



TFC Desarrollo de Aplicaciones Móviles

Juan Garrido Cobo

Enero 2013

Tutores: Ignasi Lorente Puchades

Jordi Almirall López

ANDROID

Contenido

Introducción y punto de partida del TFC.....	4
Introducción	4
Objetivos	4
Justificación elección	5
Estado del arte.....	6
Android.....	6
Backend As A Service	7
Git.....	7
Eclipse.....	8
Java	8
Especificación de requisitos.....	10
Requisitos funcionales.....	10
Requisitos no funcionales	11
Entorno de desarrollo y compilación	11
Control de versiones de código fuente.....	11
Persistencia de datos.....	11
Lenguaje de programación	11
Librerías	11
Entorno de ejecución	12
Pruebas.....	12
Documentación.....	12
Planificación	13
Metodología de desarrollo (DCU)	13
¿Qué es el diseño centrado en el usuario?.....	14
Participantes.....	14
Cronograma.....	14
Tareas	15
Estimaciones viabilidad y coste.....	16
Análisis y diseño del sistema.....	19
Usuarios y contextos de uso.....	19
Investigación	19
Perfiles de usuario.....	20

Contextos de uso.....	21
Análisis de tareas	21
Diseño conceptual.....	23
Escenarios de uso	23
Diagramas de casos de uso	25
Flujos de interacción	27
Prototipado	28
Sketches	28
Prototipo.....	30
Evaluación	33
Usuarios que realizarán el test.....	33
Tareas que realizar en el test.....	33
Preguntas sobre las tareas	34
Conclusiones resultados test.....	34
Implementación.....	36
Gestión de código.....	36
Arquitectura de la aplicación.....	37
Arquitectura cliente MapaQuejas.....	38
Persistencia de datos.....	49
Test de la aplicación final	52
Pruebas unitarias.....	54
Android Lint	56
Manual de instalación	58
Conclusiones.....	60
Objetivos pendientes	60
Software utilizado	62
Bibliografía y referencias.....	63
Agradecimientos	64
Apéndice.....	65
Test de usuarios	65
Código referenciado.....	67
Comunicación con Parse	67
Gestión SQLite.....	68

Clase Actividad Principal	70
---------------------------------	----

Introducción y punto de partida del TFC

Introducción

El proyecto actual se engloba dentro del área de desarrollo de aplicaciones móviles. De las diferentes propuestas del equipo docente se ha elegido la opción de Gestor de Mapas y GPS para plataforma Android.

La idea inicial se basa en desarrollar una aplicación para dispositivos Android, que permita a los usuarios geolocalizar, desde su posición actual, espacios o puntos, con problemas de mantenimiento, de responsabilidad municipal en un área geográfica predeterminada con el fin de poder tramitar sus quejas y/o denuncias al ayuntamiento responsable o bien informar al resto de usuarios de la aplicación de la existencia de los mismos, así como consultar los puntos dados de alta por otros usuarios.

Estos puntos se mostrarán como pines dentro del mapa y al pulsar en los mismos un cuadro de diálogo permitirá realizar las acciones oportunas: abrir un formulario de quejas, consultar los datos, tramitar la denuncia, etc.

Inicialmente para facilitar la gestión de la aplicación el ámbito geográfico se limitaría al municipio de Palma de Mallorca.

Objetivos

El objetivo principal del proyecto es crear la aplicación “MapaQuejas” descrita en el punto anterior y que la misma sea funcional. Para la consecución de este objetivo es imprescindible investigar acerca de la tecnología a emplear y gestionar eficientemente el proyecto en sus diferentes fases.

A nivel personal el objetivo es, mediante los conocimientos adquiridos previamente en mis estudios, ser capaz de gestionar correctamente el proyecto, mejorar mis conocimientos en el desarrollo de aplicaciones nativas para plataforma Android y toda la tecnología relacionada con el proyecto, y finalmente superar la asignatura.

Justificación elección

El motivo principal por el que he decidido realizar el trabajo de final de carrera en el área de desarrollo de aplicaciones móviles es el auge del concepto movilidad que existe en la actualidad y la previsible tendencia a futuro de este tipo de tecnologías.

La elección de desarrollar para Android se basa en que dispongo de dispositivos con este sistema operativo para poder realizar pruebas y en mis conocimientos del lenguaje de programación Java. Desconozco totalmente el sistema operativo IOS y el lenguaje de programación Objective-C. Previamente he realizado algún desarrollo a nivel formativo para Android con HTML5 y PhoneGapp por lo que a nivel formativo resulta más interesante aprender a desarrollar para Android de forma nativa.

El concepto de geolocalización es muy importante en la actualidad. Múltiples de desarrollos incluyen mapas en los que los usuarios pueden localizar puntos de su interés. Los mapas de Google son ampliamente utilizados. El éxito de aplicaciones basadas en mapas como FourSquare que combina los mismos con el concepto de red social, justifica el querer trabajar en esta área de conocimiento.

Estado del arte

Android

Android es un sistema operativo móvil basado en Linux enfocado para ser utilizado en dispositivos móviles como teléfonos inteligentes, tabletas, Google TV y otros dispositivos. Es desarrollado por la Open Handset Alliance, liderada por Google.

La estructura del sistema operativo Android se compone de aplicaciones que se ejecutan en un framework Java de aplicaciones orientadas a objetos sobre el núcleo de las bibliotecas de Java en una máquina virtual Dalvik con compilación en tiempo de ejecución. Las bibliotecas escritas en lenguaje C incluyen un administrador de interfaz gráfica, un framework OpenCore, una base de datos relacional SQLite, una Interfaz de programación de API gráfica OpenGL ES 2.0 3D, un motor de renderizado WebKit, un motor gráfico SGL, SSL y una biblioteca estándar de C Bionic.

Las aplicaciones se desarrollan habitualmente en el lenguaje Java con Android Software Development Kit (Android SDK), Existen otras herramientas de desarrollo, incluyendo un Kit de Desarrollo Nativo para aplicaciones o extensiones en C, C++ u otros lenguajes de programación.

Android se desarrolla de forma abierta y se puede acceder tanto al código fuente como a la lista de incidencias donde se pueden ver problemas aún no resueltos y reportar problemas nuevos.

En la actualidad existen más de 700.000 aplicaciones para Android y se estima que 1.000.000 teléfonos móviles se activan diariamente.

Android es criticado por la fragmentación que sufren sus terminales al no ser soportados con actualizaciones por parte de los fabricantes, algo que se podría considerar obsolescencia programada. Esto supone problemas para los desarrolladores que deben lidiar con la retro compatibilidad entre diferentes versiones del sistema operativo.

Resulta evidente que aprender a desarrollar proyectos para este sistema operativo mejora las perspectivas tanto laborales como académicas de cualquier estudiante de Informática o similares.

Para desarrollar sobre Android se hará uso del paquete ADT (Android Developer Tools), en su versión Bundle, que incluye el software necesario para comenzar desarrollar para esta plataforma.

Backend As A Service

Backend as a service (BAAS), o mobile backend as service (MBaaS), es un modelo que proporciona servicios Web a los desarrolladores de aplicaciones móviles y una forma de vincular el almacenamiento de sus aplicaciones a la nube. Proporciona un backend de gestión y funciones tales como la gestión de usuarios, notificaciones push e integración con servicios de redes sociales. Estos servicios se prestan a través del uso software development kits (SDK) y application programming interfaces (APIs).

BaaS es un desarrollo relativamente reciente en la computación en la nube creado por StartUps, que nace en el año 2011.

Algunos servicios de este tipo son StackMob, Parse, Kinvey o Appcelerator.

El uso de este tipo de sistemas permite prescindir para la persistencia de datos de la aplicación un sistema gestor de base de datos y el desarrollo de webservices que comuniquen la aplicación con dicho sistema. Configurar y desarrollar este tipo de modelo de persistencia de datos suele resultar costoso.

Para los desarrolladores de aplicaciones móviles resulta un ahorro en tiempo y consecuentemente coste, ya que la mayoría de BaaS siguen un modelo freemium, por lo que sólo se paga por determinadas características o si se superan ciertos umbrales de uso. Resultan adaptables a las necesidades habituales de un desarrollo de este tipo por lo que resultan una buena opción para todo tipo de desarrollos.

El elegido para este proyecto es Parse, ya que aparentemente dispone de una Api sencilla de integrar para este proyecto.

Git

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones.

Destaca su gestión distribuida. Git le da a cada programador una copia local del historial del desarrollo entero, y los cambios se propagan entre los repositorios locales. Los cambios se importan como ramas adicionales y pueden ser fusionados en la misma manera que se hace con la rama local.

Actualmente numerosos desarrollos colaborativos son gestionados con el sistema GIT.

En este proyecto se usará un sistema de control de versiones online para prevenir posibles problemas que puedan surgir en las máquinas locales así como poder revertir erratas que puedan provocarse durante el desarrollo de nuevas funcionalidades.

Eclipse

Eclipse es una plataforma de desarrollo open source basada en Java. Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Actualmente es desarrollado por la Eclipse Foundation, una organización independiente sin ánimo de lucro que fomenta una comunidad de código abierto.

En sí mismo Eclipse es un marco y un conjunto de servicios para construir un entorno de desarrollo a partir de componentes conectados, plugins... Existen plugins para el desarrollo en Java, JDT Java Development Tools, así como para el desarrollo en otros lenguajes como C/C++, PHP, Cobol, plataformas como Android, etc.

Eclipse es el IDE recomendado para el desarrollo sobre Android y se incluye de facto en el paquete Bundle de las Android Developer Tools, por lo que el proyecto hará uso de esta herramienta complementándose con los plugins necesarios para facilitar el desarrollo del mismo.

Java

Java es un lenguaje orientado a objetos que alcanzó su madurez con la popularización de Internet y que es, en cierta manera, el heredero legítimo de C++. La expansión de este lenguaje entre la comunidad de programadores ha sido vertiginosa y se ha impuesto como el paradigma de los lenguajes de programación orientados a objetos. En el entorno académico y de investigación, la enseñanza de Java ha remplazado (y está remplazando) a la enseñanza de lenguajes de programación estructurada como Pascal e incluso C, que siempre se han considerado lenguajes de elección para la introducción a la programación.

De forma resumida, Java es un lenguaje neutral, portable, robusto, estable, independiente de la plataforma, sencillo de aprender para programadores que hayan trabajado previamente con lenguajes orientados a objetos. Java puede utilizarse para realizar aplicaciones en múltiples plataformas hardware y sistemas operativos (Unix, Linux, OS/390, Windows, o HP-UX entre otros sistemas operativos para ordenadores personales o estaciones de trabajo, Android, Palm OS o EPOC entre otros sistemas operativos para dispositivos de telefonía móvil).

Una de las novedades revolucionarias que introdujo Java es la portabilidad. Sun abordó el problema introduciendo el modelo de bytecode: cuando un programa Java se compila no se transforma en un conjunto de instrucciones en código máquina nativas de la plataforma utilizada, lo cual impediría su completa portabilidad, sino que se transforma en un conjunto de bytecodes independientes de la plataforma utilizada que son leídos e interpretados por la máquina virtual Java, JVM, para ejecutar el programa. Por ejemplo, cuando se compila un programa Java en una plataforma Windows/Intel, se obtiene la misma salida compilada, el mismo bytecode, que en un sistema Macintosh o Unix.

Los requisitos de desarrollo para Android exigen el uso del JDK (Java Development Kit) en su versión 6.

El desarrollo para aplicaciones Android se realiza de forma común en Java, aunque existe la posibilidad de realizar parte de una aplicación mediante otros lenguajes mediante el set de herramientas NDK (Native Development Kit) aunque no es recomendable debido a que aumenta la complejidad del desarrollo de las aplicaciones.

Siguiendo las recomendaciones de Google el proyecto usará el lenguaje de desarrollo Java.

Especificación de requisitos

La elección del trabajo conlleva una serie de requisitos funcionales explícitos y otra serie de requisitos no funcionales implícitos que se van a presentar a continuación.

Requisitos funcionales

Los requisitos del trabajo de final de carrera elegido constan de dos módulos principales con una serie de funcionalidades requeridas. En la tabla resumen de los requisitos se ha añadido la columna funcionalidad objetiva que es la que se corresponde con la funcionalidad correspondiente que se va a implementar en la aplicación final MapaQuejas.

Módulos principales	Funcionalidad requerida	Funcionalidad objetiva
Mapa basado en Api de Google Maps		
	Localizar usuarios	Localizar puntos denunciados o consulta de puntos cercanos
	Mostrar múltiples pines	Consultar puntos dados de alta
	Configurar diálogos de los pines	Proporcionar Información básica sobre los puntos
	Aplicar acciones al diálogo de los pines	Ampliación de información, comentarios, fotos y posibilidad de enviar denuncia
	Zooming	Realizar zoom sobre los mapas
	Cómo llegar	Obtener indicaciones de la ruta hasta un punto
Gestión de GPS		
	Gestión de conversiones entre coordenadas de geolocalización-direcciones y viceversa a partir de la API de Google	Facilitar la ubicación de los puntos y su interpretación por parte de los usuarios

	maps	
	Gestión de cache local en base de datos SQLite	Almacenar datos necesarios para el correcto funcionamiento de la app y mejora de performance

Requisitos no funcionales

Aunque no se especifiquen en el trabajo, son imprescindibles para el desarrollo del proyecto.

Entorno de desarrollo y compilación

El desarrollo de la aplicación se llevará a cabo usando *Eclipse* como entorno de desarrollo con los plugins necesarios para el desarrollo de aplicaciones Android.

Control de versiones de código fuente

Se usará un sistema de control de versiones para el código fuente de la aplicación de tipo GIT, el elegido es Bitbucket de Atlassian, ya que no obliga a hacer público el repositorio.

Persistencia de datos

Los datos de la aplicación deben almacenarse en un sistema, para ello se hará uso de un backend en el que se almacenarán los objetos necesarios.

Lenguaje de programación

Para desarrollar de forma nativa para Android se utilizará el lenguaje de programación Java, requisito imprescindible para tal fin.

Librerías

Se utilizará el framework para desarrollo nativo para Android así como las librerías adicionales para el acceso a la API de Maps y al Backend Parse.

Entorno de ejecución

Se proporcionará un ejecutable en formato apk para poder instalar en dispositivos Android.

Pruebas

Se realizarán las pruebas unitarias con JUnit y los test de usuario que resulten necesarios.

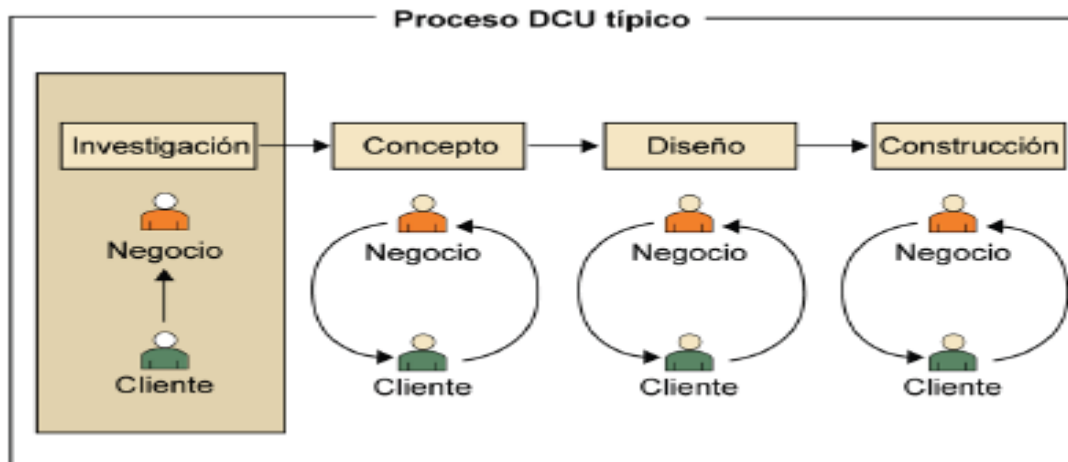
Documentación

Se proporcionará una memoria final y un vídeo de presentación del proyecto.

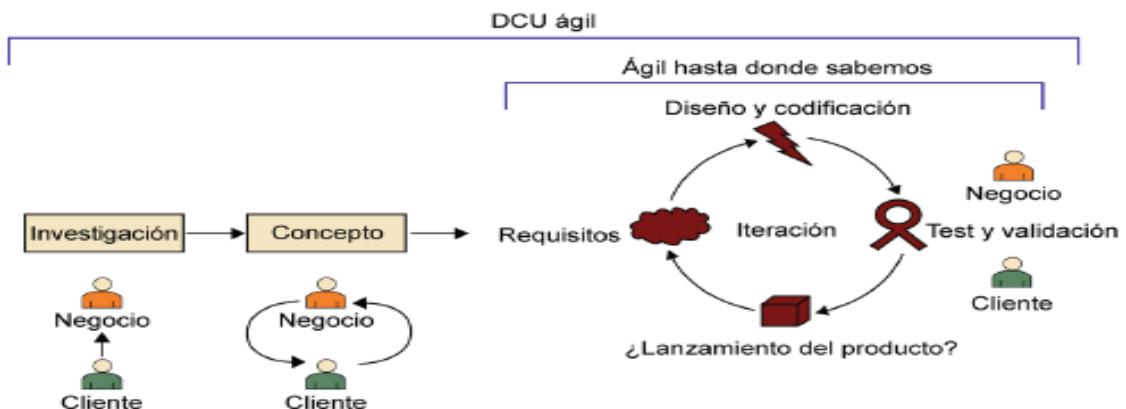
Planificación

Metodología de desarrollo (DCU)

La metodología a emplear para el desarrollo del proyecto es el diseño centrado en el usuario (DCU).



En particular este proyecto usará una variación del tipo DCU ágil. Esta metodología combina el proceso típico del Diseño Centrado en el Usuario y Agile Software Development.



La propuesta es usar el modelo DCU clásico en las fases de investigación, concepto y diseño dejando la metodología ágil exclusivamente para la fase de codificación.

La justificación de la elección de esta metodología se basa en que en uno de los entregables previos al proyecto se debía trabajar con esta metodología, por lo que resultaría contraproducente usar otro tipo de metodología, cuando se

dispone de un trabajo previo. Por otra parte no se puede obviar que actualmente la experiencia de usuario y la usabilidad son imprescindibles para el éxito cualquier tipo de producto o software.

¿Qué es el diseño centrado en el usuario?

El diseño centrado en el usuario es una aproximación al diseño de productos y aplicaciones que sitúa al usuario en el centro de todo el proceso. Así, podemos entender el DCU como una filosofía cuya premisa es que, para garantizar el éxito de un producto, hay que tener en cuenta al usuario en todas las fases del diseño. Además, también podemos entender el DCU como una metodología de desarrollo: una forma de planificar los proyectos y un conjunto de métodos que se pueden utilizar en cada una de las principales fases.

Participantes

Se considera que el proyecto debe contar con los siguientes roles como participantes del mismo:

Director de proyecto: El alumno será el encargado de la dirección del proyecto

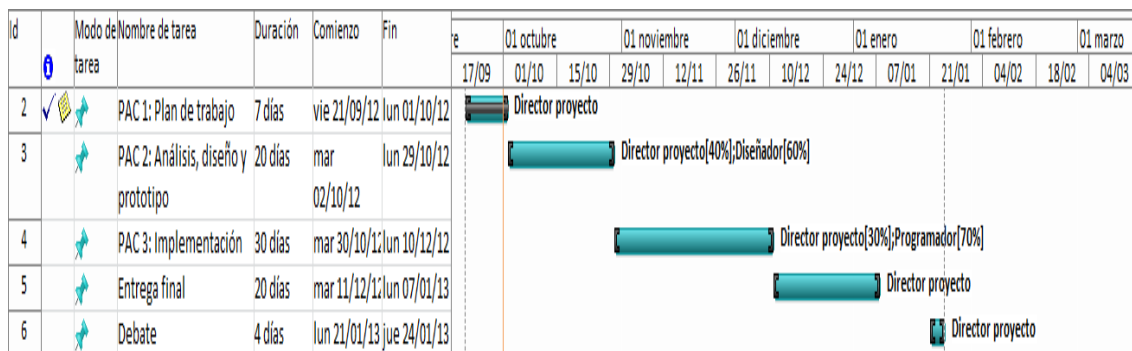
Diseñador: El alumno será el encargado del diseño del proyecto

Programador: El alumno será el encargado de la programación e implementación del proyecto.

Clientes: Consideramos clientes a los usuarios finales de la aplicación como podrían ser los testers(probadores) y los consultores del proyecto.

Cronograma

En el siguiente diagrama de Gantt se muestra de forma resumida la planificación del proyecto, sus tareas, los participantes y las fechas clave.



Tareas

A continuación se enumeran las tareas que componen el proyecto.

Nombre de tarea	Duración	Comienzo	Fin	Predecesoras	Nombres de los recursos
PAC 1: Plan de trabajo	7 días	vie 21/09/12	lun 01/10/12		Director proyecto
PAC 2: Análisis, diseño y prototipo	20 días	mar 02/10/12	lun 29/10/12	1	Director proyecto[40%];Diseñador[60%]
PAC 3: Implementación	30 días	mar 30/10/12	lun 10/12/12	2	Director proyecto[30%];Programador[70%]
Entrega final	20 días	mar 11/12/12	lun 07/01/13	3	Director proyecto
Debate	4 días	mar 08/01/13	vie 11/01/13	4	Director proyecto

A continuación se detallan de forma resumida los objetivos y entregables de cada fase.

PAC 1: Plan de trabajo

Objetivos: Definir el proyecto y realizar su planificación inicial

Entregables: Plan de trabajo, PAC 1.

El plan de trabajo servirá como guía para el resto del desarrollo.

PAC 2: Análisis, diseño y prototipo

Objetivos: Realizar el análisis de tareas, diseño y prototipo acorde a la metodología DCU.

Entregables: Documentación que incluye perfiles de usuario, contexto de uso, análisis de tareas, escenarios de uso, diagramas de flujo de interacción y un prototipo de alto nivel.

La entrega de esta documentación facilitará cumplir los objetivos finales del proyecto, ya que se estudian los potenciales clientes del proyecto, se analizan todas las tareas detalladamente y se obtiene un prototipo de alta fidelidad del proyecto. Todas estas tareas sirven como entrada de datos para la siguiente fase.

PAC 3: Implementación

Objetivos: Implementar la solución del proyecto y documentación complementaria

Entregables: Código fuente, instalables y documentación complementaria.

Es la fase final del proyecto en la que se obtiene el producto final. Puede requerir iterar con las fases anteriores en caso de detectar problemas en el producto.

Entrega final

Objetivos: Finalizar el proyecto y documentarlo.

Entregables: Memoria y video de presentación del proyecto.

En esta fase se finalizará la fase anterior en caso de no haberlo hecho y se presentará al público objetivo nuestro producto.

Debates

Objetivos: Debatir sobre los trabajos entregados.

No se ha tenido en cuenta en el coste estimado del proyecto, pero sí se ha planificado, no contiene entregables.

Estimaciones viabilidad y coste

El proyecto está claramente identificado por lo que queda pendiente analizar su viabilidad.

A nivel técnico se va a usar software y aplicaciones que ofrece el mercado tales como el SDK de Android, la Api de Google Maps, el IDE Eclipse con los plugins que resulten necesarios, un servidor para la gestión de versiones on-line como BitBucket y el Software de Backend. Aparte del hardware necesario para hacer uso del software. Esto indica que técnicamente el proyecto es viable.

A nivel operativo, se dispone de pocos recursos para la realización del proyecto lo que puede incrementar el tiempo necesario para la realización del proyecto, teniendo en cuenta la fecha límite del mismo, esto puede ocasionar que sea poco viable finalizar en el plazo estipulado.

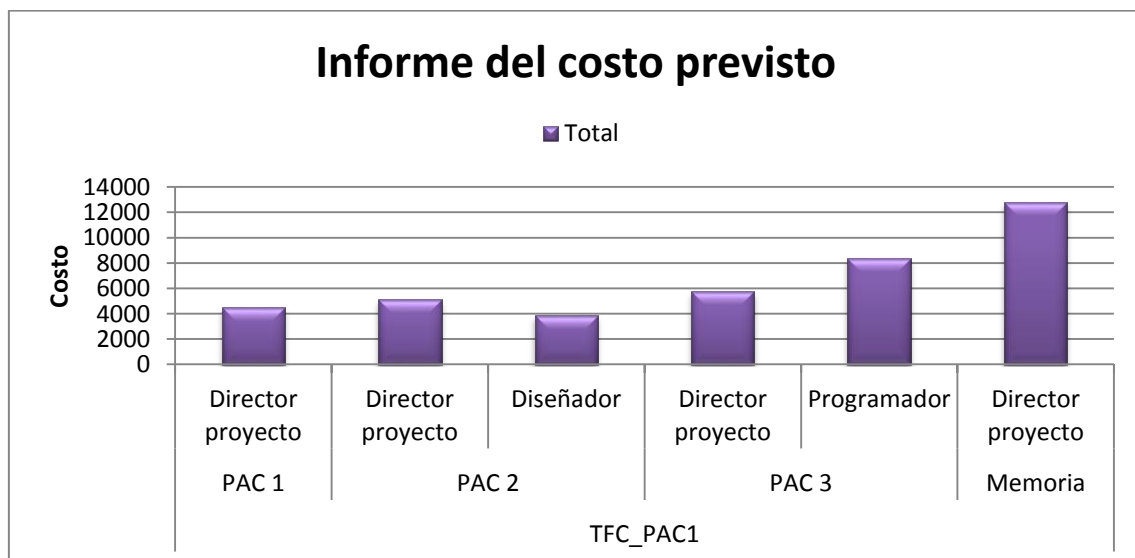
A nivel económico, al disponer del hardware necesario y hacer uso de software gratuito o de prueba sólo existe coste operacional. Al existir un único operario con diferentes roles y al tratarse de un proyecto académico el coste real del mismo es cero, por lo que resulta viable a nivel económico.

Una aproximación al coste operacional dentro del mundo empresarial, podría ser el que sigue a continuación:

Estimando un coste por hora de trabajo:

Director de proyecto: 80€ / h Diseñador: 40€/ h Programador: 50€/h

Gráficamente obtenemos:



El sumatorio de costes haría un total de 40400 €, lo que haría poco viable el proyecto, ya que no se espera un retorno económico directo dada su tipología.

Un cliente potencial podría ser algún tipo de administración pública, pero la coyuntura económica actual de las administraciones públicas no invita al optimismo.

Se puede concluir que el proyecto es viable en entornos académicos o colaborativos sin ánimo de lucro.

Análisis y diseño del sistema

El análisis y diseño del sistema se ha realizado acorde a la metodología DCU tal y como se ha mencionado anteriormente. Los siguientes puntos muestran detalladamente como se ha realizado esta fase.

Usuarios y contextos de uso

Durante la primera fase de investigación se ha usado la técnica de investigación contextual (*contextual Enquiry*) que permite observar la motivación del usuario mientras interactúa con un sistema similar al que se va a diseñar.

Investigación

Las aplicaciones seleccionadas para la investigación han sido Google Maps y FourSquare para Android.

El dispositivo sobre el que se han realizado las pruebas ha sido una Tablet Android con la versión del sistema operativo 4.0.3

A los usuarios del test se les ha solicitado realizar las siguientes tareas:

- Localizar su ubicación actual (Test 1)
- Consultar los datos de un bar cercano dado de alta en el sistema (Test 2)

A continuación se muestra el resumen de los resultados de los test con información relativa a los usuarios que lo han realizado:

Usuario	Luigi, 40 años
Perfil	Técnico, desarrollador de aplicaciones
Test 1	Éxito, sin problemas aparentes
Test 2	Éxito, sin problemas aparentes
Conclusiones	Conoce las aplicaciones y no tiene problemas en el uso

Usuario	Silvia, 31 años
Perfil	Técnico, diseñadora web
Test 1	Éxito, sin problemas aparentes
Test 2	Éxito, sin problemas aparentes
Conclusiones	Conoce las aplicaciones, pero es crítica con la velocidad de FourSquare

Usuario	Aina, 28 años
Perfil	Administrativa
Test 1	Éxito, sin problemas aparentes
Test 2	Desconoce FourSquare y tarda más que otros usuarios en conseguirlo
Conclusiones	El uso de Google Maps le resulta más familiar

Observaciones adicionales:

Se ha intentado realizar el test con personas sin conocimientos de uso de Smartphones en general sin poder completarlos con éxito, ya que se requiere estar familiarizado con el uso de estas tecnologías para poder usar las aplicaciones objeto de los test.

Perfiles de usuario

Una vez realizada la investigación podemos concluir que la segmentación de usuarios por nivel de experiencia con las aplicaciones móviles, no sería efectiva. Es necesaria una experiencia mínima con el uso de la tecnología para el uso de la aplicación a desarrollar.

Demográficamente, la idea inicial se basa en realizar denuncias en la ciudad de Palma de Mallorca (con posibilidad de ampliación en el futuro) por lo que en principio no segmentaremos usuarios de acuerdo a este criterio.

Una vez expuestos los argumentos anteriores, los usuarios van a ser segmentados de acuerdo al interés que presenten en el uso de la aplicación, asumiendo que la han descargado e instalado en sus dispositivos móviles.

Estos son los perfiles de usuario a considerar:

Usuarios pasivos:

Hacen uso de la aplicación esporádicamente y visitan los puntos que han creado otros usuarios, aunque no comentan ni añaden nueva información sobre los mismos. No crean nuevos puntos y como máximo podrían votar positiva o negativamente la información existente.

Usuarios activos:

Son usuarios habituales de la aplicación. Añaden todo tipo de información al sistema e incluso se animan a denunciar a la autoridad competente.

Contextos de uso

A nivel técnico es imprescindible disponer de un dispositivo Smartphone con sistema operativo Android, GPS y conexión a Internet móvil activa.

A nivel social no existe una limitación concreta más allá de los conocimientos mínimos necesarios para usar la tecnología del dispositivo y disponer del mismo.

Teniendo en cuenta los datos anteriores un contexto de uso de la aplicación sería aquel en el que un usuario en su vida cotidiana normal, paseando por la calle observase una anomalía de responsabilidad municipal que le llamase la atención y que deseara registrar en la aplicación.

Otro contexto de uso habitual sería aquel en que el usuario, en cualquier lugar y momento con su dispositivo móvil deseara ampliar información, consultar la misma o tramitar la denuncia de un punto existente en la aplicación.

Análisis de tareas

Las principales tareas que debe realizar el sistema son las siguientes:

- Crear un nuevo punto
- Consultar un punto existente
- Añadir información a un punto existente
- Tramitar una denuncia sobre un punto existente
- Obtener la ruta desde la posición actual hasta un punto existente

A continuación se indican los pasos necesarios para realizar las tareas

Tarea 1: Crear un nuevo punto

1. Abrir la aplicación, esta obtiene automáticamente la posición del usuario. En caso de haber navegado por el mapa pulsar botón obtener posición, ya que sólo se pueden crear puntos nuevos desde la posición actual del usuario.
2. Pulsar encima del indicador gráfico que aparece en la posición actual del usuario sobre el mapa.
3. Pulsar encima del diálogo contextual que aparecerá sobre el punto.
4. Si se desea añadir una fotografía se debe pulsar el icono y automáticamente se accederá a la aplicación cámara del dispositivo.
5. Rellenar los datos del formulario y pulsar el botón crear.

Tarea 2: Consultar un punto existente

1. Navegar por el mapa y pulsar encima del punto deseado.
2. Pulsar encima del diálogo contextual que aparecerá sobre el punto.
3. Consultar la información deseada en la pantalla de la aplicación.

Tarea 3: Añadir información a un punto existente

1. Realizar la tarea 2.
2. Si se desea añadir una fotografía se debe pulsar el icono y automáticamente se accederá a la aplicación cámara del dispositivo.
3. Rellenar los datos del que permite el formulario
4. Pulsar el botón "Añadir Información".

Tarea 4: Tramitar una denuncia sobre un punto existente

1. Realizar la tarea 2.
2. Pulsar el botón "Denunciar".
3. Rellenar los datos del formulario y pulsar el botón "Enviar".

Tarea 5: Obtener la ruta desde la posición actual hasta un punto existente

1. Realizar la tarea 2.
2. Pulsar el botón "Obtener ruta".

Diseño conceptual

Escenarios de uso

A continuación se detallan una serie de posibles escenarios de uso de la aplicación a desarrollar.

Escenario 1

Perfil de usuario	Usuario activo
Contexto	Vida cotidiana , fuera del hogar
Objetivo	Crear un nuevo punto denunciante
Tareas	Crear un nuevo punto (Tarea 1)
Necesidad de información	Localización del punto
Funcionalidades necesarias	Localización GPS, fotografía, adición de información
Desarrollo de las tareas	Localizar el punto y añadir la información deseada

El señor gato azul ruso vive frente a un solar municipal y se dirige como cada noche a tirar la basura. Últimamente ha detectado que el solar está lleno de ratas debido a que sus vecinos poco cívicos no depositan la basura en los contenedores sino en el dichoso solar. Hoy ha decidido sacar su móvil del bolsillo y usar la aplicación de denuncias urbanas para que la gente conozca su problema, para ello utiliza la ubicación GPS de su dispositivo y realiza una foto del lugar, dando de alta el punto en la aplicación.

Escenario 2

Perfil de usuario	Usuario pasivo
Contexto	Vida cotidiana, en el hogar
Objetivo	Consultar los puntos denunciados cerca del hogar
Tareas	Consultar un punto existente (Tarea 2)
Necesidad de información	Localización actual
Funcionalidades necesarias	Localizar la posición del usuario y mostrar puntos cercanos en el mapa
Desarrollo de las tareas	Abrir la aplicación y pulsar el botón de obtener posición

La señorita Cotilla, está en la cama y ha recibido un mensaje por WhatsApp, dónde le invitan a probar una nueva aplicación de denuncias urbanas. Dado que es una persona curiosa, enseguida prueba la aplicación y observa todos los puntos que tiene cerca de casa.

Escenario 3

Perfil de usuario	Nuevo usuario activo
Contexto	Vida cotidiana, fuera del hogar
Objetivo	Añadir nueva información sobre un marcador
Tareas	Añadir información a un punto existente (Tareas 2 y 3)
Necesidad de información	Localizar el punto
Funcionalidades necesarias	Localizar puntos mediante la navegación por el mapa, selección del punto, añadir información
Desarrollo de las tareas	Localizar el punto navegando por el mapa, seleccionarlo, rellenar el formulario y pulsar el botón de añadir información

La señora Cotilla al día siguiente de haber probado la aplicación, al bajar a la calle tropieza con un bache que se encuentra en la acera de una calle paralela a la que vive. Después de recuperarse del susto decide abrir la aplicación móvil y consulta que el punto dónde ha tropezado ya existe, por lo que decide añadir una nueva foto del lugar y comentar lo sucedido.

Escenario 4

Perfil de usuario	Usuario activo
Contexto	Vida cotidiana, en el hogar
Objetivo	Hacer efectiva una denuncia
Tareas	Tramitar una denuncia sobre un punto existente (Tareas 2 y 4)
Necesidad de información	Localizar el punto
Funcionalidades necesarias	Localizar puntos mediante la navegación por el mapa, selección del punto, formulario de denuncias
Desarrollo de las tareas	Localizar el punto navegando por el mapa, seleccionarlo, pulsar el botón de denuncia y rellenar el formulario

El señor gato azul ruso ante la desidia de las autoridades ha consultado su queja en la aplicación. Animado por los nuevos comentarios de los usuarios en la aplicación y los votos positivos que ha recibido el punto que dio de alta se ha decidido a denunciar la situación, para lo que rellena el formulario de denuncia y lo envía.

Escenario 5

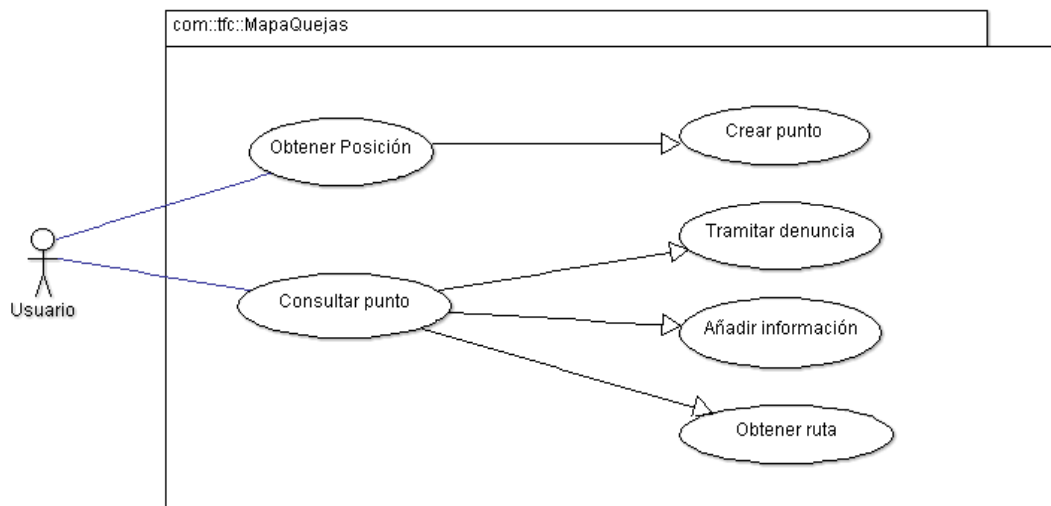
Perfil de usuario	Usuario activo
-------------------	----------------

Contexto	Vida cotidiana, fuera del hogar
Objetivo	Hacer efectiva una denuncia
Tareas	Obtener la ruta hacia un punto existente (Tareas 2 y 5)
Necesidad de información	Localizar el punto de destino
Funcionalidades necesarias	Localizar puntos mediante la navegación por el mapa, selección del punto, obtener ruta hacia el punto
Desarrollo de las tareas	Localizar el punto navegando por el mapa, seleccionarlo, pulsar el botón de obtener ruta

El señor gato azul ruso, ha decidido iniciar una cruzada ante la dejadez de su ayuntamiento, por lo que está dispuesto a consultar todos los puntos cercanos a la ruta diaria que sigue de camino a su casa desde el trabajo, para ello usa la aplicación para obtener la ruta desde su posición actual hacia los puntos que cree que puede llegar a pie con el fin de comprobar si cree pertinente realizar nuevas denuncias.

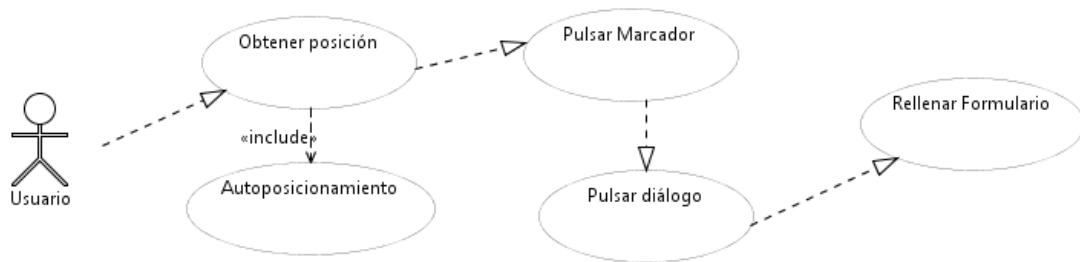
Diagramas de casos de uso

El siguiente diagrama muestra los casos de uso de las tareas que componen la aplicación.

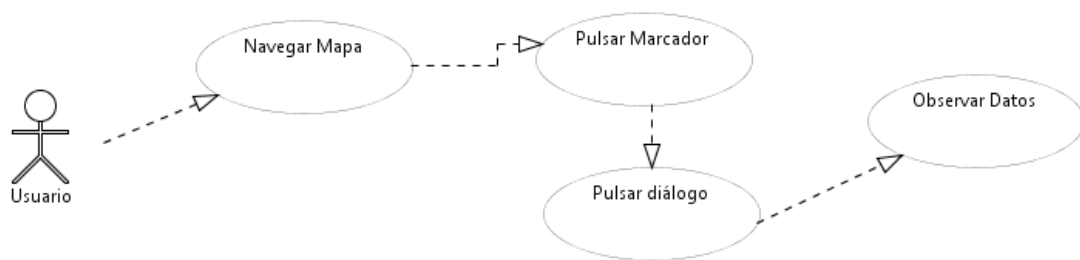


A continuación se presentan los casos de usos para cada una de las tareas de forma individual

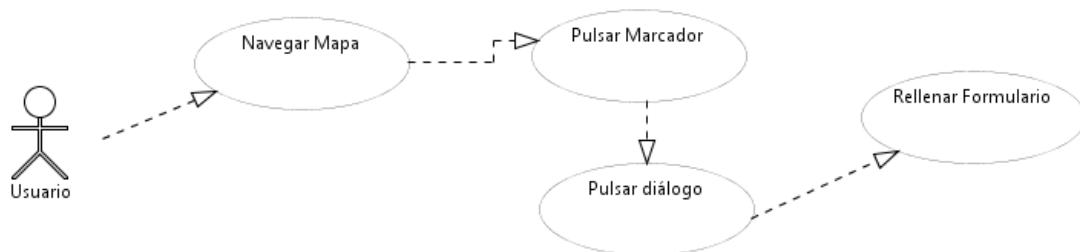
Tarea 1: Crear un nuevo punto



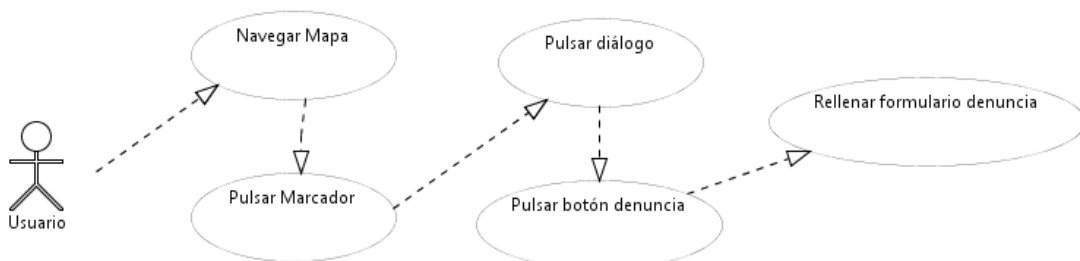
Tarea 2: Consultar un punto existente



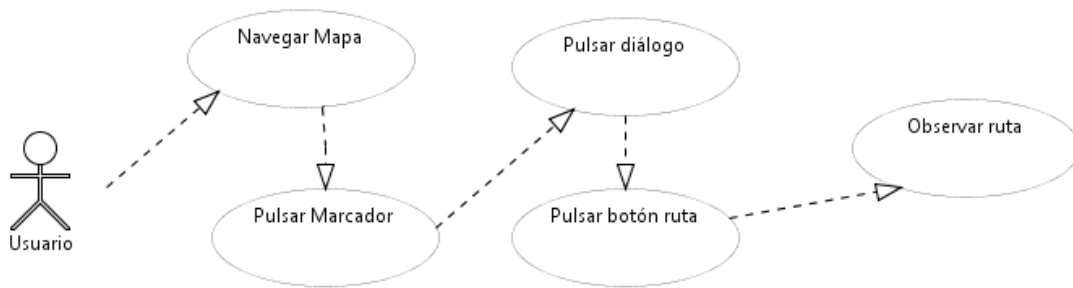
Tarea 3: Añadir información a un punto existente



Tarea 4: Tramitar una denuncia sobre un punto existente



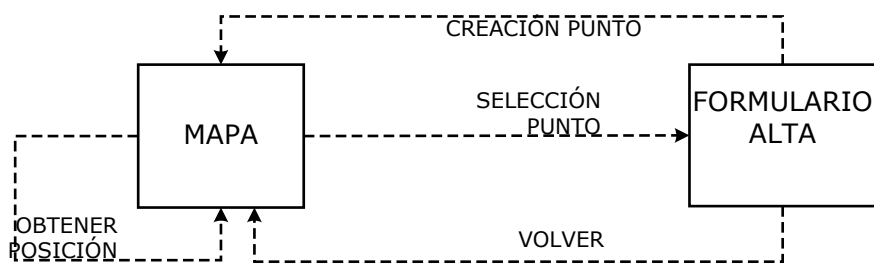
Tarea 5: Obtener la ruta desde la posición actual hasta un punto existente



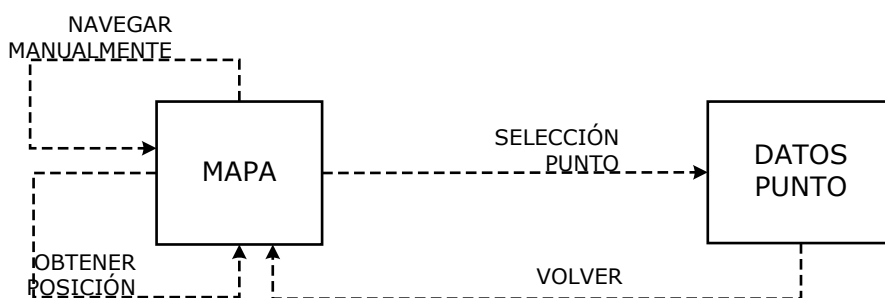
Flujos de interacción

A continuación se presentan los flujos de interacción de las tareas principales

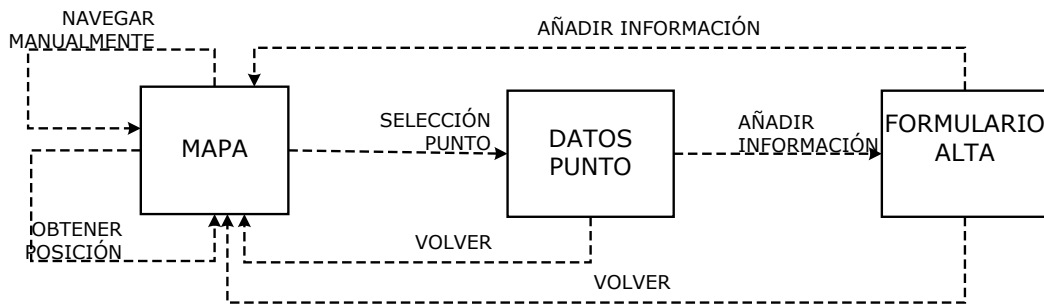
Tarea 1: Crear un nuevo punto



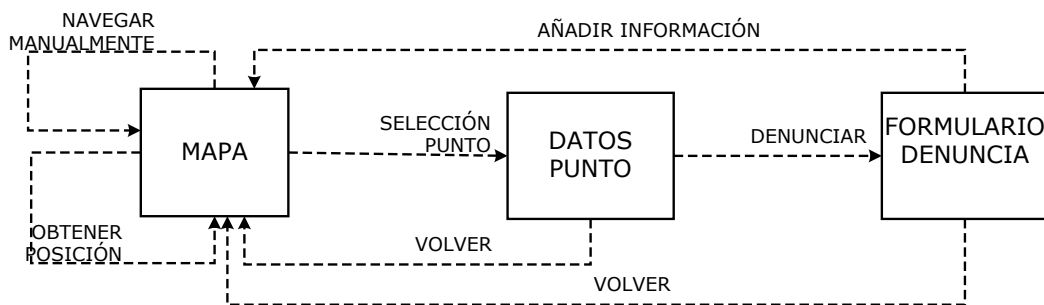
Tarea 2: Consultar un punto existente



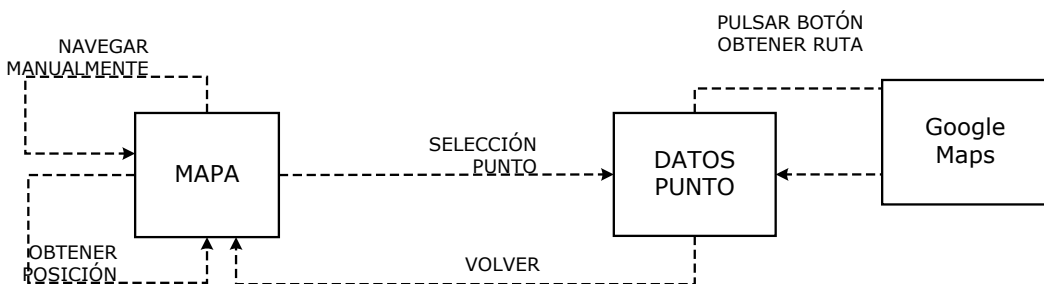
Tarea 3: Añadir información a un punto existente



Tarea 4: Tramitar una denuncia sobre un punto existente



Tarea 5: Obtener la ruta desde la posición actual hasta un punto existente



Prototipado

Sketches

No se ha realizado ningún sketch a mano, pese a las recomendaciones del DCU. Inicialmente se optó por usar la aplicación Axure Rp, con la ya que se había trabajado, para la realización del prototipo en ella donde se plasmó la primera idea de diseño de la aplicación.

Este es el sketch inicial realizado:

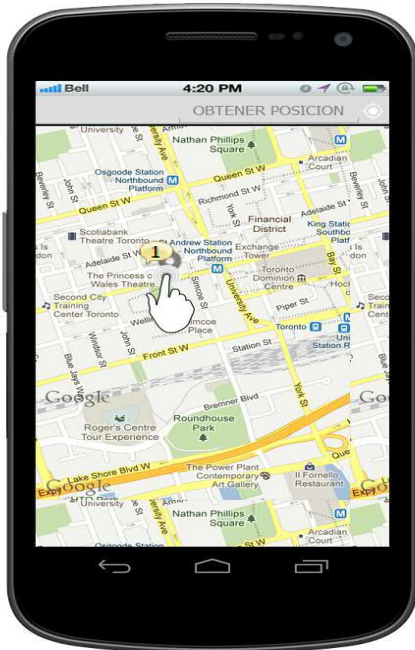


El sketch inicial muestra la ruta entre dos marcadores, que se correspondería a la tarea 5, y el botón de obtener posición.

No obstante en vista de la falta de elementos visuales en las librerías encontradas para la aplicación se optó por cambiar de herramienta y usar JustInMind Prototyper Free, con la que finalmente se ha realizado el prototipo.

Prototipo

A continuación se muestran las pantallas del prototipo final:



TAREA 1

En esta pantalla se representa la acción de obtener la posición mediante el botón “obtener posición” o la autocalización que realiza la aplicación al iniciarse.

Se observa el marker del punto en la posición obtenida



TAREA 1

Esta pantalla es resultado de la interacción anterior.

Se muestra un diálogo que indica que es un nuevo punto y que debemos pulsarlo

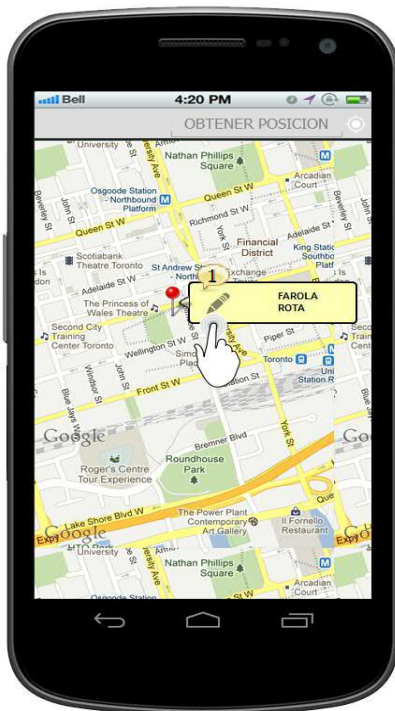


TAREA 1

Esta pantalla es resultado de la interacción anterior.

El icono foto nos permite realizar fotografías.

Si rellenamos los datos y pulsamos en el punto 1 se crea el punto. En caso de pulsar el punto 2 se vuelve al Mapa.



TAREA 2

Una vez creado el punto, este aparece en el mapa. El globo contextual muestra el título y permite pulsar para acceder a la información del punto.

**TAREA 2**

En esta pantalla observamos la información actual del punto. La opción 2 vuelve al mapa. El punto se puede votar.

TAREA 3

La opción 1 permite añadir información nueva del punto

TAREA 4

La opción 3 tramita una denuncia.

TAREA 5

Existe un botón para obtener la ruta desde la posición actual en un mapa.

**TAREA 4**

Esta pantalla muestra el formulario de denuncia.

Se deben rellenar los datos y pulsar el botón enviar para tramitar la denuncia.

El punto 1 vuelve al mapa.

El prototipo está disponible online para probarlo en la siguiente url:

<https://www.justinmind.com/usernote/tests/10328394/10328402/10328421/index.html>

Evaluación

Para la evaluación de la aplicación se ha tomado la decisión de usar dos herramientas web: Usernote, para la que se ha dispuesto de una versión trial de 1 mes, y Usertesting.com, mediante la integración e invitación gratuita que nos proporciona UserNote para usar una vez con tres usuarios.

Usuarios que realizarán el test

El perfil de usuarios que van a realizar el test en Usernote, bajo invitación, es de profesionales. Para ello se ha invitado a Luigi y Silvia amigos e integrantes de <http://www.ma-no.org/> y a los consultores del TFC Jordi e Ignasi.

El rango de edad de estos usuarios se sitúa en la franja de los 30 a 40 años.

En la herramienta Usertesting.com no se ha discriminado a ningún tipo de usuario, aunque existe el problema de que no existe una comunidad hispana actualmente en la herramienta (los usuarios son de Gran Bretaña, Canadá y Estados Unidos) y el prototipo está en español, lo que puede resultar confuso para estos usuarios.

Tareas que realizar en el test

A los usuarios del test en Usernote se les requeriría intentar completar las tareas detectadas en el análisis de tareas:

- Crear un nuevo punto
- Consultar un punto existente
- Añadir información a un punto existente
- Tramitar una denuncia sobre un punto existente

En el test de Usertesting.com se ha utilizado una plantilla predefinida para prototipos con las siguientes tareas:

- Tarea 1: Mira alrededor y di lo que piensas acerca de esta página ¿qué se puede hacer aquí? ¿Dónde se hace clic por primera vez?
- Tarea 2: ¿Qué te gustó acerca de esta esta página? ¿Qué no te gusta de esta página?

- Tarea 3: Ahora, ves a la página siguiente. ¿Qué te gusta de esta página?
¿Qué no te gusta de esta página?

Preguntas sobre las tareas

A los usuarios de Usernote se les harían las siguientes preguntas:

- ¿Qué dificultades ha encontrado para completar las tareas?
- ¿De qué elementos carece el prototipo actual?
- ¿Qué defectos ha detectado en el uso del prototipo?
- ¿Cómo se podría mejorar la herramienta?

En el test de Usertesting .com se ha utilizado la batería de preguntas predefinidas:

- ¿Esta web cumple con sus expectativas?
- ¿Qué es lo mejor de este sitio? ¿Qué es lo peor de este sitio?
- Si tuviera una varita mágica, ¿cómo mejoraría este sitio?
- ¿Utilizaría este sitio en el futuro (por favor explique por qué o por qué no)?

Conclusiones resultados test

Los test realizados en Usertesting se pueden consultar en el apéndice que incluye esta memoria.

Resulta evidente que el prototipo no se ha explicado de la manera adecuada y que la barrera idiomática ha resultado un hándicap para ello. Disponer de usuarios hispanohablantes o haber elegido unas preguntas más adecuadas podría haber mejorado el resultado de los mismos.

Respecto los resultados obtenidos en Usernote, la falta de crítica ha inducido a pensar que el prototipo era correcto. Innovar en el diseño de una aplicación basada en mapas, de uso habitual en la comunidad de usuarios de dispositivos

móviles, podría crear problemas de usabilidad y resultaría contraproducente para el resultado final de la aplicación.

El cambio más evidente que ha afectado a la aplicación es el de obtener automáticamente la posición del usuario, funcionalidad añadida en la segunda iteración de diseño durante el proceso del DCU. Esta sugerencia fue recibida por los usuarios del test.

La idea inicial obligaba a pulsar el botón de obtener posición. Al final este botón pasa a tener una funcionalidad que sirve para reubicar al usuario durante la navegación por los mapas o bien refrescar la misma en caso de detectar que la posición actual no se corresponde con la que presenta la aplicación en ese momento.

El formulario de denuncia envía un correo mediante la aplicación de la que disponga el móvil, por lo que no se almacenan datos en la aplicación de carácter personal que obliguen a cumplir la LOPD, evitando así problemas legales. Esta advertencia fue realizada por parte los usuarios del test.

Implementación

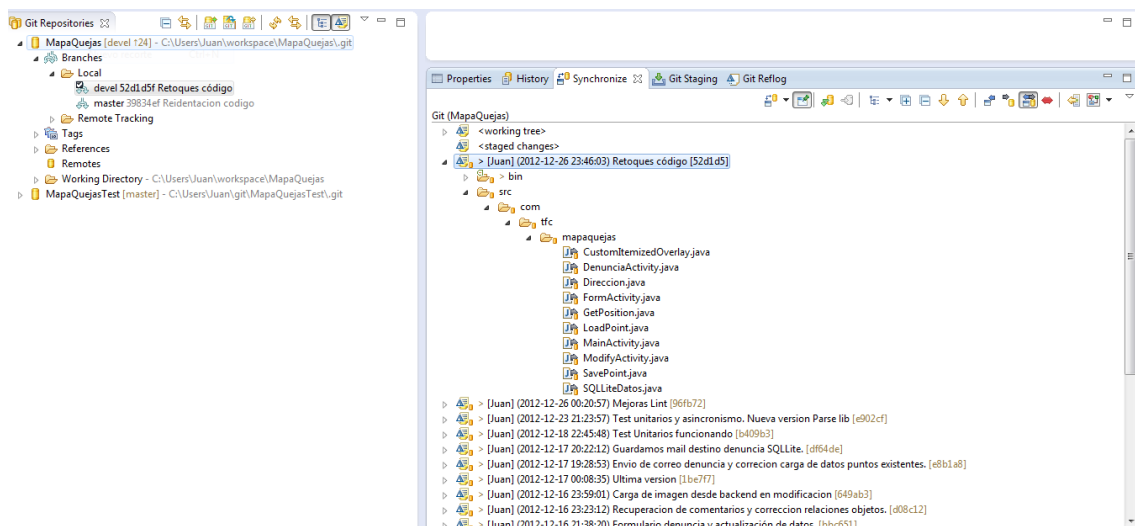
Durante esta fase se ha realizado la codificación de la aplicación intentando seguir el método de desarrollo ágil, creando múltiples iteraciones del producto hasta llegar al resultado final. Las diferentes iteraciones se han almacenado en un gestor de versiones GIT.

El código se ha desarrollado con la ayuda del IDE Eclipse y ADT (Android development tools)

Gestión de código

Durante el desarrollo de la aplicación el código de la aplicación se ha gestionado mediante GIT, sistema de control de versiones.

A nivel local se ha usado el plugin para el IDE Eclipse EGIT.



El histórico se ha sincronizado en las diferentes iteraciones contra dos repositorios en bitbucket.

https://bitbucket.org/jugarridoco/tfc_jugarridoco






En él se aloja el código relacionado de la aplicación MapaQuejas.

https://bitbucket.org/jugarridoco/tfc_jugarridoco_test

Proyecto para crear los test unitarios para la aplicación.

Finalmente los test se han integrado en el repositorio principal.

La rama sobre la que se ha trabajado principalmente se denomina devel, en la imagen podemos observar los últimos cambios.

devel 9 ahead / 1 behind				Compare
	Juan Garrido	52d1d5f	Retoques código	11 days ago
	Juan Garrido	96Eb726	Mejoras Lint	11 days ago
	Juan Garrido	e902cf0	Test unitarios y asincronismo. Nueva version Parse lib	14 days ago
	Juan Garrido	b409b32	Test Unitarios funcionando	19 days ago
	Juan Garrido	dff64ded	Guardamos mail destino denuncia SQLite.	20 days ago

Arquitectura de la aplicación

La documentación de arquitectura se ha pospuesto hasta la fase de implementación, en la que finalmente se puede mostrar de forma completa.

La aplicación consta de dos capas diferenciadas el cliente que se ejecuta en un dispositivo Android y la parte servidora, dónde se alojan los datos de la aplicación.

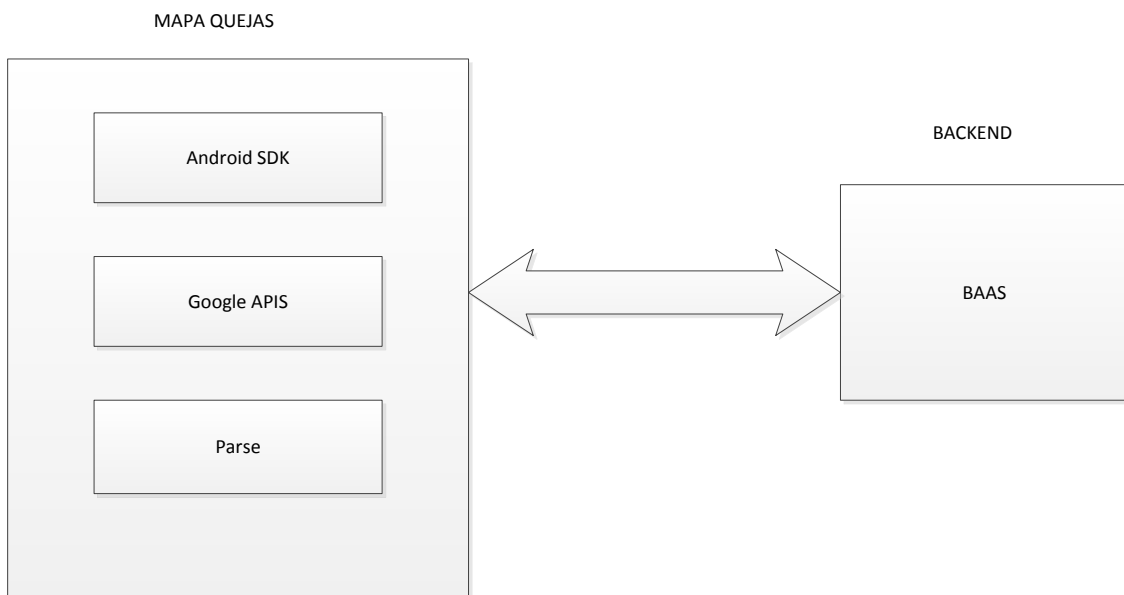


Para realizar la aplicación MapaQuejas se ha decidido prescindir del uso de un backend en lenguaje de servidor y base de datos para usar un tipo de servicio en la nube denominado Backend as a Service.

En este caso se ha optado por usar Parse.com, aunque también se ha valorado la opción de StackMob, optando finalmente por Parse, por la facilidad de uso de su API.

Parse es un servicio en la nube que nos permite guardar objetos en nuestra aplicación y posteriormente recuperarlos.

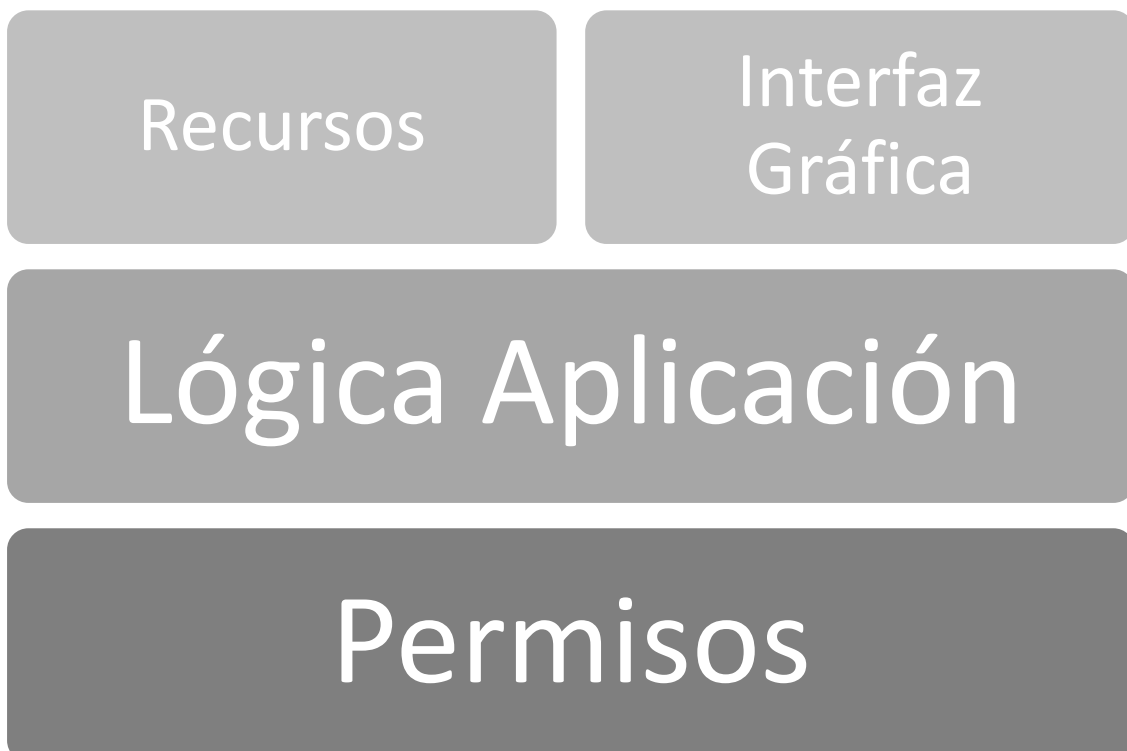
La estructura de la aplicación Mapa Quejas de forma resumida es la siguiente:



Se hace uso de Android SDK y las librerías Google Apis para generar los mapas. La librería parse permite la comunicación con el Backend.

Arquitectura cliente MapaQuejas

La aplicación MapaQuejas se puede dividir en diferentes capas:



La **capa de presentación** formada por los recursos textuales y la interfaz gráfica.

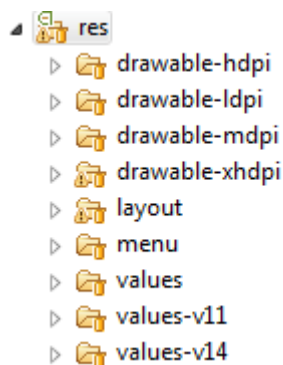
Los **permisos** que permiten a la aplicación acceder a diferentes características del dispositivo.

La **lógica de aplicación** formada por las diferentes clases que dan forma a la aplicación.

Esta separación de capas es la habitual en las aplicaciones para Android.

Capa de presentación

La capa de presentación se encuentra en el directorio **MapaQuejas\res** de la aplicación



Dentro de las carpetas que se nombran como **drawable** encontramos las imágenes de las que hace uso la aplicación para las diferentes resoluciones de dispositivo.

En la aplicación MapaQuejas se ubicarán los pines y el icono de la aplicación.

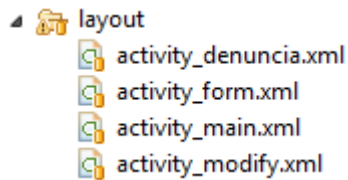


Las carpetas nombradas como **values** contienen documentos XML que incluyen los textos para poder internacionalizar la aplicación.

```
strings.xml  strings_activity_form.xml  styles.xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="app_name">MapaQuejas</string>
  <string name="hello_world">Hello world!</string>
  <string name="menu_settings">Settings</string>
  <string name="title_activity_main">MapaQuejas</string>
  <string name="edit_message">Queja</string>
  <string name="title_activity_modify">ModifyActivity</string>
  <string name="title_activity_denuncia">DenunciaActivity</string>
</resources>
```

La carpeta **menu** incluye documentos XML para realizar la presentación de los menús de la aplicación, esta funcionalidad no se ha implementado.

La carpeta layout contiene documentos XML donde se ha realizado el diseño de las diferentes actividades (pantallas) principales que contiene la aplicación.



Como ejemplo el diseño del activity main sería el siguiente:

```
activity_main.xml
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity" >

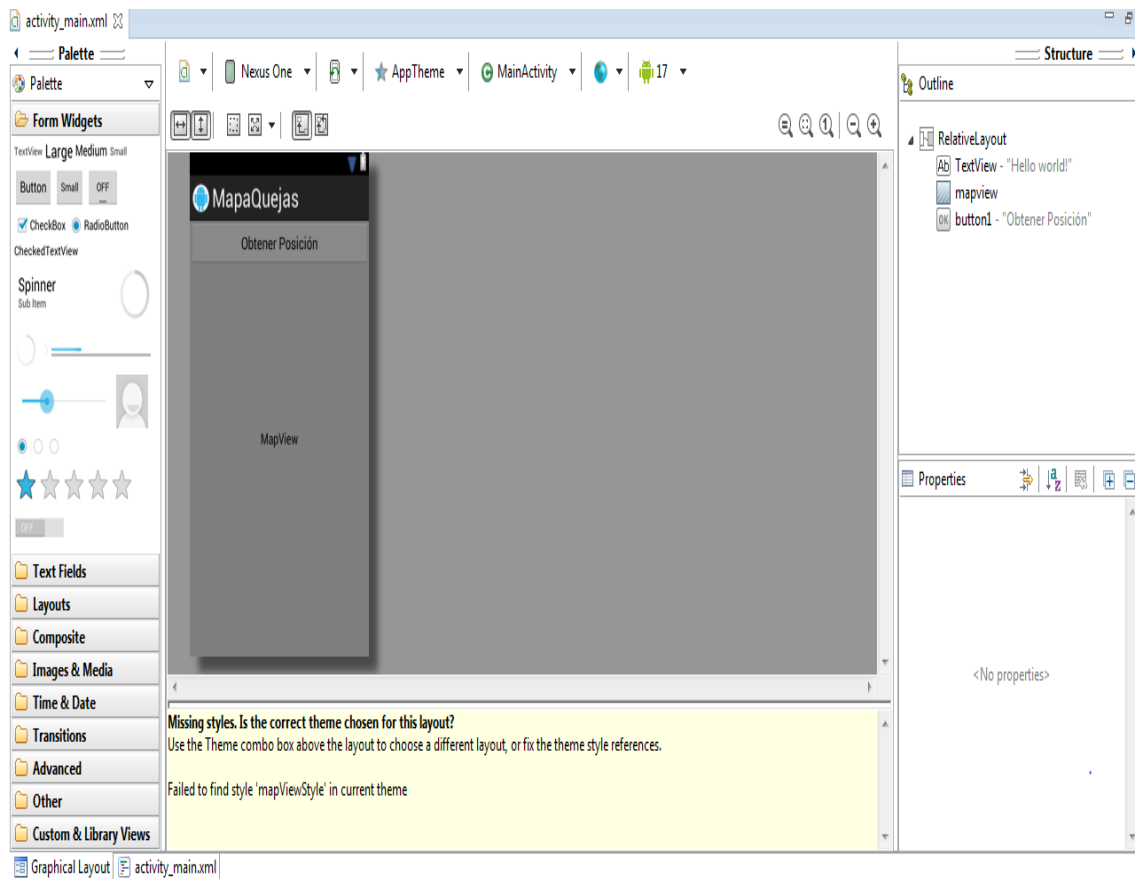
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

    <com.google.android.maps.MapView
        xmlns:android="http://schemas.android.com/apk/res/android"
        android:id="@+id/mapview"
        android:layout_width="fill_parent"
        android:layout_height="fill_parent"
        android:apiKey="@string/api_key"
        android:clickable="true" />

    <Button
        android:id="@+id/button1"
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_alignParentLeft="true"
        android:layout_alignParentRight="true"
        android:contentDescription="@string/button_string"
        android:onClick="updatePosition"
        android:overScrollMode="always"
        android:text="@string/button_string" />

</RelativeLayout>
```

La vista en el diseñador es la siguiente.



Los principales elementos gráficos que se ha usado en la aplicación mapaQuejas son los siguientes:

Layouts (bloques de diseño), MapView (vista de mapa), Buttons (botones), EditText (entradas de texto) y TextView (vistas de texto).

Permisos

Los permisos de la aplicación se definen en el fichero **MapaQuejas\AndroidManifest.xml**

```

MapaQuejas Manifest
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.tfc.mapaquejas"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="17" />

    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_MOCK_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

    <application
        android:allowBackup="false"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <uses-library android:name="com.google.android.maps" />

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

</manifest>

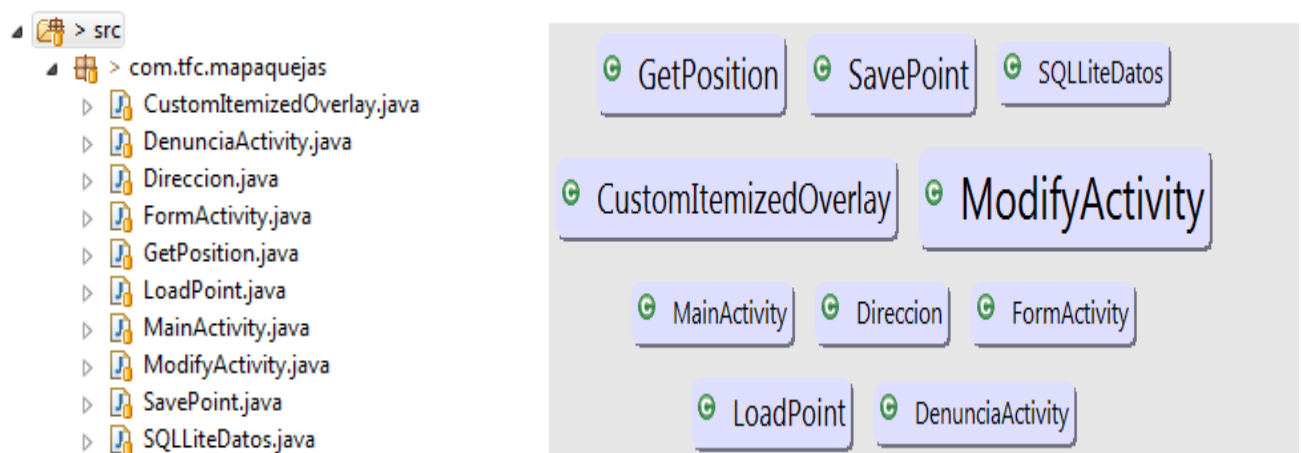
```

Este fichero en formato XML también incluye datos relativos a las actividades (pantallas) que forman la aplicación y las versiones de SDK para los que se desarrolla la misma.

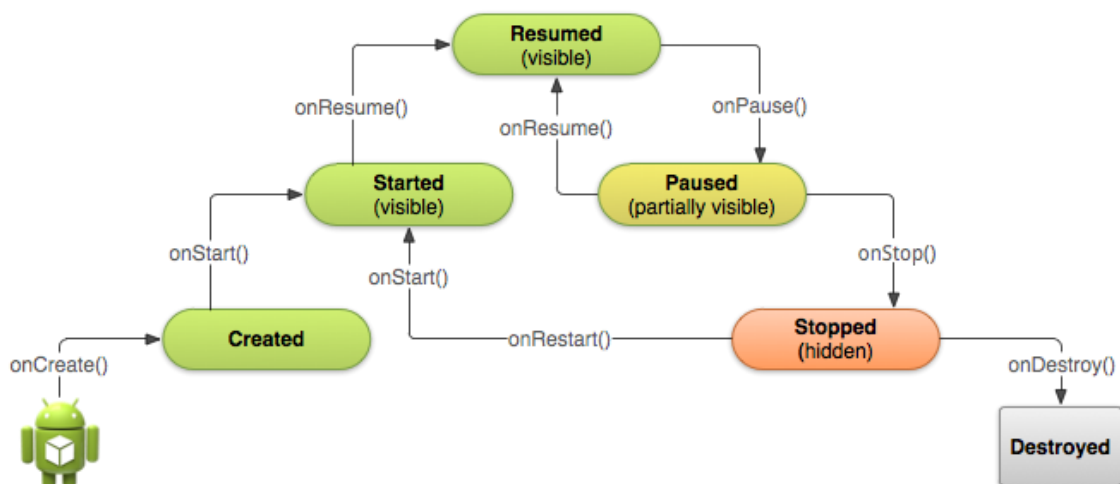
Los permisos usados en la aplicación son los referentes al acceso a Internet, localización y estado de red del dispositivo.

Lógica aplicación

La lógica de la aplicación contiene todas las clases en lenguaje Java de las que hace uso la aplicación. Se ubica en la carpeta MapaQuejas\src.



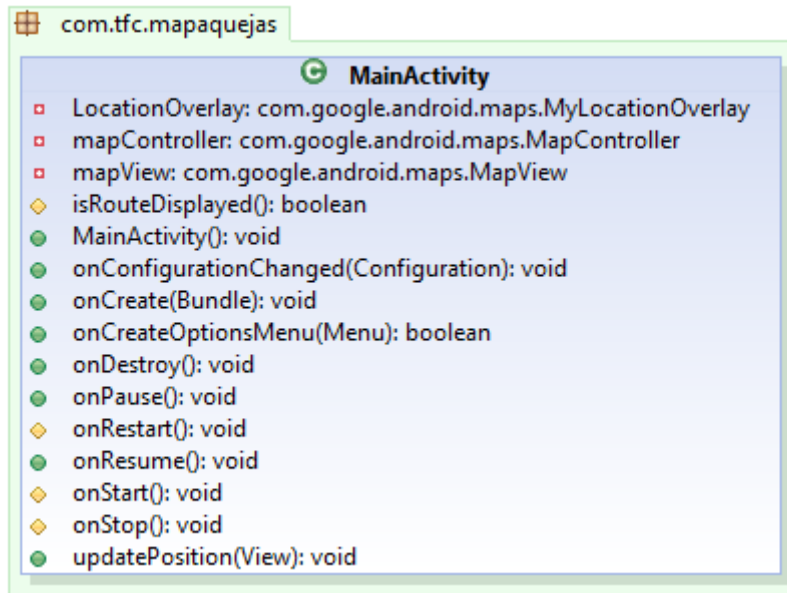
Las clases de tipo Activity siguen el ciclo de vida de una aplicación Android:



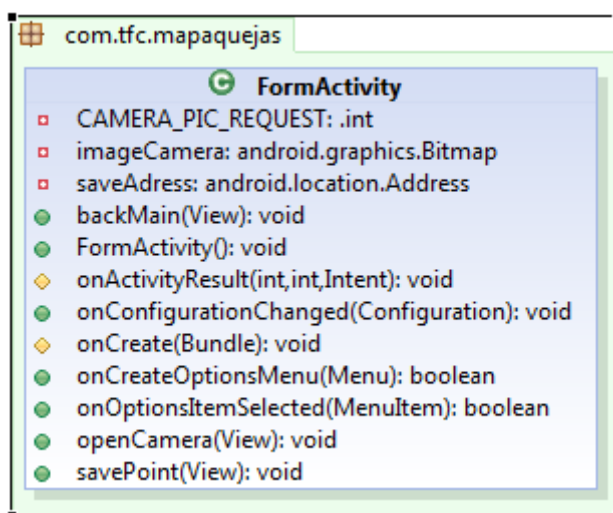
Las clases de tipo **Activity** son las diferentes pantallas que contiene la aplicación. Utilizamos la clase **Intent** para comunicarnos entre ellas y con otras aplicaciones (Cámara, Navegador, etc.).

Diagramas de clases

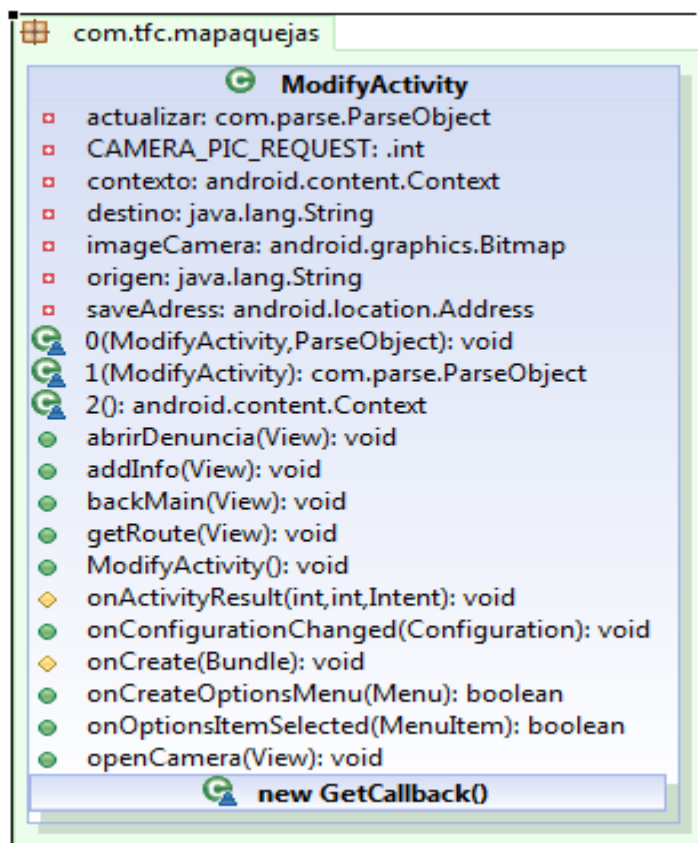
MainActivity: clase principal de la aplicación. Contiene la vista de mapas.



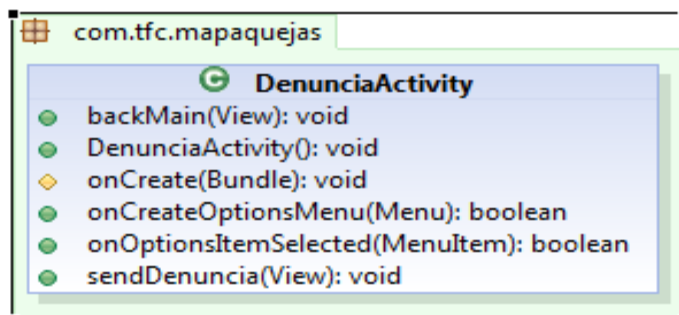
FormActivity: Clase tipo Activity que contiene el formulario de alta de puntos.



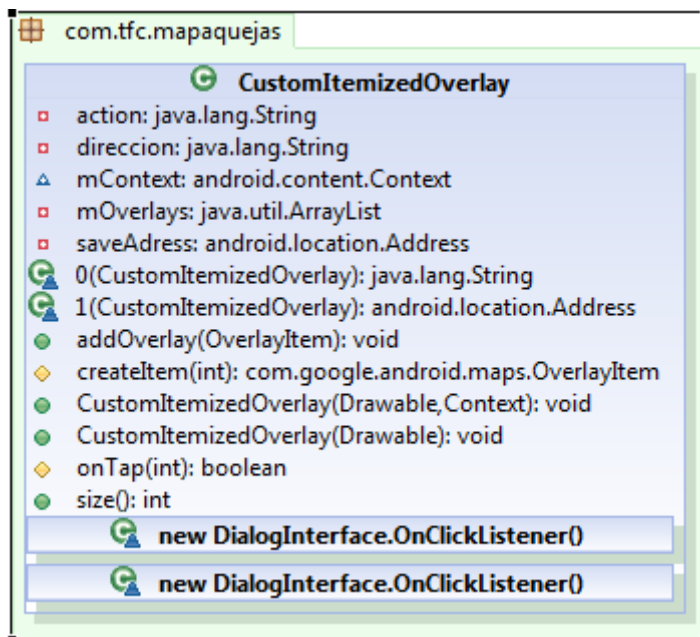
ModifyActivity: Clase tipo Activity que contiene el formulario de modificación y consulta de puntos. Lleva integrada en sí la lógica de comunicación con el Backend.



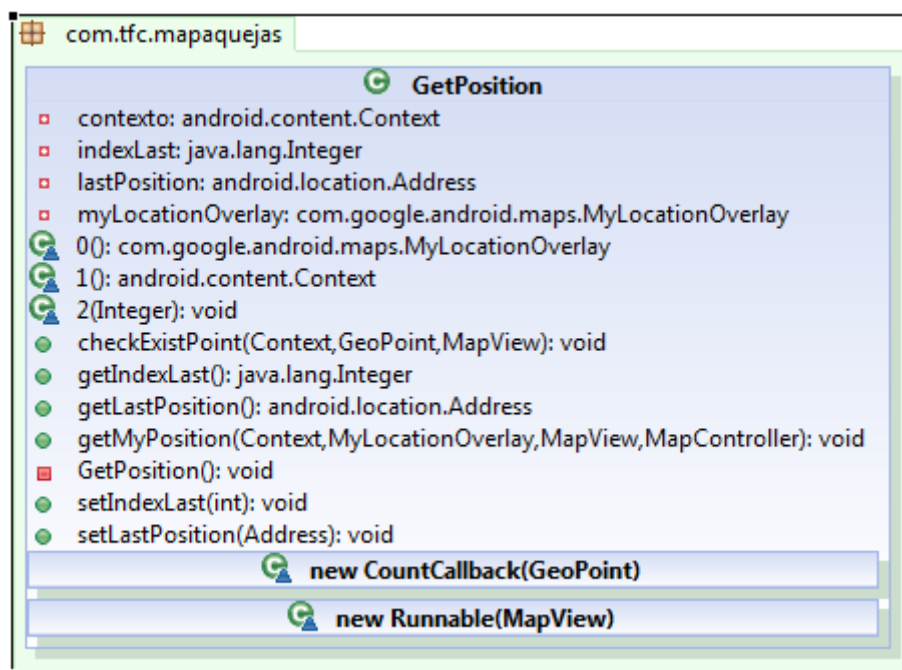
DenunciaActivity: Clase de tipo Activity que contiene el formulario de denuncia.



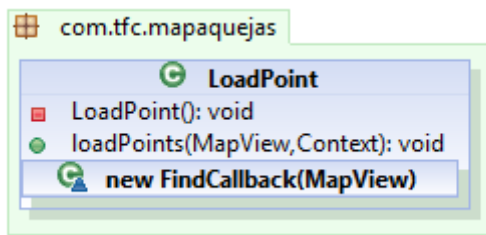
CustomItemizedOverlay: Clase particular para personalizar los pines (overlays) y las acciones (Dialogs) que se realizan al pulsar sobre los mismos.



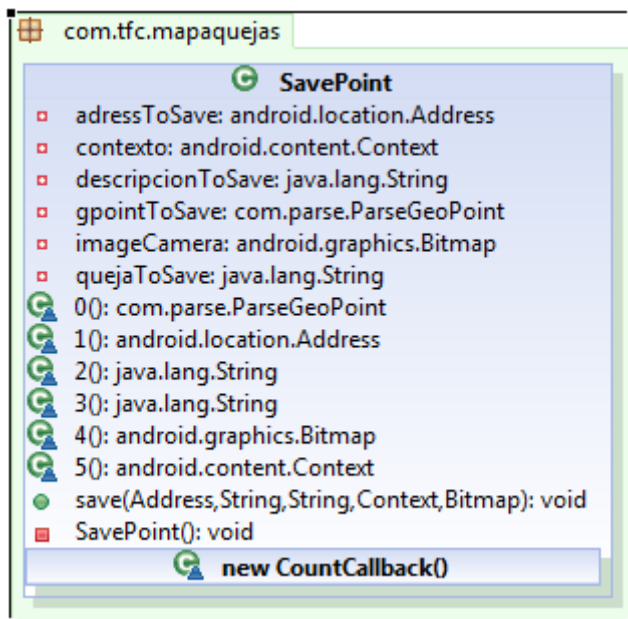
GetPosition: Clase utilizada para obtener la posición actual del dispositivo y mostrar el pin (tipo nuevo o existente) al abrir la aplicación o pulsar el botón de obtener posición.



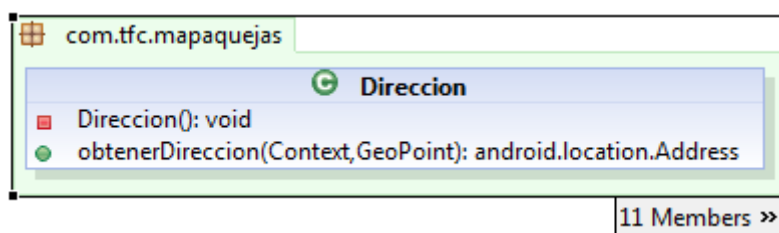
LoadPoint: Clase que se encarga de cargar todos los puntos existentes, obteniendo los datos desde el backend.



SavePoint: Clase que se encarga de guardar los nuevos puntos en el backend.



Direccion: Clase que se encarga de transformar puntos geográficos en direcciones.



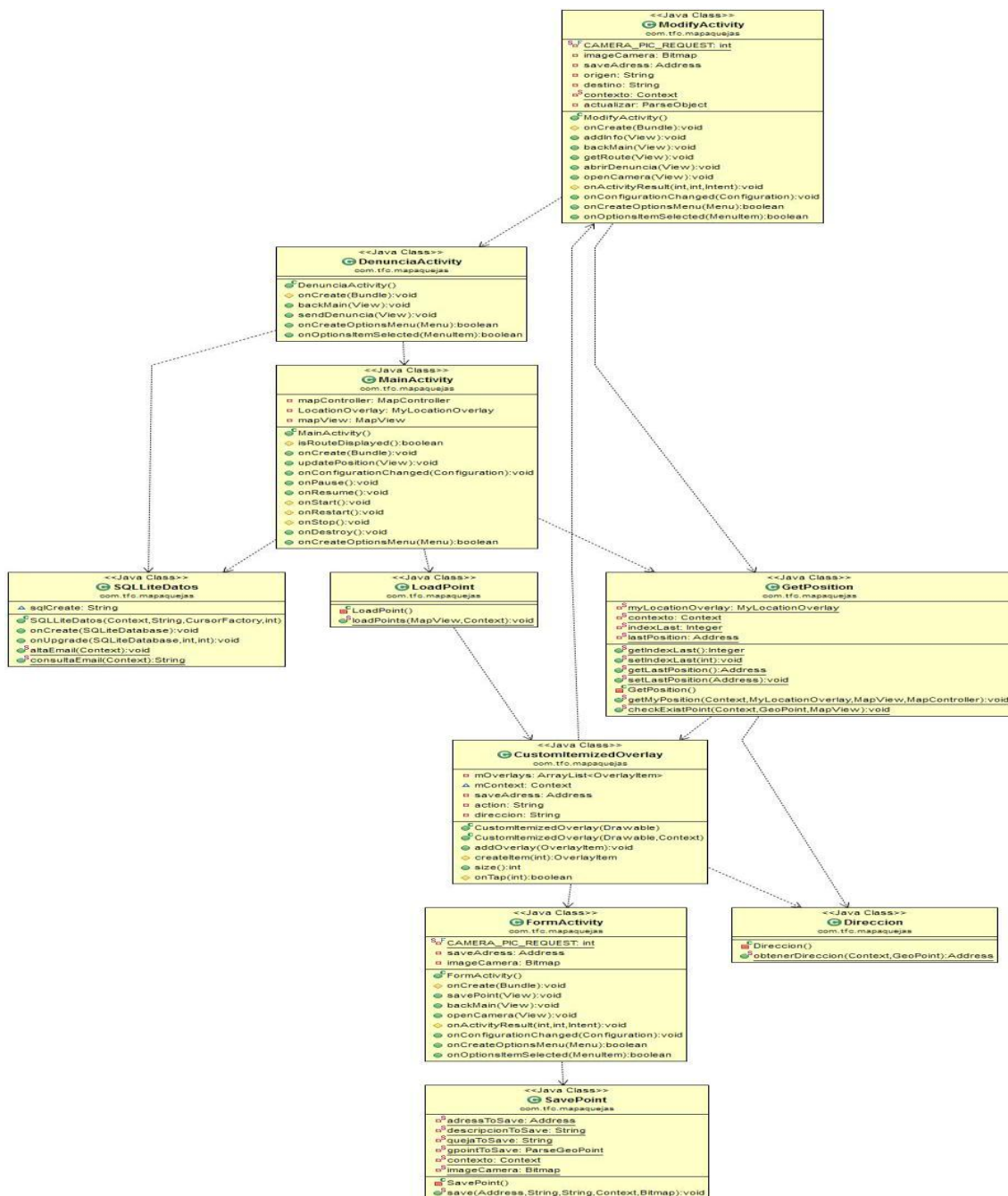
SQLiteDatos: Clase encargada de realizar todas las operaciones sobre la base de datos interna SQLite.

```

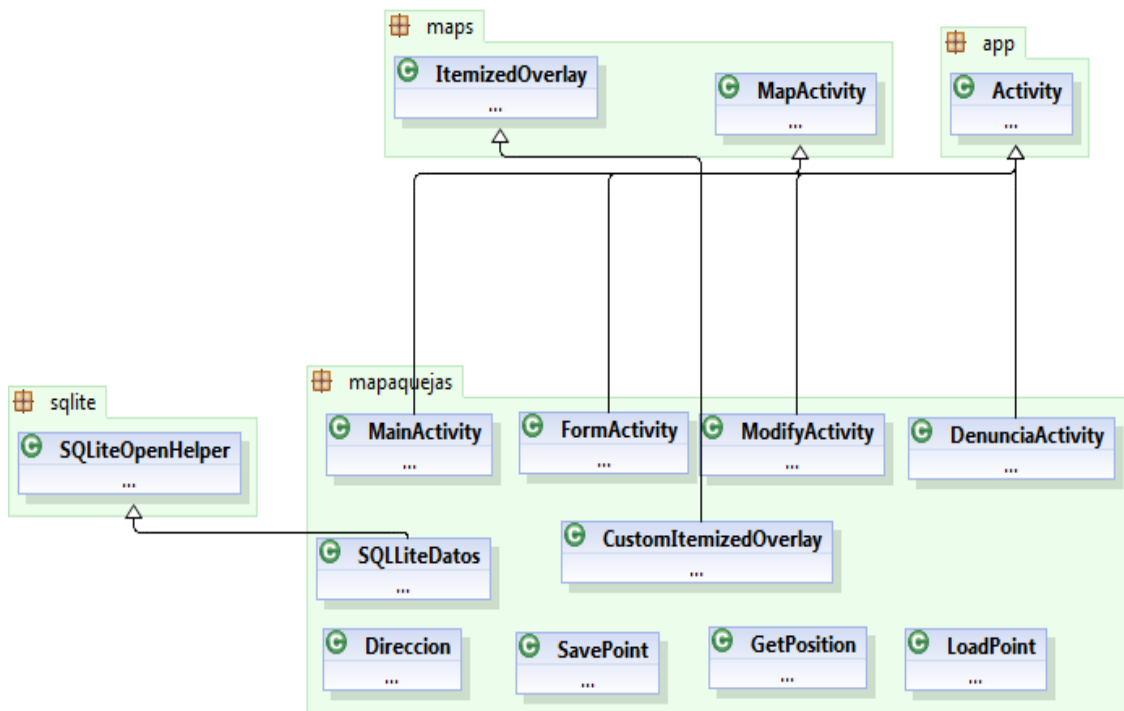
com.tfc.mapaquejas

    SQLiteDatos
    sqlCreate: java.lang.String
    altaEmail(Context): void
    consultaEmail(Context): java.lang.String
    onCreate(SQLiteDatabase): void
    onUpgrade(SQLiteDatabase,int,int): void
    SQLiteDatos(Context,String,CursorFactory,int): void
    
```

Las relaciones entre clases se pueden observar en el siguiente gráfico



El siguiente gráfico muestra las clases padre de las que heredan algunas de las clases



Activity: clase de la que heredan todas las clases que forman las pantallas de la aplicación. Son las clases con las que el usuario puede interactuar. Forma parte de la API Android.

ItemizedOverlay: clase de la que hereda `CustomItemizedOverlay`. La clase `Itemized overlay` permite personalizar los markers (puntos) presentes en la aplicación. Permite asignar los iconos deseados y la funcionalidad que se ejecutará cuando los markers sean pulsados. Forma parte de la API de Maps.

MapActivity: clase utilizada para instanciar en la actividad principal (`MainActivity`) la vista de mapas. Es la clase sobre la que se basan los mapas.

Pertenece a la API de Maps.

Diagrama de secuencia actividad principal (MainActivity)

La siguiente figura muestra el diagrama de secuencia de la actividad principal de la aplicación.



El código de esta clase se adjunta en el apéndice de la memoria.

Persistencia de datos

Para el almacenamiento de datos se ha utilizado el Backend Parse. La siguiente figura muestra el navegador de objetos de la aplicación.

Classes	+ Row	- Row	+ Col	More											
User	0														
Comentarios	14														
Poi	18														
Poimage	12														
+ New Class															
+ Import															
objectId	String	CP	String	Coordenadas	GeoP...	Direccion	String	Localidad	String	descripcion	String	titulo	String	createdAt	Date
aP97p8vGA	07003			39.5773031, 2.6450...		Carrer Llorenç Cerdà...		Palma de Mallorca		y		t		Dec 26, 2012, 22:33	
YIKjIYrG7	07003			39.5780366, 2.6479...		Carrer Sant Joan Sal...		Palma de Mallorca		e		e		Dec 26, 2012, 22:18	
HmduPBSSVv	07003			39.5771021, 2.6477...		Avinguda Alemanya, 11		Palma de Mallorca		emu		emu		Dec 26, 2012, 22:15	
sL9dL9w0W	07003			39.5780002, 2.6468...		Carrer de Jesús, 5		Palma de Mallorca		test		tttttt		Dec 26, 2012, 21:01	
fPgC8fpPwP	07003			39.5780163, 2.6498...		Avinguda Comte de ...		Palma de Mallorca		sin foto		test		Dec 23, 2012, 20:04	
i0bWN8M1hL	07001			39.5700002, 2.6499...		Carrer Paners, 9A		Palma de Mallorca		www		tttt		Dec 23, 2012, 20:01	
x2pTsr7UJ4	07008			39.5814222, 2.6670...		Carrer Aragó, 136A		Palma de Mallorca		rest		nuevo		Dec 17, 2012, 21:09	
yoi0jmrLkh	07005			39.5807012, 2.6612...		Plaça Santa Elisabet 2		Palma de Mallorca		a ver si te afeitas		Capullin		Dec 17, 2012, 20:06	
xHXsA3urao	07004			39.5837023, 2.664606		Carrer Miquel Fleta, 1		Palma de Mallorca		siempre duerme		simba duerme		Dec 16, 2012, 23:03	
dVShuGj1Aw	07005			39.5833563, 2.6652...		Carrer Jacint Verdag...		Palma de Mallorca		Tezuka		test		Dec 16, 2012, 21:38	
pxvK3O0ZbY	07004			39.5839719, 2.6647...		Carrer Andrés Segov...		Palma de Mallorca		ggggg		ttty		Dec 16, 2012, 14:01	
Fu2pjuLqO9	07010			39.5883254, 2.650231		Carretera Validemos...		Palma de Mallorca		luigi		cumpleaños		Dec 15, 2012, 21:08	
BRK6hIZ7K	07010			39.5892952, 2.6498...		Carretera Validemos...		Palma de Mallorca		tirada		pizza		Dec 15, 2012, 21:07	
HHRCBmJd5R	07008			39.5815029, 2.6676...		Carrer Semolera, 6		Palma de Mallorca		ggggggg		pollo		Dec 15, 2012, 19:39	
wYos5tUuI74	07008			39.5814531, 2.6676...		Carrer Semolera, 8A		Palma de Mallorca		lleva dos meses rota		farola rota		Dec 13, 2012, 20:46	
LhCtd97ov	07008			39.5815485, 2.6675...		Carrer Semolera, 4		Palma de Mallorca		Nuevo comentario		(undefined)		Dec 13, 2012, 17:17	
aru5rK0CYC	07010			39.5960339, 2.6484...		Carrer Cas Capiscot...		Palma de Mallorca		(undefined)		(undefined)		Dec 10, 2012, 12:56	
O11zKoggyJ	07004			39.5836661, 2.6642...		Carrer Henri Dunant, 2		Palma de Mallorca		(undefined)		(undefined)		Dec 09, 2012, 20:27	

Este sistema almacena la información en forma de objetos que pueden relacionarse entre sí.

Para la aplicación se han creado tres tipos de objeto:

- Poi: contiene los punto de interés

Su estructura es la siguiente en formato JSON

```
{
  "descripcion": "lleva dos meses rota",
  "titulo": "farola rota",
  "CP": "07008",
  "Coordenadas": {
    "__type": "GeoPoint",
    "latitude": 39.5814531,
    "longitudo": 2.6676795
  },
  "Localidad": "Palma de Mallorca",
  "Direccion": "Carrer Semolera, 8A",
  "createdAt": "2012-12-13T20:46:41.317Z",
  "updatedAt": "2012-12-13T20:46:41.317Z",
  "objectId": "wYo5Btul74"
}
```

- Comentarios: contiene los comentarios asociados a un punto de interés

Su estructura es la siguiente en formato JSON

```
{
  "parent": {
    "__type": "Pointer",
    "className": "Poi",
    "objectId": "dVShuGj1Aw"
  },
  "File": {
    "__type": "File",
    "name": "157f6760-be82-48a6-a446-da137b9f92d8-imagen.png",
    "url": "http://files.parse.com/4ce6afd8-5390-4e86-84e6-1e743f3903e9/157f6760-be82-48a6-a446-da137b9f92d8-imagen.png"
  },
}
```

```

"Direccion": "Carrer Jacint Verdaguer, 94",
"createdAt": "2012-12-16T21:38:13.001Z",
"updatedAt": "2012-12-16T21:38:13.001Z",
"objectId": "Q55593OdTd"
}

```

- Poilimage: contiene las imágenes asociadas a un punto de interés

Su estructura es la siguiente en formato JSON

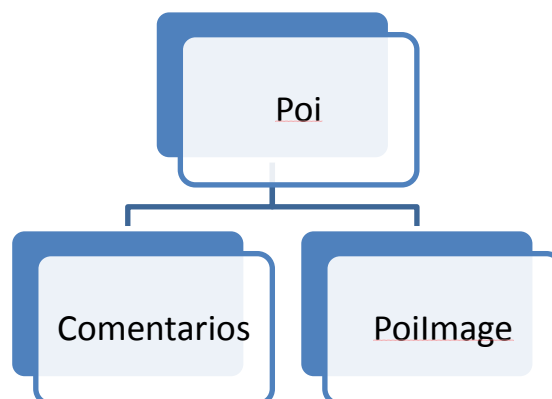
```

{
  "parent": {
    "__type": "Pointer",
    "className": "Poi",
    "objectId": "O11zKogyjJ"
  },
  "Comentario": "otro",
  "createdAt": "2012-12-16T21:24:40.719Z",
  "updatedAt": "2012-12-16T21:24:40.719Z",
  "objectId": "I1nbYuK6sa"
}

```

Se muestra la estructura de los objetos en formato JSON ya que Parse permite la exportación de todos los datos en este formato de intercambio, lo que facilitaría la migración a cualquier otro sistema de almacenamiento con un procesado previo.

La relación entre los objetos es la siguiente



La comunicación con este sistema se ha realizado mediante el Api que Parse ofrece para Java, en el apéndice se muestra código de ejemplo de comunicación usado por la aplicación.

La ventaja de usar un sistema de este tipo aparte del ahorro de tiempo en el desarrollo de una solución propia es la facilidad para migrar entre diferentes plataformas ya que se dispone de Apis para Android, IOS, Windows, OS X, JavaScript, Windows Phone Y Rest.

Aparte nos ofrece una serie de funcionalidades para la gestión de usuarios, conexión con redes sociales, integración con aplicaciones tipo cloud y ejecución de código propio en la nube.

Almacenamiento local

Android dispone de una base de datos interna de tipo SQLite en la que se pueden almacenar datos y de una API para realizar operaciones sobre la misma.

En la aplicación MapaQuejas se ha implementado una pequeña base de datos que almacena el correo electrónico de destino del mail de denuncia.

En el apéndice se muestra el código que realiza esta gestión.

Test de la aplicación final

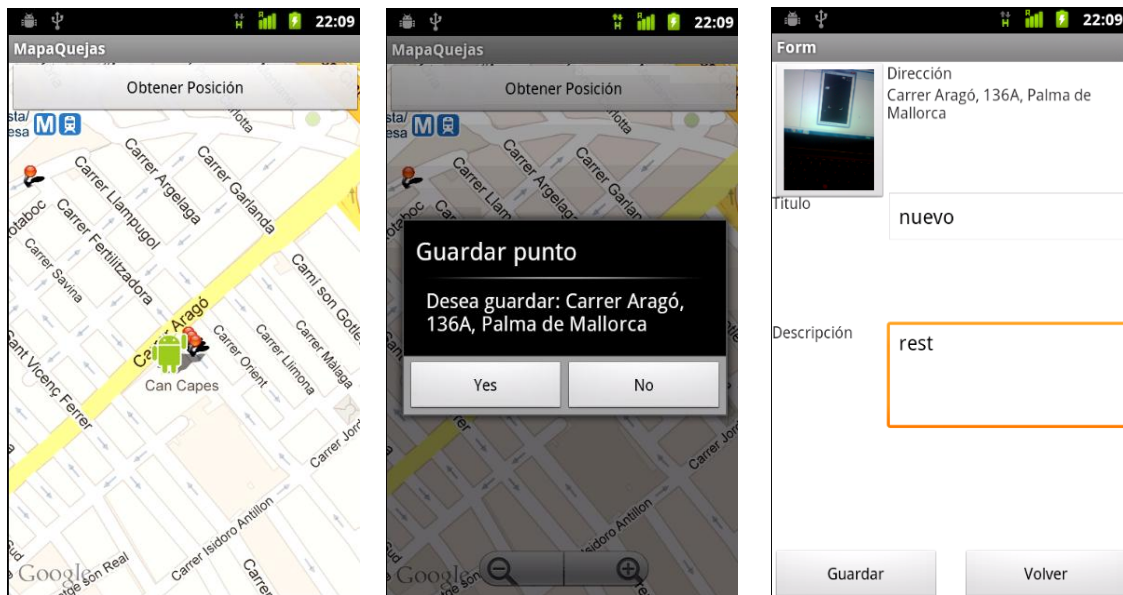
La aplicación se ha probado mediante dispositivos Android en las versiones 2.3.6 hasta la versión 4.0.4 tanto en teléfonos móviles como en tabletas.

La última versión de Android 4.1.2 se ha probado con el emulador mediante el AVD Manager (gestor de dispositivos virtuales de Android).

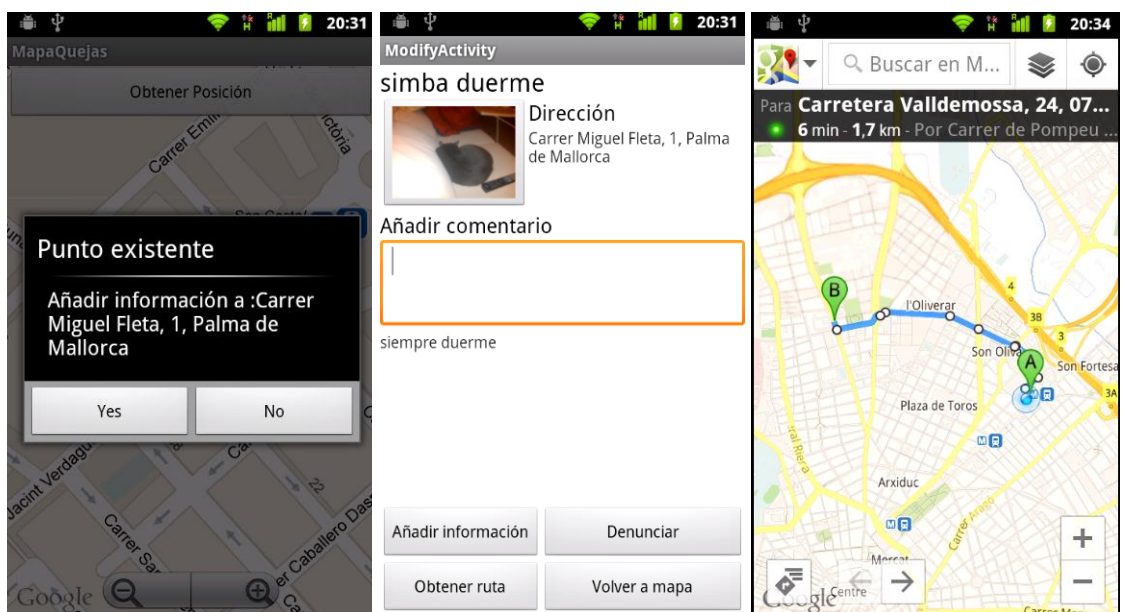
Todas las tareas definidas en la fase de análisis se han superado con éxito.

Ejemplos de captura de la aplicación:

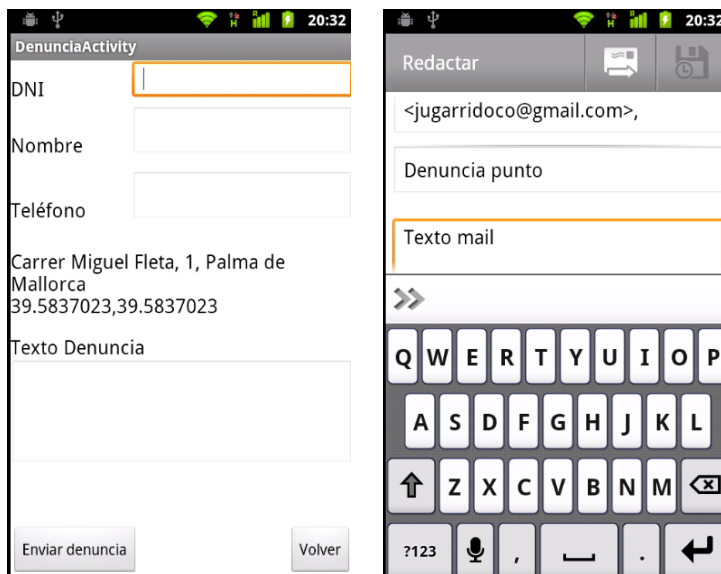
Creación de un nuevo punto (Tarea 1)



Modificar/Consultar punto (Tareas 2, 3 y 5)



Formulario denuncia (Tarea 4)



Pruebas unitarias

Las pruebas unitarias en una aplicación Android se ubican en el directorio `MapaQuejas\tests`.

`MapaQuejas/`

```

    AndroidManifest.xml

    res/
        ... (recursos MapaQuejas)

    src/
        ... (código fuente MapaQuejas) ...

    tests/
        AndroidManifest.xml

        res/
            ... (recursos MapaQuejasTest)

        src/
            ... (código fuente MapaQuejasTest)

```

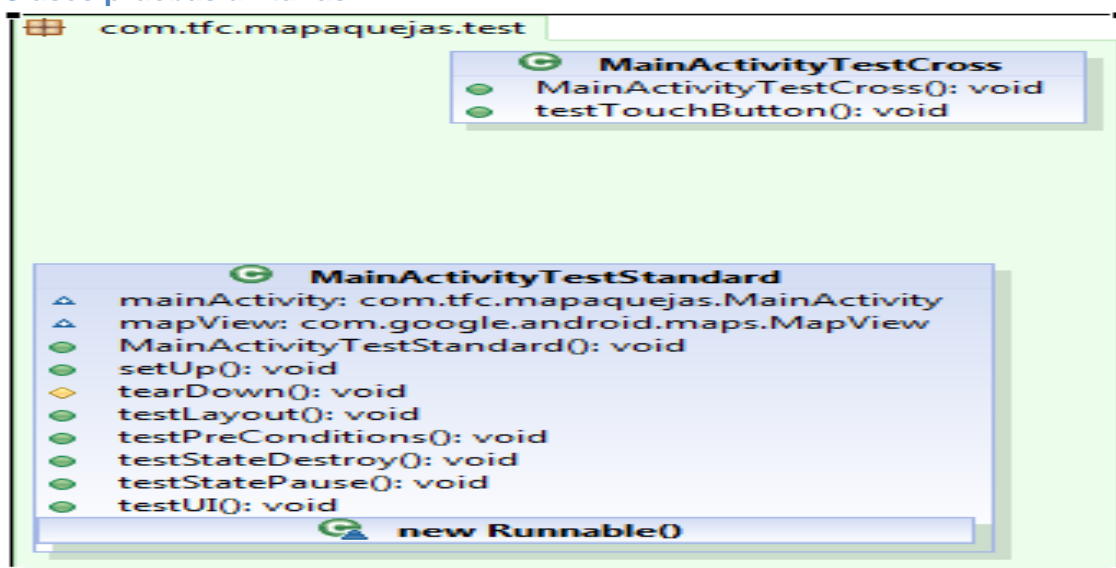
No obstante, los tests unitarios son un proyecto independiente que debe tratarse como tal dentro del entorno Eclipse.

Las pruebas unitarias se realizan mediante el uso de Android Junit Test una extensión del conocido framework Junit adaptado a Android.

El código fuente se programa de forma manual. Las subclasses principales sobre las que se suelen programar las pruebas unitarias son **ActivityInstrumentationTestCase2**, diseñada para realizar pruebas funcionales, y **ActivityUnitTestCase**, diseñada para realizar test de actividades Android de forma aislada.

Las clases creadas para realizar las pruebas se han realizado extendiendo las subclasses nombradas.

Clases pruebas unitarias



MainActivityTestCross: Extiende la clase ActivityInstrumentationTestCase2.

Métodos de prueba implementados: **testTouchButton()**

Realiza una prueba unitaria sobre la clase MainActivity, crea la actividad, localiza y simula la pulsación del botón obtener posición que contiene la actividad.

MainActivityTestStandard: Extiende la clase ActivityUnitTestCase.

Realiza las pruebas sobre la clase MainActivity.

Métodos de prueba implementados:

testPreConditions()

Realiza una prueba para comprobar la existencia de elementos dentro de la actividad principal.

testUI()

Realiza una prueba que comprueba la correcta creación de la vista de Mapa sobre la actividad principal.

testLayout()

Realiza una prueba que comprueba la existencia del botón obtener posición y de que el texto que incluye sea el correcto.

testStateDestroy():

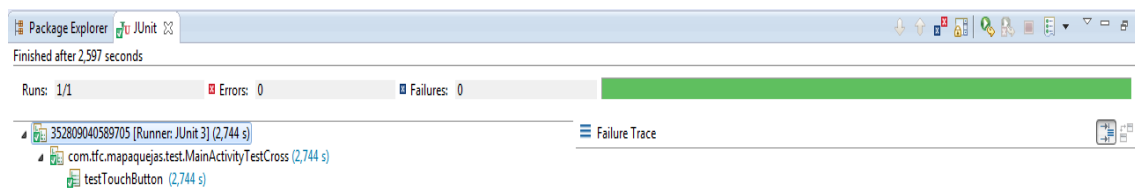
Realiza una prueba de ciclo de vida de la actividad. La actividad se destruye y se vuelve a crear.

testStatePause()

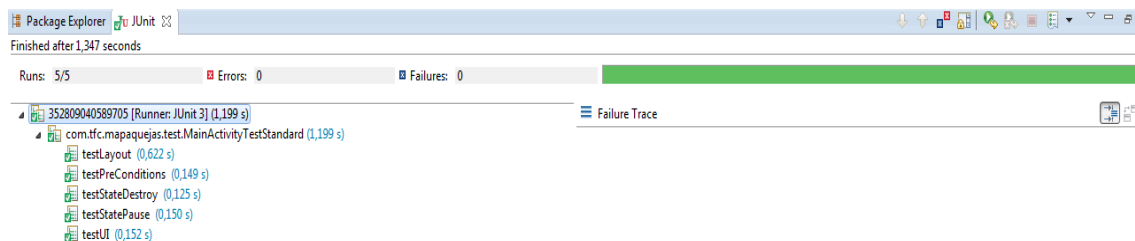
Realiza una prueba de ciclo de vida de la actividad. La actividad pasa del estado Pause al estado Resume.

Resultados pruebas unitarias

El resultado de las pruebas unitarias sobre la clase MainActivityTestCross es satisfactorio, como muestra la siguiente figura.



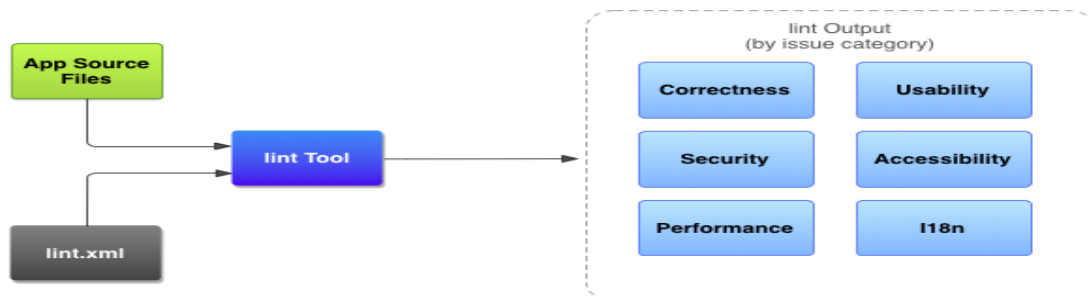
El resultado de las pruebas unitarias de la clase MainActivityTestStandard es satisfactorio, como muestra la siguiente figura.



Android Lint

Android Lint es una herramienta de análisis de código estático que comprueba los archivos de código fuente un proyecto Android ante posibles errores y

mejoras de optimización para la corrección, la seguridad, el rendimiento, la usabilidad, la accesibilidad y la internacionalización del mismo.



La herramienta Android Lint se instala por defecto con las Android Development Tools.

En el entorno Eclipse se dispone de una vista donde se pueden observar los resultados de ejecutar el análisis que realiza la herramienta.

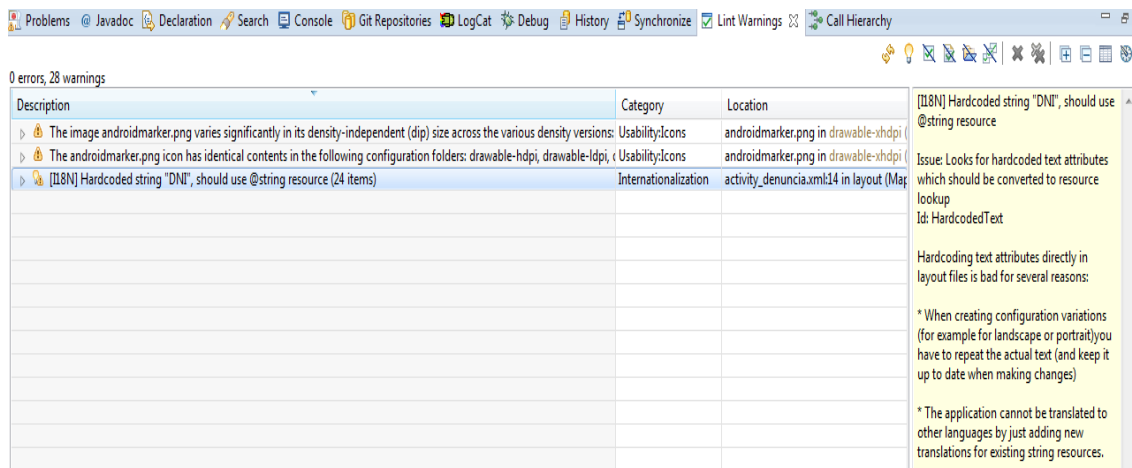
Android Lint permite configurar un archivo “lint.xml” que permite obviar ciertas comprobaciones en las que no estemos interesados o bien darles mayor importancia.

```

<?xml version="1.0" encoding="UTF-8"?>
<lint>
  <!-- Disable the given check in this project -->
  <issue id="IconMissingDensityFolder" severity="ignore" />
  <!-- Change the severity of hardcoded strings to "error" -->
  <issue id="HardcodedText" severity="error" />
</lint>
  
```

Para ejecutar la herramienta es suficiente ejecutar el botón que proporciona Eclipse para tal fin o bien generar la apk o modificar cualquier xml del apartado “res” del proyecto.

El resultado de usar la herramienta sobre el código fuente, sin configurar ningún archivo lint.xml, del proyecto MapaQuejas es el siguiente:



Como se puede observar existen advertencias acerca del tamaño de los iconos para dispositivos de alta densidad (Error definido como Usabilidad) y problemas de Internacionalización de la aplicación (Error de tipo I18n).

Los problemas de internalización se deben a que muchas de las cadenas de textos se han declarado exclusivamente en los ficheros de tipo "layout" sin ser traspasados a los ficheros de recursos de texto de la aplicación.

Manual de instalación

Para ejecutar la aplicación mediante el archivo MapaQuejas.apk es necesario conectar mediante usb un dispositivo Android al pc dónde se encuentre el fichero y alojarlo en dicho dispositivo.



La opción orígenes desconocidos debe estar marcada, dentro de

Ajustes →Aplicaciones

Mediante un explorador de ficheros debemos acceder a la ruta interna dentro del teléfono o dispositivo donde hayamos alojado el fichero de la aplicación.

Un ejemplo de explorador de ficheros es: ES File Explorer:
<https://play.google.com/store/apps/details?id=com.estrongs.android.pop>

Seleccionamos el fichero y damos los permisos para ejecutar la aplicación.

Conclusiones

Las conclusiones que se pueden derivar de la realización de este trabajo de final de carrera son muy positivas.

Los requisitos iniciales que se presentaban tras la elección del proyecto se han cumplido en su mayoría si son revisados uno a uno, lo que se puede considerar un éxito.

El hecho de trabajar con la metodología de trabajo de diseño centrado en el usuario me ha permitido mejorar mis conocimientos y valorar positivamente este tipo de técnicas para incluirlas en futuros proyectos en los que pueda participar.

Utilizar nuevas tecnologías muy actuales como el uso de un software tipo Baas, herramientas de test de usuarios o el propio desarrollo en Android que evoluciona constantemente ha supuesto un reto y una fuente de aprendizaje inimaginable con respecto a mis objetivos iniciales.

El volver a programar en Java después de haber trabajado en los últimos años en el entorno .NET, me ha servido de refresco para los conocimientos en ese lenguaje y en las herramientas de desarrollo empleadas como Eclipse, aparte de ser una dificultad añadida.

Reiterar que el aprendizaje del conocimiento del desarrollo nativo en Android conseguido proyecto ha sido enriquecedor y espero no acabe aquí ya que la sensación que me queda es que tengo mucho que aprender.

El resultado final no es perfecto, pero después de haber atravesado algunos problemas laborales, estoy contento con el resultado conseguido. En el siguiente punto me gustaría destacar los objetivos pendientes que tiene el proyecto.

Objetivos pendientes

- Mejorar requisito inicial SQLite.

En los requisitos se solicitaba gestionar la caché. Se ha usado SQLite de formar experimental para almacenar datos sin cumplir ese propósito. Quizás se debería combinar el uso de SQLite con la gestión de caché que ofrece el Baas.

- Mejorar diseño layouts en la aplicación final

El resultado final de la aplicación no se corresponde fielmente con el diseño del prototipo y además se podría mejorar algunas pantallas para mejorar su usabilidad y diseño.

- Implementar funcionalidad de votos de usuario como se definió en el prototipo

Esta funcionalidad a pesar de no plantearse como requisito si se mostró en el prototipo y resultaría interesante desarrollarla junto a una gestión de usuarios para la aplicación.

- Usar un nuevo Baas como StackMob

La elección de Parse supone tener una limitación de peticiones mensuales a su API. El baas StackMob no tiene esa limitación, por lo que resultaría interesante desarrollar en él.

- Migrar a la nueva versión de la API de Maps para Android

En diciembre de 2012 Google lanzó la nueva versión de la API de Maps para Android limitando el uso de la API anterior utilizada en este proyecto hasta marzo de 2013. Sería un requisito imprescindible migrar de API para poder mantener la aplicación activa.

- Finalizar la aplicación y publicarla en el Google Play

Como resulta evidente la aplicación no está finalizada y posiblemente contenga bugs que se deberían corregir. El reto final sería poder finalizarla y publicarla en Google Play, aunque quizás es un objetivo ambicioso para una única persona.

Software utilizado

Este es un resumen de parte del software más importante que se ha usado durante el desarrollo del proyecto.

Ms-project: Creación de diagramas y gestión del proyecto.

Prototyper: Software para la creación de prototipos de la empresa JustInMind.

UserNote: Aplicación web que permite publicar los prototipos de la aplicación prototyper y compartirlos con un grupo de usuarios.

UserTesting.com: Aplicación web en la que se publican prototipos y se realizan test de usuario.

Eclipse: Entorno de desarrollo Java

ADT Plugin para Eclipse: Plugin que permite crear proyectos de tipo aplicación Android y añade funcionalidades a Eclipse como un editor gráfico, Android Lint , DDMS, LogCat, accesos al Sdk Manager y al Virtual Device Manager

BitBucket: control de versiones de código tipo GIT

Parse: servicio de Backend donde se almacenan y consultan los datos principales de la aplicación.

Droid Eplorer: Aplicación que permite controlar un dispositivo Android y realizar screencast (videos)

Bibliografía y referencias

Materiales UOC

Módulo IHO : Diseño centrado en el usuario

ContextualEnquiry

<http://infodesign.com.au/usabilityresources/contextualenquiry/>

Visual Vocabulary Cheat Sheet

<http://www.jjg.net/ia/visvocab/>

Android Design

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

Prototyper y Usernote

<http://www.justinmind.com/>

Useresting.com

<http://www.useresting.com>

Android developers

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

Parse

<https://www.parse.com/>

StackMob

<https://www.stackmob.com/>

Wikipedia

<http://en.wikipedia.org>

Agradecimientos

A todos los que están y a aquellos que en algún momento estuvieron y por desgracia hoy no están y me apoyaron, sobre todo tu papá, nunca te olvidaré.

A mi madre por incitarme a estudiar, entre otras muchas cosas.

A mi mujer por quererme, aguantarme y por su paciencia y durante estos meses.

A Simba, mi gato, por posar para este trabajo

.A los consultores de la asignatura por su apoyo.

A la UOC por permitirme finalizar mis estudios.

Gracias a todos por permitirme acercarme a la meta.

Apéndice

Test de usuarios

A continuación se presentan los resultados de la herramienta Usertesting.com en el idioma nativo de la aplicación.

Nombre de usuario: Bradz1988

País: Reino Unido

Edad: 25

Género: Masculino

Experiencia: Usuario Promedio Web

Uso diario web: Más de 7 horas

Video: <http://www.usabilitytestresults.com/videos/qQ4pseeeANA%3d>

1. Did this site meet your expectations?

My expectation were that I could use the site to buy something, which is not what I felt it did by the end.

2. What was the best thing about this site? What was the worst thing about this site?

Id say the best thing was the navigation, it was very clear. The worst being that it was hard to understand the premise of the page.

3. If you had a magic wand, how would you improve this site?

I would add more clues at to what my goal is using the app.

4. Would you use this site in the future (please explain why or why not)?

Perhaps if I knew what the service was for.

Nombre de usuario: license2ill

País: Estados Unidos

Edad: 38

Género: Masculino

Experiencia: Usuario Promedio Web

Uso diario web: Más de 7 horas

Video: <http://www.usabilitytestresults.com/videos/ODpcwXXfj%2b8%3d>

1. **Did this site meet your expectations?**
It met expectations in terms of an interactive google map.
2. **What was the best thing about this site? What was the worst thing about this site?**
The caption looked helpful on the point. The hand marker looked pretty clear for pointing to a specific spot. The worst thing was not really knowing how to interact or what to use it for.
3. **If you had a magic wand, how would you improve this site?**
I'd lik more explanation on how to use it effectively for a specific purpose. The purpose was unclear.
4. **Would you use this site in the future (please explain why or why not)?**
I don't think I really understand what it would be useful for outside of regular use of google maps on the phone, so I guess not.

Nombre de usuario: JMontal01

País: Estados Unidos

Edad: 65

Género: Femenino

Experiencia: Usuario Promedio Web

Uso diario web: 3-5 horas

Video: <http://www.usabilitytestresults.com/videos/%2bWQ%2bs5GcgO4%3d>

1. **Did this site meet your expectations?**
I really didn't have any expectations but I did like how the site flowed with starting where you wanted to be and then if you wanted to email etc.

2. **What was the best thing about this site? What was the worst thing about this site?**

I liked that it looked like a cell phone because I am very familiar with the way to use the cell phone maps etc. I didn't like that I couldn't tell what each thing meant on the second page.

3. **If you had a magic wand, how would you improve this site?**

I would put a few more explanations so you knew what you could do with this site.

4. **Would you use this site in the future (please explain why or why not)?**

No probably not. I have an apt on my phone that does this so that's what I would use

Código referenciado

Comunicación con Parse

Ejemplo de código comentado de la clase SavePoint.java

```
//Definición nombre de objeto a consultar
ParseQuery query = new ParseQuery("Poi");
query.whereEqualTo("Direccion", adressToSave.getAddressLine(0));
query.countInBackground(new CountCallback() {
    public void done(int count, ParseException e) {
        if (e == null) {
            // Si no existe el punto lo guardamos
            if (count == 0) {
                //Nombre del objeto que se va a guardar en Parse
                final ParseObject poi = new ParseObject("Poi");
                //Campos del Objeto
                poi.put("Coordenadas", gpointToSave);
                poi.put("Direccion",adressToSave.getAddressLine(0));
                poi.put("Localidad", adressToSave.getLocality());
                poi.put("CP", adressToSave.getPostalCode());
                poi.put("titulo", quejaToSave);
                poi.put("descripcion", descripcionToSave);
                //Se comprueba si existe previamente la imagen
                if(imageCamera!= null){
                    //Objeto a guardar en Parse
                    final ParseObject poiImage = new ParseObject("PoilImage");
                    ByteArrayOutputStream bos = new ByteArrayOutputStream();
                    imageCamera.compress(CompressFormat.PNG, 0, bos);
                    //Objeto tipo fichero
                    final ParseFile file = new ParseFile("imagen.png", bos.toByteArray());
                    file.saveInBackground(new SaveCallback() {
                        public void done(ParseException e) {
                            if (e == null) {
                                //Campos del Objeto
                                poiImage.put("File", file);
```

```

poiImage.put("Direccion", adressToSave.getAddressLine(0));
//Referencia al objeto padre
poiImage.put("parent", poi);
//Guardado final de los objetos : el objeto hijo guarda al objeto padre
poiImage.saveInBackground();
} else {
    Toast.makeText(contexto, "Error comprobando datos: " + e.getMessage(),
    Toast.LENGTH_LONG).show();
}
});
} else {
    //Guardado objeto Poi sin imagen
    poi.saveInBackground();
}
} else {
    Toast.makeText(contexto, "El punto ya existe", Toast.LENGTH_SHORT).show();
}
} else {
    Toast.makeText(contexto, "Error: comprobando datos "+
    e.getMessage(), Toast.LENGTH_LONG).show();
}
}
});

```

El uso de los objetos de Parse resulta intuitivo, aunque el hecho de que las llamadas se hagan asíncronas puede inducir al programador a confusiones si no se gestiona correctamente.

Gestión SQLite

Código de la clase SQLiteDatos comentado

```

*/
public class SQLiteDatos extends SQLiteOpenHelper {

    //Sentencia SQL para crear la tabla de correo
    /** The sql create. */
    String sqlCreate = "CREATE TABLE Mail (Localidad TEXT, Email TEXT)";

    /**
     * Instantiates a new sQL lite datos.
     *
     * @param contexto the contexto
     * @param nombre the nombre
     * @param factory the factory
     * @param version the version
     */
    public SQLiteDatos(Context contexto, String nombre,
        CursorFactory factory, int version) {
        super(contexto, nombre, factory, version);
    }

    /** (non-Javadoc)
     * @see
     android.database.sqlite.SQLiteOpenHelper#onCreate(android.database.sqlite.SQLiteDatabase)

```

```

*/
@Override
public void onCreate(SQLiteDatabase db) {
    //Se ejecuta la sentencia SQL de creación de la tabla
    db.execSQL(sqlCreate);
}

/* (non-Javadoc)
 * @see
android.database.sqlite.SQLiteOpenHelper#onUpgrade(android.database.sqlite.SQLiteDatabase, int, int)
*/
@Override
public void onUpgrade(SQLiteDatabase db, int versionAnterior, int versionNueva) {

    //Se elimina la versión anterior de la tabla
    db.execSQL("DROP TABLE IF EXISTS Mail");

    //Se crea la nueva versión de la tabla
    db.execSQL(sqlCreate);
}

/**
 * Alta email.
 *
 * @param context the context
 */
public static void altaEmail(Context context){

    //Ejemplo de conexión a SQLite
    SQLiteDatos mailTable =
        new SQLiteDatos(context, "Mail", null, 1);
    SQLiteDatabase db = mailTable.getWritableDatabase();
    //Si hemos abierto correctamente la base de datos
    if(db != null)
    {
        //Generamos los datos
        String Localidad = "Palma de Mallorca";
        String Mail = "jugarridoco@gmail.com";
        //Insertamos los datos en la tabla de correo
        db.execSQL("INSERT INTO Mail (Localidad, Email) " + "VALUES (" +
Localidad + ", " + Mail + ")");
        //Cerramos la base de datos
        db.close();
    }
}

/**
 * Consulta email.
 *
 * @param context the context
 * @return the string
 */
public static String consultaEmail(Context context){

    String data="";
    SQLiteDatos mailTable =
        new SQLiteDatos(context, "Mail", null, 1);

```

```

        SQLiteDatabase db = mailTable.getWritableDatabase();
        //Si hemos abierto correctamente la base de datos
        if(db != null)
        {
            String[] args = new String[] {"Palma de Mallorca"};
            Cursor c = db.rawQuery(" SELECT Email FROM Mail WHERE
Localidad=? ", args);
            //Nos aseguramos de que existe al menos un registro
            if (c.moveToFirst()) {
                //Recorremos el cursor hasta que no haya más registros
                do {
                    data = c.getString(0);
                } while(c.moveToNext());
            }
            db.close();
        }

        return data;
    }
}

```

Como se puede observar la gestión realizada de la base de datos SQLite es testimonial en este proyecto.

Clase Actividad Principal

```

public class MainActivity extends MapActivity {

    /** The map controller. */
    private MapController mapController;

    /** The Location overlay. */
    private MyLocationOverlay LocationOverlay;

    /** The map view. */
    private MapView mapView;

    /** (non-Javadoc)
     * @see com.google.android.maps.MapActivity#isRouteDisplayed()
     */
    @Override
    protected boolean isRouteDisplayed() {
        return false;
    }

    /** (non-Javadoc)
     * @see com.google.android.maps.MapActivity#onCreate(android.os.Bundle)
     */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        //Inicializacion conexion Baas
        Parse.initialize(this, "JH92G2FUKKj4032dusQqn0eQsVmTtSysrKD6yIKe",
"GRSGjmJ2t5xUZG2YhLSJb00pCAUSbCdYV9FTWqKV");
    }
}

```

```

//Inicializacion app
super.onCreate(savedInstanceState);
setContentView(R.layout.activity_main);

//Inicializacion Vista mapa
mapView = (MapView) findViewById(R.id.mapview);
mapView.setBuiltInZoomControls(true);

//Controles capa de mapas
mapController = mapView.getController();
mapController.setZoom(18);

LocationOverlay = new MyLocationOverlay(this, mapView);

//Carga de puntos guardados
LoadPoint.loadPoints(mapView,this);

GetPosition.getMyPosition(this, LocationOverlay ,mapView, mapController);

//Ejemplo conexion SQLite
SQLiteDatos.altaEmail(this);
}

/**
 * Update position.
 *
 * @param view the view
 */
public void updatePosition(View view) {

    List<Overlay> mapOverlays = mapView.getOverlays();
    if (GetPosition.getIndexLast() != null){
        mapOverlays.remove(GePosition.getIndexLast().intValue());
    }
    GetPosition.getMyPosition(this, LocationOverlay ,mapView, mapController);
}

/* (non-Javadoc)
 * @see android.app.Activity#onConfigurationChanged(android.content.res.Configuration)
 */
@Override
public void onConfigurationChanged(Configuration newConfig) {
    super.onConfigurationChanged(newConfig);
}

/* (non-Javadoc)
 * @see com.google.android.maps.MapActivity#onPause()
 */
@Override
public void onPause() {
    super.onPause(); // Always call the superclass method first
    LocationOverlay.disableCompass();
    LocationOverlay.disableMyLocation();
}

/* (non-Javadoc)
 * @see com.google.android.maps.MapActivity#onResume()
 */
@Override
public void onResume() {
    super.onResume(); // Always call the superclass method first
}

```

```

        LocationOverlay.enableMyLocation();
        LocationOverlay.enableCompass();
    }

    /* (non-Javadoc)
     * @see android.app.Activity#onStart()
     */
    @Override
    protected void onStart() {
        super.onStart(); // Always call the superclass method first
    }

    /* (non-Javadoc)
     * @see android.app.Activity#onRestart()
     */
    @Override
    protected void onRestart() {
        super.onRestart(); // Always call the superclass method first
        // Activity being restarted from stopped state
    }

    /* (non-Javadoc)
     * @see android.app.Activity#onStop()
     */
    @Override
    protected void onStop() {
        super.onStop(); // Always call the superclass method first
    }

    /* (non-Javadoc)
     * @see com.google.android.maps.MapActivity#onDestroy()
     */
    @Override
    public void onDestroy() {
        super.onDestroy(); // Always call the superclass
        // Stop method tracing that the activity started during onCreate()
        android.os.Debug.stopMethodTracing();
    }

    /* (non-Javadoc)
     * @see android.app.Activity#onCreateOptionsMenu(android.view.Menu)
     */
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        getMenuInflater().inflate(R.menu.activity_main, menu);
        return true;
    }
}

```

Se adjunta el código de esta clase para entender el diagrama de secuencia incluido y las referencias al cumplimiento del ciclo de vida de una aplicación Android