en las que fijarse (Se recomienda ver): mostrará una lista de elementos del paisaje que se deberían ver, o de pequeños contrastes de imágenes de la antigüedad con el presente. Se mostrará un texto informativo y un audio que explique qué lo hace tan característico.
  - Siguiente punto: muestra al usuario el punto de interés siguiente de la ruta.
  - Anterior punto(¡Nuevo!): muestra al usuario el punto de interés anterior de la ruta (por si se hubiese equivocado al elegir el punto el día que terminó, o por si quiere recorrer la ruta del final al principio). Con esta opción damos total libertad al usuario para que recorra la ruta como quiera.

Datos de la reunión

- Lugar: Aula 10 de la primera planta de la Facultad de Informática
- Fecha: 15h. del 05/12/2012
- Secretario: Alberto Segovia
- Duración: Hora y media aproximadamente

Lista de asistentes

- Miguel Bueno
- Alberto Segovia
- Víctor Torres
- Guadalupe Miñana
- Victoria López
- Inmaculada Pardines
- Otros grupos relacionados con el ayuntamiento:
  - Hábitat
  - Recycla.te (antiguo Recycla.me adultos)
  - Camino seguro al cole
  - Mapadrid

Orden del día

- Presentación de un descriptor de la arquitectura del sistema (a muy alto nivel)
- Presentación del flujo funcional de la aplicación (a muy alto nivel)
- Experiencias con el ayuntamiento
- Críticas, planteamientos de proyectos, compartir impresiones...

Desarrollo de la reunión

Acerca de la reunión

Antes de las presentaciones, Victoria nos preguntó distintas ideas acerca de lo que nos ha parecido el personal del ayuntamiento (a efectos de Ing. Software, el cliente). Surgieron varios temas, entre los cueles:

- Bases de datos: al final nosotros nos hemos tenido que buscar la vida
  - El IAM (Informática del Ayuntamiento de Madrid) posiblemente quiera acaparar todos nuestros datos y que todos ellos se adapten a su estándar.
    - También pensamos que se quieren colgar la medalla de nuestros proyectos, por lo que posiblemente tomemos medidas.
    - El grupo de Mapas RA (Mapadrid) y Recycla.te tendrán una reunión el viernes 21 de diciembre donde aunarán estándares y nos comunicarán qué han acordado, por si nos interesa seguir su estándar. Tendremos que saber qué es lo que nos ofrecen para que le ofrezcamos en bandeja nuestro trabajo.

- Protección de datos: en nuestro caso no nos afecta. No hay copyrights de por medio ni asuntos de protección de claves de usuario.
- El personal en general no lo tiene nada claro. Contradicen su propia idea de una manera alarmante, pese a todo, el cliente siempre debe tener la razon. El departamento de medioambiente en general no tienen mucha idea de informática y se les puede hacer creer que tienen la razón y salir nosotros ganando.
    - Factor que con el IAM puede no ocurrir.
- Privatización de las empresas: han despedido a Carlos y hemos tenido que cambiar algunas decisiones del proyecto, por lo que hemos tenido que aplicar medidas ante este tipo de riesgos.
- Participación externa de las empresas: en nuestro caso no nos preocupa demasiado, ya que nos daran las imágenes, los textos y las directivas a seguir

Victoria también destacó que siempre que nos reunamos con personal del ayuntamiento hiciésemos actas. Lo venimos haciendo desde el principio, incluso cuando nos reunimos, así que no hay problema por ello. También sirve para dejar constancia de que nos hemos reunido y que hemos dedicado ciertas horas al proyecto, ya que al fin y al cabo son créditos y su literal traducción son horas.

Acerca de nuestra presentación

Aunque hiciésemos bien la presentación en términos generales, siempre conviene hacer caso de los consejos

- Usar colores variados (solo dos es muy pobre)
- Usar colores distintos para procesos, documentos, bases de datos, etc...
- Una caja con dos barras verticales es un proceso que no tiene que ser externo.
- Se me ha olvidado por completo utilizar las BBDD en el flujo funcional a bajo nivel
- Los flujos funcionales a presentar por transparencias son muy sencillos para que gente de todo tipo lo entienda (incluídas las personas no asociada necesariamente a la informática)

Datos de la reunión

- Lugar: Mesas de la primera planta, Facultad de Informática
- Fecha: 08/01/2013
- Secretario: Alberto Segovia

Lista de asistentes

- Miguel Bueno
- Alberto Segovia
- Víctor Torres

Orden del día

- Vernos las caras tras las vacaciones de navidad
- Sistema de base de datos definitivo
    - Compromiso entre lentitud y operatividad de la aplicación
- Elección de gestor de base de datos
- Descargar imágenes a través de una url

Desarrollo de la reunión

Tras haberlo meditado bastante, hemos decidido elegir un sistema híbrido de base de datos relacional. Es decir, usaremos SQLite (interna) dentro de la propia aplicación donde únicamente se ubicará texto plano. Archivos como voz o imagen estarán alojados en un servidor y lo único que guardaremos serán las url y los datos necesarios para ello (algo así como la altura y la anchura de la foto -o píxeles-). Con esta idea conseguimos lo siguiente:

- No es necesario internet para leer todos los textos de la aplicación (el texto plano del orden de menos de 2000 líneas cabe perfectamente en una base de datos interna sin sobrecargar)
- La aplicación no será pesada, pues las únicas fotos que contendrá serán las comunes a todos, como los menús y cosas de este estilo.
- En caso de que la red de datos no esté operativa permite al usuario terminar su recorrido, gracias a la ubicación del gps (y el mapa .jpg, que vendrá por defecto dentro de la aplicación).
- La aplicación podría ser muy lenta si se dependiese totalmente de internet, por lo cual, aumenta el rendimiento de una manera sobrehumana.
- Con sólo la URL de la foto se podría descargar desde el móvil la imagen asociada a ella, por lo que se podría tener diferente hosting gratuito con imágenes replicadas por si algún servidor dejase de funcionar para que, en ese caso, buscar en otro servidor lo que queremos. No dependeríamos de un servidor en concreto, lo que nos daría cierta libertad a la hora de elegir

Aunque esta solución parece la más certera, también queremos investigar la base de datos externa donde se están alojando datos del proyecto Recycla.te

Datos de la reunión

- Lugar: Despacho de Guadalupe, Facultad de Informática
- Fecha: 31/01/2013
- Secretario: Alberto Segovia

Lista de asistentes

- Guadalupe Miñana
- Alberto Segovia
- Víctor Torres

Orden del día

- Puesta a punto y orientación sobre el balance del proyecto.
- Nuevo orden de distribución de carpetas en el Dropbox
- Apartados de la memoria a mejorar y una distribución uniforme de la misma con el resto de los compañeros.

Desarrollo de la reunión

BBDD

Se estuvo explicando a grandes rasgos el funcionamiento de la base de datos. Consta de una base de datos interna que únicamente contiene texto plano. De esta manera, se consigue que la aplicación no pese en exceso en el futuro, trayendo así las imágenes y las locuciones desde fuera, aunque no desde una base de datos externa. Me explico: dentro del texto plano podremos formar la URI donde estén los ficheros a descargar, los descargaremos, y los mostraremos dentro de la aplicación.

Ante la posibilidad de caída de algunos servidores de hosting gratuito, la opción es la réplica de los archivos a otros servidores. De esta manera, cogiendo la lista de servidores disponibles en la base de datos, a la hora de descargar un archivo de imagen o de audio, se empezará por el primero, si no se puede, seguirá el segundo, asi hasta el último. Si este último fallase no se podría mostrar la imagen, pero ya sería muy mala suerte que esto se diera.

Equipo de desarrollo

Tenemos que tener una mayor cohesión, no se ha solucionado todavia (a día 5 de febrero) la imposibilidad de integrar la bbdd con el proyecto del Retiro que va funcionando. Supongo que será por los exámenes de febrero, pero esto tiene que ir para arriba.

Proyecto

Hay dos problemas con los caminos dentro del Retiro de Google Maps:

- Logística: la furgoneta de Google no puede entrar por determinados caminos del Retiro que son demasiado pequeños o que no son transitables, por lo que no podemos contar con ellos a la hora de trazar el camino óptimo.
- Manazas: los caminos largos de las vías principales del Retiro están configurados como si fueran una autopista y, según en el 'carril' en el que se esté, es imposible cruzar la supuesta 'autopista'

Esta cuestión nos ha llevado a almacenar el camino óptimo de un punto a otro mediante capas dibujadas por nosotros mismos. Guadalupe dice tener una solución a este problema, pero no sabemos todavia de qué consta.

Documentación

La documentación se ha criticado bastante duramente, por lo que debe ser revisada y mejorada siempre que se pueda. La falta de inspiración es uno de los grandes factores que afectan al escribir, por lo que Guadalupe nos ha recomendado escribir 'lo que sea que tenga sentido' siempre que podamos, sentándonos ante el ordenador para poder decir que hemos escrito algo y en el futuro poder tirar de una base y no sobre cero.

Datos de la reunión

- Lugar: Aula 13 de la Facultad de Informática
- Fecha: 06/02/2013
- Secretario: Alberto Segovia

Lista de asistentes

- Miguel Bueno
- Alberto Segovia
- Guadalupe Miñana
- Victoria López
- Inmaculada Pardines
- Grupos de Proyecto del Ayuntamiento

Orden del día

- Estructura de carpetas del Dropbox
- Meternos caña para que avancemos en el proyecto
- Orientación general de qué progresos hemos hecho.

Desarrollo de la reunión

- A Victoria le ha gustado el que la base de datos interna contenga sólo texto plano, para, que si en el futuro se tenga que migrar de servidor, no haya muchos problemas
- Victoria ha puesto muchas pegas porque no hemos subido el código de la aplicación ni hemos actualizado la memoria. Tenemos que hacer actualizaciones periódicas de todo.
- No debemos borrar los diagramas, es mejor dejar las versiones, así se puede ver el desarrollo del proyecto sin problemas.
- A la hora de poner documentación recibida se puede dejar un fichero .txt con enlaces a páginas web donde haya información relevante como tutoriales y cosas de este estilo.

Hablando con Guadalupe

Guadalupe ha pensado dos maneras de operar según geolocalización o ir a un punto:

- Ir a un punto desde otro punto: se basa en dibujar flechas sobre las capas del mapa, así no tiramos de las rutas de Google Maps y son unos cuantos puntos bien guardados... (ya se me ha ocurrido método para guardarlo en base de datos)
- Ir a un punto desde cualquier otro punto (o geolocalización al punto más cercano): se basa en únicamente mostrar la posición del usuario, la posición del destino y una especie de flechita desde el usuario explicando la dirección al destino. Como una brújula vamos.

Entrega del viernes 8 de febrero por la mañana (reparto de trabajo)

- Miguel: diseño de la interfaz nueva con los botones que pidió Alberto (Retiro). Subir el apk actual al Dropbox y el de la nueva versión el jueves.
- Alberto: redactar este acta, seguir intentando hacer que el proyecto me compile (con la nueva key de Google Maps no funcionaba mi Eclipse).
- Guadalupe nos va a pasar un pdf donde se explica muy bien la geolocalización y los mapas, por lo que hay que recordárselo que nos lo pase.
- Se va a intentar quedar con Alberto (Retiro) sobre el 20 y muchos de abril, por lo que tenemos que tener la APK puesta a punto

Datos de la reunión

- Lugar: Aula 12 de la Facultad de Informática
- Fecha: 15/02/2013
- Secretario: Alberto Segovia

Lista de asistentes

- Miguel Bueno
- Alberto Segovia
- Victor Torres
- Guadalupe Miñana
- Victoria López
- Inmaculada Pardines
- Marián (encargada de documentarse acerca de los estándares)
- Grupos de Proyecto del Ayuntamiento

Orden del día

- Hablar del estándar de Informática del Ayuntamiento de Madrid (I.A.M.)

Desarrollo de la reunión

- Se ha hablado de todo lo que se tiene que hacer para poder tener una base de datos en el Ayuntamiento de Madrid.
    - Lo hemos añadido a nuestra Wishlist. De momento seguimos con nuestro proyecto y nuestra idea de base de datos, ya que tenemos que avanzar y tenemos unos objetivos fijados y no queremos cambiar de requisitos por el momento.
- Nos ha parecido que nadie tiene mucha idea de cómo utilizar el servidor y la base de datos proporcionada por el I.A.M.
- Nuestro proyecto no depende directamente de ninguna base de datos externa.
    - Solamente depende de archivos descargados en cualquier lugar externo, como un servidor web, y por el momento, gratuito.
    - La base de datos es lo primero que se debe realizar en un proyecto. Las interfaces gráficas son más secundarias, pero lo primordial es tener este proyecto acabado a mediados de abril.
    - Aún asi, aquellos recursos que podríamos delegar y 'meterlos' dentro del ayuntamiento, serían las imágenes y las locuciones, por lo que únicamente habría que cambiar una mínima parte de la API de la BBDD. Nada relacionado con la interfaz ni con el proyecto, gracias al adagio 'divide y vencerás'

Asuntos del estándar

Pese a que Marián está recopilando información, Victoria nos ha introducido un poco al tema de cómo acceder mediante diferentes tecnologías (un batiburrillo de nombres) de las cuales algunas nos suenan, otras nada pero de ninguna somos diestros en el tema. Según definición de la Wikipedia:

- SQL Server 2005
- Procesos Batch
- Webservices
- Hibernate

En lo que se refiere a nuestro proyecto

Dejaremos las bases de datos adaptadas para que en el futuro se pueda implementar este sistema, pero no vamos a investigar qué otras alternativas hay (que han llegado a mediados de febrero, demasiado tarde) cuando ya vemos viable el desarrollo. Adaptaremos, ya que modularmente es facilmente cambiable.

Utilizaremos la documentación de adaptación del estándar del IAM tal y como ha dicho Victoria, explicando nuestras razones para no adoptarla en la actualidad.

Datos de la reunión

- Lugar: Despacho 309 de Victoria
- Fecha: 07/03/2013
- Secretario: Alberto Segovia

Lista de asistentes

- Alberto Segovia
- Victor Torres
- Victoria López

Orden del día

- Hablar de la evolución del proyecto
- El tema del Campus de Excelencia

Desarrollo de la reunión

El Grupo de Tecnología (a partir de ahora G-Tec) de la Universidad Complutense de Madrid es un grupo no consolidado desde el punto de vista del Ministerio de Educación, por lo que tiene que ser sometido a determinadas exigencias.

El Vicerrectorado de investigación, para poder incluirlo como grupo en el Cluster de Patrimonio de Campus de Excelencia, debe estar categorizado como tal, por lo que hemos de presentar nuestros proyectos con una presentación, para que puedan valorar lo cierto o incierto de la empresa.

En lo que a nosotros respecta, tenemos que realizar una presentación exclusiva para el Vicerrectorado de investigación para posiblemente el lunes. De esta manera, tenemos que avanzar trepidantemente el diseño de la aplicación

Tras esto, destacar la opinión de Victoria de que **vamos lentos** en lo que viene siendo el desarrollo de la aplicación, por lo que nos tenemos que poner las pilas y organizarnos mucho mejor. Nuestro proyecto es interesante porque se puede extender a una Visita guiada de la Complutense, o usar una adaptación sin muchos problemas.

Detalles de la presentación:

- La primera impresión de pulsar el botón de la application debe estar siempre presente.
- Por lo general, debe estar la explicación de un botón por transparencia (dos por cada una queda muy feo)
- Dicho lo anterior, debe estar el boton pulsado en la transparencia de su explicación, para así ser más pedagógico

Detalles de la aplicación:

- Hay defectos como el darle hacia 'atrás' y quitar de la pila las activities, así que hay que ponerse serios con estos pequeños bugs porque acaban siendo importantes (y recurrentes para el usuario).
- En la presentación de los puntos, la selección de los puntos es muy pequeña, por lo que se nos ha ocurrido que fuese una lista con un scroll para que cupiese en la pantalla.

- Victoria nos ha pedido que aparezca el logo de la UCM, el logo de G-Tec y personalmente yo (Alberto) creo que también debería aparecer el de la FdI.
    - Para ello, cuando pulsamos el botón menú (donde suele salir el settings), debería aparecer un acerca de que inicie otra activity o algo por el estilo.

Datos de la reunión

- Lugar: Sala de reuniones de la planta primera
- Fecha: 14/03/2013
- Secretario: Alberto Segovia

Lista de asistentes

- Victoria López
- Guadalupe Miñana
- Alberto Segovia
- Victor Torres
- Representantes de los grupos del resto de

Orden del día

- Reunión rutinario y un mínimo control de cada grupo
- Existencia de una técnica novedosa algorítmica
- Articulo del CEDI (Congreso Español De Informática)

Desarrollo de la reunión

La reunión transcurrió con normalidad, todos los grupos expusieron sus puntos de vista, sigue habiendo incoherencias y confusión con el IAM, por lo que lo más seguro es que se implementen eso el año que viene otros grupos.

Nuestro proyecto

En lo que a nosotros respecta, nos surgió el siguiente problema:

- El mapeado de caminos a pie de Google Maps está en fase beta, por lo que no está completo y no es aplicable a nuestro proyecto. Hay zonas en las que funciona y zonas en las que no.
- A consecuencia de ello, Victoria nos dió una idea muy sencilla, ¿por qué no mapeamos unos cuantos puntos y utilizamos un algoritmo de Dijkstra para elegir el camino óptimo?
    - A todos nos pareció una idea que solucionaría nuestras limitaciones.
    - Aunque no sea una funcionalidad super importante para el proyecto Itinerarios, se nos pide por parte de la facultad para que sea un proyecto de cierto alcance.
    - Esto serviría como técnica de algoritmia novedosa y aplicada al contexto de nuestro proyecto.

Tecnologías y convenios

En lo demás, se nos habló de NDPs, como códigos de números de calles y demás en formato del IAM. En este sentido, el Retiro no está numerado, por lo que no es aplicable.

Sobre las tecnologías: hay problemas a la hora de utilizar Spring, aunque realmente no hemos indagado en cómo realizar las interfaces Webservices, asi que será cuestión de ponerse a indagar, aunque realmente esto ya no lo necesitamos.

Para el proyecto fin de carrera hay que tener alguna técnica de decisión bajo incertidumbre y técnicas borrosas de programación.

Respuestas al tribunal

Se nos ha dicho que el tribunal puede ir a pillar y que respuestas como:

- ¡Sí, ya habíamos pensado en ello en un trabajo futuro!

pueden sacarte del apuro en el momento. El tribunal quiere que te pongas nervioso y no puedas responder, por lo que hay que estar tranquilo y evitar los cafés.

Trabajo para casa

Victoria nos ha dicho que ojito con la memoria, que hay que ir escribiéndola y que ahora es el momento. Hasta que no dejemos la arquitectura estable para no cambiarla más que para pequeñísimos recortes (antes de abril debe estar terminada, o al 90 %), no se debería tocar demasiado, salvo para los resúmenes o trabajo de campo de pruebas, que eso siempre vendria bien

Articulo del CEDI, tenemos que escribir un artículo de una técnica novedosa a entregar en nuestro proyecto (araña de puntos y camino óptimo por Dijkstra ya nos valdría). La idea posterior es la de ir el dia 24 y 25 de abril a 'hablar' sobre nuestro proyecto en la Calle Gran Vía de Madrid.

Datos de la reunión

- Lugar: El huerto del Retiro
- Fecha: 20/03/2013
- Secretario: Alberto Segovia y Víctor Torres

Lista de asistentes

- Guadalupe Miñana
- Alberto Segovia
- Victor Torres
- Miguel Bueno
- Alberto (del Retiro)
- Nuria (del Retiro)

Orden del día

- Presentación de un prototipo para su posterior visto bueno
- Determinar el tema de la locución, si se ponía o no
- Logística de datos que nos faltan
- Refijar los requisitos, si fuere necesario.
- Entregas próximas
- Temas post-reunión

Desarrollo de la reunión

Se divide en subsecciones para no confundir temas.

Presentación del prototipo

La presentación en el huerto del Retiro fue un tremendo éxito, les encantó y nos dieron la enhorabuena porque estaba bastante bien hecha. Hicieron resaltar el buen diseño pulido que tenía y que íbamos por buen camino.

Locución

La locución finalmente constará en la aplicación y, actualmente, se puede encadenar audios en un mismo punto sin ningún problema, además, se puede repetir si se desea. Quién realice el audio es aún un tema a barajar.

- El audio lo debe grabar alguien que tenga una voz consonante con el contenido y con el contexto.
- El audio debe estar libre de ruidos
    - El ayuntamiento no se sabe si pondrá algún medio para esto, o si tendrá algún intérprete que pueda ayudarnos
    - Miguel puede limpiar el audio ya que es experto en el tema del sonido

La locución del menú principal contendrá el texto locutado propuesto para la introducción. El resto de puntos de manera similar.

Logística de datos

Ya se nos ha proporcionado imágenes de cada punto, textos del mismo y datos que rellenar para nuestra base de datos. Para dentro de algunas semanas nos pasarán audio, más fotos y las anécdotas del apartado "Sabías qué", que detallaremos posteriormente

Refijar los requisitos

Por suerte hemos sido buenos analistas y todo lo que les hemos enseñado era realmente lo que querían. Aún así, se ha puesto de manifiesto el sinsentido del boton 'Más cosas' que hay en el centro de cada punto. La idea es poner un apartado para las curiosidades de cada punto, ya que con estas anécdotas son con las que se queda la gente cuando viene a visitar el Parque del Retiro. El apartado se llamará 'Sabías qué...' con este nombre dejamos la puerta abierta a la curiosidad del usuario.

Entregas proximas

La próxima reunión será el 8 de mayo de 2013, con todas las funcionalidades de la aplicación funcionando (Nube de puntos by Google Maps, textos y audios disponibles, el apartado 'Sabías qué...' sería sometido a presentación exclusiva). La idea es tenerla funcional y que únicamente nos dediquemos de ese día en adelante a corregir *bugs*.

Temas post-reunión

- El desarrollo de la nube de puntos implica un cambio menor de la base de datos si se quiere tener normalizada.
- Ya se puede ir metiendo todo en la base de datos y dedicarnos a utilizar la base de datos dentro de la aplicación y a tocar lo menos posible salvo algún dato que esté mal metido.
- Las imágenes que nos ha pasado el Retiro, ¿tienen Copyright? Dentro de la aplicación se pueden recuperar con ingeniería inversa.
- Se puede investigar el asuntillo de loggear, ya que ir al Retiro y reproducir un error es muy costoso y no se puede reproducir siempre, así seríamos más precisos en la búsqueda de errores

Datos de la reunión

- Lugar: Aula 7 de la Facultad de Informática
- Fecha: 07/05/2013
- Duración: 2 horas
- Secretario: Alberto Segovia

Lista de asistentes

- Guadalupe Miñana
- Victoria López
- Alberto Segovia
- Victor Torres
- Miguel Bueno
- Resto de compañeros de apps móviles del ayuntamiento

Orden del día

- Presentación para la Catedral de la Innovación
- Poner a prueba los nervios y ver qué tal el sistema de tarjetas
- Observar si el video y las transparencias gustaba
- Limar pequeños detalles a la hora de presentar y ver si todo cuadra bien
- Hablar de cómo ir a la Catedral de la Innovación

Desarrollo de la reunión

Victoria nos habló acerca de la exposición en la Catedral de la Innovación y nos dijo que Wayra, de Telefónica e incluso TribunaComplutense van a estar allí, por lo que hay que vestir con cierta elegancia. Además, nos dijo que los representantes de la aplicación Itinerarios no iban a poder asistir por temas burocráticos (teóricamente tienen que estar en otro sitio y no se pueden multiplicar), por lo que tendremos menos presión.

Presentación del prototipo

La presentación para la Catedral de la Innovación que hicimos en el ensayo en líneas generales estuvo muy bien. Pero no fue perfecta, hay que destacar los siguientes aspectos:

- No decir nada de *Conserjería de Medioambiente*, porque estamos hablando de algo que realmente no sabemos. Mejor decir *Areas de Gobierno de Medioambiente del Ayuntamiento* y nos curamos en salud.
- ¡Hay que tener presente que el sonido esté activado! El vídeo sino podria ser una catástrofe...
- El detalle que Víctor dijo cuando estaba en marcha el video de “como hace Alberto con el dedo” es una mezcla de naturalidad y cercanía que hace que te ganes al público.
- Mejor no hablar de *Dijkstra*, directamente decir cosas de algoritmos de caminos mínimos y listo. Si preguntan, que sea después y ya le ponemos lo que queramos.

- Video: la idea sería hacer un vídeo que diese más tiempo a Víctor a explicarlo todo, pero bueno, es lo que se hizo y con ello iremos el viernes.
    - Para el Proyecto Fin de Carrera necesitamos otro video y que sea más vistoso...
- Faltan iconos que muestren de lo que habla Miguel en la introducción, por lo que se hace patente una transparencia entre medias. Quizá la de los puntos de ruta del diseño vaya ahí...
- Debemos siempre hablar en presente. Ni en pasado, ni en futuro.
- Debemos hacer mención de alguna manera al SVN.
- Y, en particular, podemos decir que hemos seguido una metodología ágil *just in time,* actualmente yo estoy usando *Kanban* con el *Trello*, podéis meteros por si os gusta `https://trello.com`. Va de tareas que hacer, tareas que estás haciendo para no perderte y de tareas ya completadas. Lo bueno que tiene es que intentas seguir haciendo cosas mientras las haces, por lo que motiva y mucho.
- Hablando con Guadalupe, me dijo que le pareceria muy bien que las trasparencias del Retiro tuviese un *minicollage* en la parte superior de las trasparencias, y que el fondo que habíamos elegido le gustaba, porque una imagen carga muchísimo.
    - Si acaso, también se ponía un pequeño fondo en los cuadros de texto, pero nada más.
- Aún queda pendiente si el nombre 'Itinerarios' es comercial o no lo es... aunque a título personal, ya podrían haberlo dicho al inicio del proyecto, ahora no creo que esto sea reversible...

Datos de la reunión

- Lugar: El huerto del Retiro
- Fecha: 14/05/2013
- Duración: 1,5 horas
- Secretario: Alberto Segovia

Lista de asistentes

- Guadalupe Miñana
- Miguel Bueno
- Alberto Segovia
- Victor Torres
- Alberto Núñez (personal del Retiro)

Orden del día

- Presentación de la aplicación a Alberto
- Distribución de la apk temporalmente a través de la página http://www.tecnologiaUCM.es
- El apartado *Acerca de...* era muy inconcluso.

Desarrollo de la reunión

A Alberto le gustó mucho la aplicación en términos generales. Guadalupe preguntó si el nombre Itinerarios le parecía bien, y hemos llegado al acuerdo de llamarlo temporalmente Itinerarios: jardines del Retiro con la motivación de que en el futuro puede haber más Itinerarios de Madrid vía móvil y que la aplicación Itinerarios fuera una super aplicación de aplicaciones, donde se pudiera elegir entre jardines del Retiro y monumentos del Retiro, por ejemplo.
Hay ciertas tareas que realizar tras esta reunión:

- En el logotipo de la pantalla principal, se debe poner más grande el 'jardines del Retiro'
- Puede haber un problema con las nuevas versiones que vayamos sacando, por lo que debemos notificar de las nuevas versiones a Victoria y a Alberto Núñez
- Rellenar en el ¿Cómo usar? los iconos del mapa y revisar su formato adaptado al mismo estilo que todos los textos
- Sabías que... el titulo del árbol debe ser más peculir
    - Estamos a la espera de lo que quieran poner, de momento se deja como está, ya que no se especificó de qué forma lo querían
- Comprobar que el mapa está mandando más peticiones GPS de las que tenemos establecidas, y en su caso, arreglarlo.
- Sobre el apartado *Acerca de...* Alberto dijo que lo hablaría con Rafa

Datos de la reunión

- Lugar: La Catedral Innova, cerca de la Vaguada
- Fecha: 17/05/2013
- Duración: 2,5 horas
- Secretario: Alberto Segovia

Lista de asistentes

- Victoria López
- Guadalupe Miñana
- Miguel Bueno
- Alberto Segovia
- Victor Torres
- Recycla.me
- Recycla.te
- Mapa de Recursos Ambientales
- Camino seguro al cole
- Demás grupos

Orden del día

- Presentación de la aplicación a la Catedral de la Innovación, del Ayuntamiento de Madrid

Desarrollo de la reunión

Presentamos tal y como se había dicho en los ensayos, fue bastante bien salvo en un asunto: hay que asegurarse siempre que el video de la presentación se reproduce antes de llevarlo. De alguna manera u otra, a veces, un video incrustrado dentro de las transparencias de una presentación de *PowerPoint* deja de verse. Hay que indagar el porqué de esta cuestión y anticiparse a que no pase más.

El desarrollo de la reunión post-presentación fue bien, sin ningún detalle relevante que dejar en acta. Hay un caso que se nos preguntó a todos en general y era lo siguiente:

- ¿Por qué habiamos elegido *Google Maps* y qué pasa con los datos de geolocalización que son sensibles a proporcionar a *Google*?
    - Héctor Alejandro, del proyecto Camino seguro al Cole, respondió que en el caso, para impedir proporcionar demasiada información para niños que van al cole por una ruta en especial, únicamente se pide un nombre en el acceso del usuario. Además, por si esto fuera poco, *Google* únicamente está enterado de las coordenadas a las que se accede, no de lo que se pinta encima de ellas. *Google* ya sabe qué buscamos y qué no en su propio buscador, por lo que intentar proteger todas estas cuestiones siempre es digno de motivación.

Datos de la reunión

- Lugar: Despacho 317, Facultad de Informática
- Fecha: 30/05/2013
- Duración: 1,5 horas
- Secretario: Alberto Segovia

Lista de asistentes

- Guadalupe Miñana
- Alberto Segovia

Orden del día

- Corrección por parte de Guadalupe de la versión 0.9 de la memoria.

Desarrollo de la reunión

Se nos ha dado la corrección y la parte a corregir de cada integrante se ha escaneado y entregado respectivamente.

Cambios

Hay que corregir muchas cosas, ya que hay algunos apartados que no han gustado. Por partes:

- **Resumen en castellano:** hay que fijarse en la plantilla de *Hábitat* que tenemos en el Dropbox. Nos debemos basar en esa plantilla debido a que es un buen resumen y cataloga todo lo que es la aplicación y los entes ajenos a ella
- **Resumen en inglés:** la traducción literal del apartado anterior.
- **Capítulo 1: Introducción**
  - Este capítulo tiene que ser redactado casi enteramente
  - **Primer apartado:** Qué es nuestra aplicación y la motivación de la misma
    - Esto ha de estar muy bien redactado porque la introducción siempre es vital
    - Una imagen de la aplicación ayudaría para ilustrar lo que estamos explicando
  - **Segundo apartado:** Al final de la introducción se debe dar un resumen de cómo va a estar resumida la memoria. De manera breve y definiendo poco a poco lo que se va a contar en cada apartado
- **Capítulo 2: Estado del arte**
  - En general este capítulo está mejor que el anterior, pero se debe seguir un cierto criterio:
    - Primera parte: Describir otras aplicaciones (**sin comparar** con la nuestra)
    - Segunda parte: Definir las caracteristicas **buenas** de nuestra aplicación, así cómo nos hemos visto influenciados por las aplicaciones anteriormente citadas
- **Capítulo 3: Especificación de requisitos, herramientas y recursos**
  - Mirar la página 39 de SocialSport para influenciarnos sobre este tipo de enfoque
  - El ex-capítulo 7 ha desaparecido y se ha fusionado con este.

- En primer lugar estarán los requisitos
- En segundo lugar estarán las herramientas y recursos
  - En general se dividirá herramientas y recursos en los siguientes dos grupos:
    - Herramientas software imprescindibles: Entorno de trabajo, librerias utilizadas y programas necesarios para el desarrollo del proyecto como tal
    - Herramientas software de ayuda: Programas útiles (SVN, Dropbox...) que han servido para hacernos la vida más fácil
    - Recursos físicos utilizados, móviles, tablets..
  - En lugar de tablas de caracteristicas, dar la referencia a la bibliografia de la tabla
  - Organizarlo mejor e hilar cada apartado con el anterior de alguna manera más natural, no tan salteado.

- **Capítulo 4: Especificación de casos de uso**
  - Sigue inamovible
- **Capítulo 5: Diseño de Itinerarios del Retiro**
  - Sin cambios
- **Capítulo 6: Técnicas Algorítmicas**
  - Sin cambios
- **Capítulo 7: Operaciones de despliegue**
  - Sin cambios
- **Capítulo 8: Manuales**
  - Sin cambios, aunque puede que se quede algo fuera debido a que ya hay un manual dentro de la aplicación
- **Capítulo 9: Bibliografía**
  - Podría ser un anexo en lugar de un capítulo
- **Anexo A - Historial de Revisiones de este documento**
  - Ha gustado la idea para ir corrigiendo en lo que se haya tocado únicamente y ahorrarnos trabajo
- **Anexo B - Historial de bugs y mejoras realizadas segun versión**
  - Sin cambios
- **Anexo C - Conjunto de Actas de Reunión del grupo**
  - Hay que quitar las portadas de cada acta porque ocupan mucho
  - Quizá hay que decir que la fecha superior del acta es la fecha en la que se redactó el acta y que la fecha de la reunión fue justamente el día en que nos hemos reunido
  - Avisar a modo de *disclaimer* que está redactado en un lenguaje informal

Comentarios generales y consejos

Hay que cambiar algunas cosas al redactar y seguir unos convenios entre nosotros:

- **Redactar en forma de pasiva impersonal:** nada de '*hemos desarrollado esto*', si no '*se ha desarrollado esto*'
- **No ser totalmente sincero, ni tampoco quitarse mérito:** si algo es facil o sencillo, no hablar de que lo es, simplemente que se ha usado eso debido a factores de interés (buena curva de aprendizaje o experiencia en ello)
- **Añadir referencias a la bibliografia**
- **Añadir referencias a las figuras**
- **Evitar las muletillas al redactar**
- **Nunca asegurar que nuestra aplicación es única y difícil de igualar**
- **Evadir problemas:** si algo es difícil de explicar o algo tiene algún inconveniente, que se pregunte. No tiene por qué estar escrito en la memoria
- **El alcance del documento es a informáticos**, por lo que no se debe tratar a las personas como personas que no tengan suficiente conocimiento de informática
- **Redactar bien y revisarlo antes de ser entregado**