UNIVERSIDAD
DE LA RIOJA

PROYECTO FIN DE CARRERA

Título
Desarrollo de una aplicación móvil cliente-servidor basada en Android para la configuración y control de parámetros de un vehículo particular
Autor/es
David Ruiz Urraca
Director/es
Francisco Javier Martínez de Pisón Ascacíbar
Facultad
Facultad de Ciencias, Estudios Agroalimentarios e Informática
Titulación
Proyecto Fin de Carrera
Departamento
Ingeniería Mecánica
Curso Académico
2012-2013

Desarrollo de una aplicación móvil cliente-servidor basada en Android para la configuración y control de parámetros de un vehículo particular, proyecto fin de carrera
de David Ruiz Urraca, dirigido por Francisco Javier Martínez de Pisón Ascacíbar (publicado por la Universidad de La Rioja), se difunde bajo una Licencia
Creative Commons Reconocimiento-NoComercial-SinObraDerivada 3.0 Unported.
Permisos que vayan más allá de lo cubierto por esta licencia pueden solicitarse a los titulares del copyright.

© El autor
© Universidad de La Rioja, Servicio de Publicaciones, 2013
publicaciones.unirioja.es
E-mail: publicaciones@unirioja.es

UNIVERSIDAD

DE LA RIOJA

Facultad de Ciencias, Estudios Agroalimentarios e Informática

PROYECTO FIN DE CARRERA

Ingeniería Técnica en Informática de Gestión

DESARROLLO DE UNA APLICACIÓN MÓVIL CLIENTE-SERVIDOR BASADA EN ANDROID PARA LA CONFIGURACIÓN Y CONTROL DE PARÁMETROS DE UN VEHÍCULO PARTICULAR.

Alumno: David Ruiz Urraca
Director: Francisco Javier Martínez de Pisón Ascacíbar

Logroño, junio de 2013

Resumen

En la presente memoria se especifican y detallan todos los pasos que el alumno ha realizado, desde que se propuso realizar una aplicación Android como proyecto fin de carrera, hasta que esta aplicación es totalmente funcional en el dispositivo de desarrollo y pruebas.

Agradecimientos

Muchas gracias…

… a mis padres, por haberme proporcionado una buena educación y haberme apoyado siempre.

… a mis amigos, que siempre han creído en mí, incluso en los momentos malos.

… a todos los profesores que me han enseñado a lo largo de toda mi vida, porque gracias a ellos he adquirido todos los conocimientos que ahora tengo.

… a mi familia en general por haberme animado siempre.

… a todas aquellas personas que de un modo u otro han influido en mi vida, puesto que gracias a ellos, soy la persona que soy.

Índice de contenidos

1 Introducción 11
1.1 Reflexión inicial 12
1.2 Motivación 12
2 Documento de Objetivos de Proyecto 13
2.1 Objetivos 14
2.2 Descripción 14
2.3 Justificación 14
2.4 Alcance 15
2.5 Entregables finales 15
2.6 Personal implicado 15
2.7 Riesgos 15
2.7.1 Detección de posibles riesgos 15
2.7.2 Planes de contingencia 16
2.8 Esquema de descomposición de tareas 18
2.8.1 Gestión del proyecto: 19
2.8.2 Formación 20
2.8.3 Análisis 20
2.8.4 Diseño 20
2.8.5 Construcción 21
2.8.6 Pruebas 21
2.8.7 Otra documentación 22
2.9 Tecnologías a utilizar 22
2.9.1 Servidor 22
2.9.2 Aplicación móvil 23
2.9.3 Simulador del vehículo 24
2.9.4 Comunicación 24
2.9.5 Sistema operativo de desarrollo 24
2.9.6 IDE 24
2.9.7 Generación de documentación 24

2.10 Estimación de tiempo 26
2.11 Calendario de trabajo 26
2.12 Ciclo de vida 26
3 Gestión del proyecto 29
3.1 Replanificación 30
3.2 Nueva estimación de tiempo 31
3.3 Nuevo ciclo de vida 31
3.4 Comparativa de estimación de tiempo 33
4 Análisis del proyecto 35
4.1 Especificación de requisitos 36
4.2 Roles 36
4.2.1 Usuario 36
4.2.2 Vehículo 36
4.3 Diagramas de casos de uso 37
4.3.1 Vehículo 37
4.3.2 Usuario 41
4.3.3 Servidor 45
4.4 Identificación de capas 46
4.4.1 La capa Vista 47
4.4.2 La capa Controlador 47
4.4.3 La capa Modelo 47
4.5 Diagramas de actividad 48
4.5.1 Diagrama 1 48
4.5.2 Diagrama 2 50
4.6 Diagramas de secuencia 52
4.7 Presupuesto 54
4.7.1 Listado de mano de obra 55
4.7.2 Presupuesto y mediciones 55
4.7.3 Resumen del presupuesto 56
5 Diseño del proyecto 57
5.1 Arquitectura del sistema 58

5.1.1 Capa de presentación: ... 58
5.1.2 Capa de negocio: ... 59
5.1.3 Capa de persistencia: ... 59
5.2 Patrón de diseño ... 59
5.3 Diseño de clases ... 60
5.3.1 Diagramas ... 60
5.3.2 Descripción ... 63
5.4 Diseño de la base de datos ... 68
5.4.1 Diagrama ... 68
5.4.2 Descripción ... 70
6 Construcción del proyecto ... 71
6.1 Tecnologías empleadas ... 73
6.1.1 Entorno de desarrollo ... 73
6.1.2 Alternativas de desarrollo ... 74
6.1.3 Alternativas para la base de datos ... 75
6.2 Librerías utilizadas ... 76
6.2.1 CardsUI ... 76
6.2.2 Google Maps API V2 ... 77
6.3 Estructura del proyecto ... 78
6.3.1 Cliente ... 78
6.3.2 Vehículo ... 84
6.3.3 Servidor ... 87
6.4 Interfaz ... 89
6.4.1 Usuario ... 89
6.4.2 Vehículo ... 97
7 Pruebas del proyecto ... 99
8 Conclusiones finales ... 109
8.1 Evaluación de objetivos y conclusión ... 111
8.2 Tiempo dedicado ... 112
8.3 Mejoras y ampliaciones ... 113
8.4 Conocimientos adquiridos ... 114

9 Bibliografía 117
9.1 Libros 118
9.2 Recursos web 118
10 Anexos 119
10.1 Otra documentación 121
10.1.1 Arquitectura de Android 121
10.1.2 Código de la aplicación 124
10.2 Actas de reunión 128
10.2.1 Acta 1 128
10.2.2 Acta 2 129
10.2.3 Acta 3 130
10.2.4 Acta 4 131
10.2.5 Acta 5 132
10.2.6 Acta 6 133
10.2.7 Acta 7 134
10.2.8 Acta 8 135
10.2.9 Acta 9 136
10.2.10 Acta 10 137
10.2.11 Acta 11 138
10.2.12 Acta 12 139
10.2.13 Acta 13 140
10.2.14 Acta 14 141
10.2.15 Acta 15 142
10.2.16 Acta 16 143
10.2.17 Acta 17 144
10.2.18 Acta 18 145
10.2.19 Acta 19 146
10.2.20 Acta 20 147
10.2.21 Acta 21 148
10.2.22 Acta 22 149
10.2.23 Acta 23 150

1 Introducción

1.1 Reflexión inicial

En la actualidad, el acceso rápido a la información desde cualquier dispositivo cada vez cobra mayor importancia gracias al auge de los dispositivos móviles y las tarifas de internet. Por otra parte, un ámbito que aún no ha sido demasiado explotado para los usuarios pero que es considerado como muy importante, es el de la monitorización de cualquier sistema electrónico, desde dichos dispositivos móviles. Por este motivo parece evidente que desarrollar un sistema de estas características orientado a los vehículos particulares cobra aún mayor interés.

En un primer momento, dicho sistema estaría orientado más hacia el mundo empresarial, ya que no es muy corriente disponer de una conexión a internet dedicada para el vehículo. Aunque la estimación vendría a indicar que en un futuro no muy lejano esto sería algo más común y por tanto dicho sistema podría ser fácilmente adaptable a usuarios fuera del ámbito empresarial.

1.2 Motivación

En estos tiempos tan cambiantes, la informática se ha convertido en una especialidad en la que los sujetos no pueden dejar de aprender nuevas tecnologías, lenguajes y plataformas. Comparando el índice de adopción de los ordenadores personales que empezó en la década de los 70 hasta hoy, con el índice de adopción de telefonía móvil y el índice de acceso a internet en ambas plataformas, se observa una clara tendencia en su uso que indica que cada vez más, la telefonía móvil está desbancando al clásico ordenador de sobremesa o portátil.

Algunos analistas denominan esta época, la época post-pc aunque este es un término demasiado aventurado puesto que estos dispositivos en movilidad de momento no tienen la potencia ni las prestaciones suficientes como para poder desbancar por completo al ordenador clásico de sobremesa/portátil, aunque si que son un buen sustituto para ciertas tareas para las que antes el equipo tradicional era la única alternativa viable.

En resumen, una persona que en la actualidad quiera dedicarse al mundo de la informática, necesita adaptar sus conocimientos hacia una nueva área que son los dispositivos móviles. Esto es lo que ha hecho que a la hora de afrontar el proyecto fin de carrera, el alumno se decida por este tipo de plataformas, ya que no solamente servirá como un buen ejercicio de aprendizaje si no que posteriormente abrirá puertas para un futuro mercado laboral, en una situación tan difícil como la actual y con el sector informático en pleno proceso de “smartphonización”.

2 Documento de Objetivos de Proyecto

El Documento de Objetivos de Proyecto (DOP) es un documento muy importante en cualquier proyecto. En este capítulo se especifican entre otras cosas las motivaciones del proyecto, la descripción del mismo, el alcance, etc.

Este documento consta de los siguientes apartados.

- Objetivos
- Descripción
- Justificación
- Alcance
- Entregables finales
- Personal implicado
- Riesgos
- Esquema de descomposición de tareas
- Tecnologías a utilizar
- Estimación de tiempo
- Calendario de trabajo
- Ciclo de vida

2.1 Objetivos

El objetivo del proyecto es poder disponer de una aplicación para el seguimiento y la gestión de un vehículo o flota de vehículos, accesible allá donde nos encontremos, en forma de aplicación para dispositivo móvil.

2.2 Descripción

Al ser este un proyecto que comienza desde cero, habrá que desarrollar una aplicación para el vehículo con la que se registrarán una serie de datos del estado actual del mismo, una aplicación para el dispositivo móvil del usuario con la cual poder leer dichos datos, y por último un servidor encargado de centralizar y almacenar dichos datos.

2.3 Justificación

En Estados Unidos ya existen algunas compañías aseguradoras que tienen servicios como desbloqueo de coches cuando el usuario se ha dejado las llaves dentro. Por otra parte en España se están empezando a comercializar sistemas de seguridad y localización de vehículos en caso de que estos hayan sido sustraídos.

Por este mismo motivo, se ha visto que podría ser muy interesante un servicio que no sólo sirva para estos dos casos, si no que también ayude al seguimiento del estado del coche.

2.4 Alcance

El proyecto se considerará completo, cuando se haya obtenido una aplicación cliente desde la cual poder consultar en todo momento los últimos estados del vehículo, un servidor en el cual poder almacenar dichos datos y una aplicación que simule la recogida de datos del estado del coche.

El alumno es consciente de que se podría obtener una interacción real del propio coche con un dispositivo, gracias al protocolo OBD2, pero este apartado quedará fuera del proyecto al considerarse que la entidad del mismo ya es suficientemente completa.

2.5 Entregables finales

- Memoria del proyecto
- Aplicación móvil
- Scripts de creación de la BBDD del sevidor
- Scripts para el webservice del servidor.
- Aplicación para la simulación de la recogida de datos del vehículo

2.6 Personal implicado

- Director de proyecto: Francisco Javier Martínez de Pisón Ascacíbar
- Proyectante: David Ruiz Urraca

2.7 Riesgos

Durante el proyecto fin de carrera pueden surgir imprevistos, que pueden retrasar el proyecto. Se explicarán cuáles son estos riesgos, las soluciones que se proponen en el caso de que ocurran, la probabilidad de que estos se produzcan y el momento previsto para su detección.

2.7.1 Detección de posibles riesgos

- Errores en la estimación de fechas: A pesar de haber adquirido los conocimientos teóricos necesarios durante la carrera y haber estado durante más de tres años trabajando en una de las empresas de desarrollo de más

renombre en el ámbito regional, la experiencia del alumno a la hora de afrontar un proyecto de esta entidad es bastante limitada, por lo que un error de este tipo es más que probable.
    - Probabilidad: Alta
    - Posible momento: Al principio del mismo
- Insuficiente conocimiento en algunas áreas: El desarrollo para plataformas móviles es algo nuevo para el alumno, por lo que es muy probable que tenga que dedicarle tiempo al aprendizaje y estudio de dicha plataforma.
    - Probabilidad: Alta
    - Posible momento: A lo largo de todo el proyecto
- Posibles problemas en el entorno de desarrollo. Aquí se engloban todos los posibles problemas derivados de fallos técnicos, tanto en el entorno de desarrollo, como en el sistema de comunicaciones.
    - Probabilidad: Baja
    - Posible momento: A lo largo de todo el proyecto

- Errores en el diseño del proyecto: Al igual que con el insuficiente conocimiento, se intentará realizar un diseño lo más correcto posible, pero podría ser que dicho diseño sea incompleto y haya que rehacerlo.
    - Probabilidad: Alta
    - Posible momento: A lo largo de todo el proyecto

2.7.2 Planes de contingencia

- Errores en la estimación de fechas: Debido a la inexperiencia del alumno en proyectos de esta entidad, es un riesgo a tener muy en cuenta, por tanto la única solución sería volver a planificar la estimación de fechas del mismo.
- Insuficiente conocimiento en algunas áreas: Dicho problema se tendrá en cuenta antes del comienzo de desarrollo del mismo y se dedicará una parte de la planificación al estudio de la plataforma, pero en caso de que el tiempo dedicado a ello no sea suficiente, el alumno deberá dedicar horas extras para evitar que se vea afectada la fecha final del mismo.
- Posibles problemas en el entorno de desarrollo: Este riesgo en principio es el menos probable de todos, pero en caso de ocurrir, sería de gran impacto, por lo que el alumno va a utilizar plataformas como GitHub.com, Dropbox o Google Drive para ir guardando copias de seguridad.
- Errores en el diseño del proyecto: En caso de que el diseño sea incompleto o incorrecto, se procederá a realizar las modificaciones pertinentes intentando que el impacto de las mismas no afecte a lo realizado con anterioridad ni a la fecha final del mismo.

Cualquiera de los cuatro puntos en los cuales hay un posible riesgo, podría suponer un impacto en la extensión en el tiempo del proyecto.

En el hipotético caso de que se produjese cualquiera de los posibles riesgos ya mencionados, el alumno intentará aumentar el número diario de horas dedicadas al mismo para evitar alterar la fecha de finalización.

En el caso de que dicha medida no fuera suficiente, se realizaría una modificación en la estimación de fecha.

2.8 Esquema de descomposición de tareas

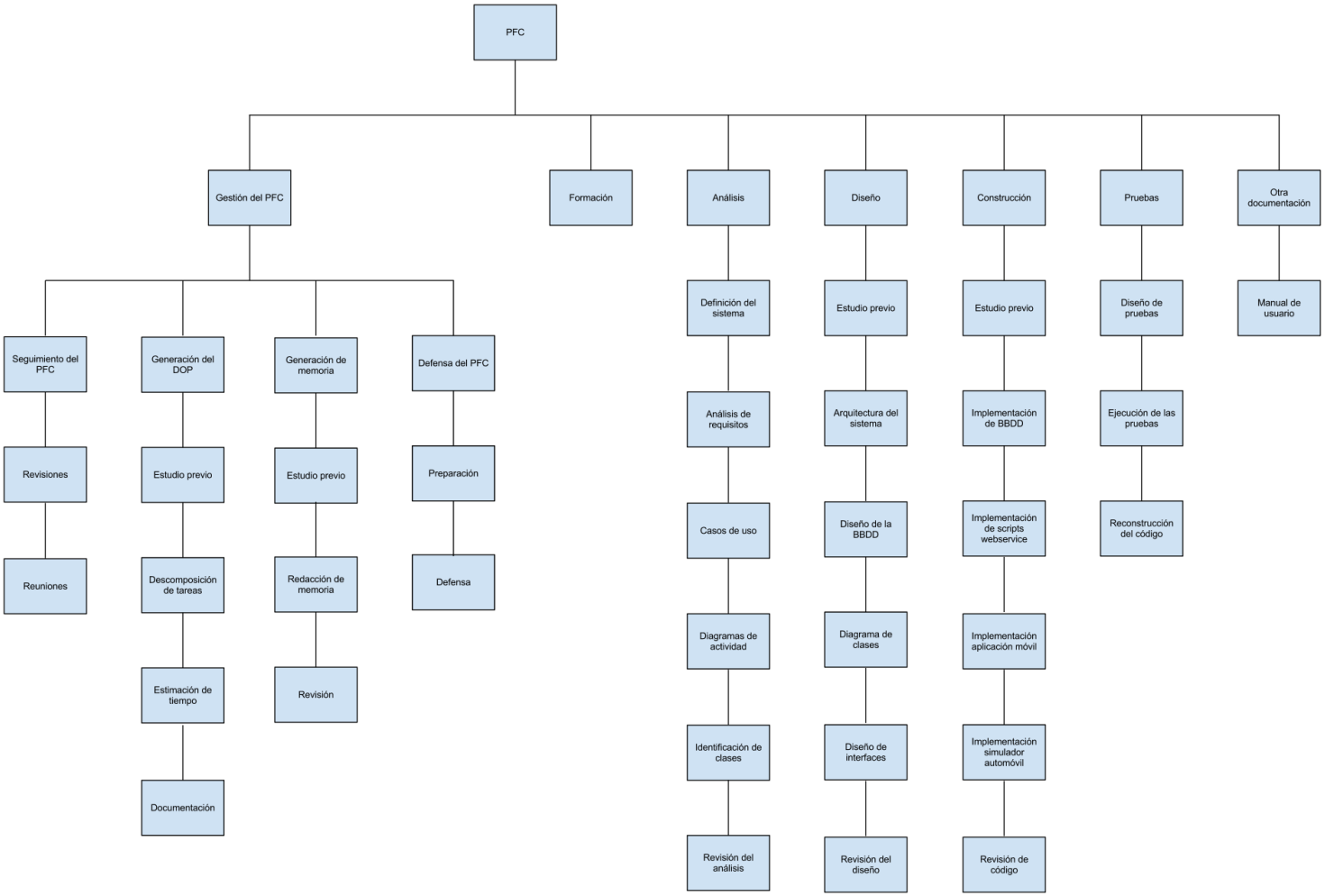


Diagrama 1

2.8.1 Gestión del proyecto:

Conjunto de tareas formales a realizar desde el comienzo del proyecto hasta su finalización, entrega y defensa.

2.8.1.1.0 Seguimiento del proyecto

Conjunto de tareas críticas para la correcta realización del mismo.

2.8.1.1.1 Revisiones

Conjunto de comprobaciones para asegurar la coherencia del proyecto y su evolución.

2.8.1.1.2 Reuniones

Una serie de revisiones periódicas entre el alumno y el director del proyecto para asegurar el correcto seguimiento y ejecución del proyecto.

2.8.1.2.0 Generación del DOP

Tareas necesarias para la generación del documento de objetivos del proyecto fin de carrera.

2.8.1.2.1 Estudio previo

Recopilación y estudio de documentación obtenida durante la carrera, y análisis de proyectos anteriormente realizados.

2.8.1.2.2 Descomposición de tareas

Estructuración de todas las tareas que componen dicho proyecto y ordenación de las mismas en un diagrama EDT (Estructura de descomposición del Trabajo) para asegurar la mejor organización posible del mismo.

2.8.1.2.3 Estimación de tiempo

Realización de un diagrama de Gantt asignando a cada tarea la estimación de tiempo correspondiente y de esta forma, calcular de forma aproximada la duración completa del proyecto

2.8.1.2.4 Documentación

Otro tipo de documentación perteneciente al documento de objetivos no reflejada en ninguno de los puntos anteriores

2.8.1.3.0 Generación de la memoria

Tareas necesarias para la realización de la memoria del proyecto final de carrera

2.8.1.3.1 Estudio previo

Recopilación y estudio de documentación obtenida durante la carrera, y análisis de proyectos anteriormente realizados

2.8.1.3.2 Redacción de memoria

Redacción de del documento en el que se reflejarán los pasos seguidos para la correcta realización del proyecto, así como posibles cambios a lo largo del mismo.

2.8.1.3.3 Revisión

Revisión del documento "Memoria del proyecto" tanto por parte del alumno, como por parte del director de proyecto para su valoración y posible corrección o modificación.

2.8.1.4.0 Defensa del PFC

Tareas relacionadas con la presentación y defensa del proyecto.

2.8.1.4.1 Preparación

Estudio exhaustivo de las partes más importantes del proyecto fin de carrera, y preparación de la presentación a realizar del proyecto, así como asistencia a defensas de otros proyectos y estudio de las mismas.

2.8.1.4.2 Defensa

Presentación y defensa del proyecto fin de carrera del alumno frente al tribunal académico.

2.8.2 Formación

Conjunto de tareas relacionadas con el estudio de la plataforma seleccionada, y de las tecnologías necesarias para poder llevar a cabo el mismo. Dicha tarea se considera una de las más importantes de dicho proyecto, ya que supone el estudio de una plataforma nueva para el alumno, ya que no ha cursado ninguna asignatura relacionada con plataformas ni tecnologías móviles. De esta tarea dependerá en gran parte el éxito o fracaso del resultado final.

2.8.3 Análisis

En esta etapa se realizará un estudio de cuáles son los objetivos del proyecto y de sus características técnicas.

2.8.3.1.0 Definición del sistema

Estudio exhaustivo del alcance del proyecto, sus características, limitaciones, etc.

2.8.3.2.0 Análisis de requisitos

Se realizará un estudio de los requisitos de la aplicación, la viabilidad de los mismos, intentando evitar una futura modificación de los mismos.

2.8.3.3.0 Casos de uso

Se reflejarán en forma de diagramas de casos de uso los posibles actores que pudieran intervenir en la aplicación, así como los posibles usos de la misma.

2.8.3.4.0 Diagramas de actividad

Se realizarán los diagramas necesarios para reflejar el flujo de información en la aplicación.

2.8.3.5.0 Identificación de clases

Análisis inicial de las posibles clases de las clases del sistema para poder hacernos una idea inicial de la estructura del mismo.

2.8.3.6.0 Revisión del análisis

Revisión y corrección de errores de toda la información generada en la fase de análisis.

2.8.4 Diseño

En esta fase se alcanza con mayor precisión una solución óptima de la aplicación, teniendo en cuenta los recursos físicos como lógicos del sistema.

2.8.4.1.0 Estudio previo

Recopilación de todo tipo de información referida al proyecto o que pueda ser necesaria para el mismo

2.8.4.2.0 Arquitectura del sistema

Realización de un diseño del mismo lo más previsor posible para adelantarnos a posibles fallos posteriores.

2.8.4.3.0 Diseño de la BBDD

Definición de valores a almacenar en la base de datos y estructuración de los mismos en las posibles distintas tablas y las relaciones entre ellas. En esta fase se incluye tanto la creación de la base de datos del servidor, como la base de datos (de menor entidad) del dispositivo móvil, que usaremos sobre todo a modo de caché para cuando la conexión con el servidor no esté disponible.

2.8.4.4.0 Diagrama de clases

Identificación de las clases que componen el sistema y las relaciones entre las mismas.

2.8.4.5.0 Diseño de interfaces

En esta fase se realizará el diseño de la interfaz de la aplicación, intentando adelantarnos a lo que el futuro usuario vaya a querer/necesitar, y de esta forma intentando hacer la interfaz más intuitiva y fácil posible, y a la vez adaptándonos a las guías de diseño del propio sistema operativo elegido.

2.8.4.6.0 Revisión del diseño

Revisión y corrección de errores surgidos durante la fase de análisis del sistema.

2.8.5 Construcción

Fase operativa en la que se realiza el trabajo.

2.8.5.1.0 Estudio previo

Se realizará una recopilación de los conocimientos adquiridos durante la carrera, y se elegirán las herramientas adecuadas para su realización.

2.8.5.2.0 Implementación de BBDD

Instalación y configuración del servidor y creación y despliegue de la base de datos.

2.8.5.3.0 Implementación de scripts webservice

Una vez diseñado el protocolo de comunicación entre dispositivo móvil y servidor, se procederá a crear el webservice que será el que realice dicha actividad en el servidor.

2.8.5.4.0 Implementación aplicación móvil

Desarrollo de la aplicación cliente en el dispositivo móvil. En esta fase se incluyen tanto el código funcional de la aplicación, como su UI y la BBDD interna.

2.8.5.5.0 Implementación simulador automóvil

Al no disponer con un vehículo con las características deseadas, se desarrollará una aplicación que simulará la recogida de datos dicho vehículo.

2.8.5.6.0 Revisión de código

Revisión y corrección de errores en el código de la aplicación.

2.8.6 Pruebas

En esta fase se crearán un conjunto de pruebas para comprobar el correcto funcionamiento de la aplicación.

2.8.6.1.0 Diseño de pruebas

Se diseñarán dichas pruebas atendiendo a criterios de funcionalidad y usabilidad, intentando contemplar por un lado todas las posibilidades que presenta la aplicación y final, y su correcta usabilidad.

2.8.6.2.0 Ejecución de las pruebas

Ejecución de las pruebas creadas para el testeo y extracción de posibles errores y/o fallos de la aplicación

2.8.6.3.0 Reconstrucción del código

Corrección de errores detectados en la fase anterior, y rediseño en caso de que los errores sean de usabilidad.

2.8.7 Otra documentación

Otra documentación asociada al proyecto, no incluida en ninguna de las fases anteriores.

2.8.7.1.0 Manual de usuario

Documento en el que se explicará de forma lo más natural posible, todas las funcionalidades que nos ofrece la aplicación, y la correcta forma para realizarlas.

2.9 Tecnologías a utilizar

2.9.1 Servidor

Para la construcción del servidor se necesitan dos tecnologías diferentes, la primera para bases de datos y la segunda para el servicio Web y posible cliente web.

2.9.1.1.0 Bases de datos

El sistema de gestión de bases de datos elegido es mysql por dos motivos fundamentales. El primero, es un sistema ya conocido por el alumno y por tanto la curva de aprendizaje será menor. El segundo, es un sistema libre (por ahora, ya que se rumorea que deje de serlo al haber sido adquirido hace no demasiado tiempo por Oracle Corporation). Además de lo ya mencionado, es un sistema potente, relacional, multihilo y multiusuario.

2.9.1.2.0 PHP

El alumno se ha decidido por PHP como lenguaje de programación por su facilidad de uso y similitud con otros lenguajes de programación ya conocidos por el alumno. Otro motivo de esta elección es la gran integración que presenta con el SGBD escogido en el punto anterior, que limita posibles problemas añadidos.

2.9.1.3.0 Apache

Se ha apostado por Apache como servidor web http por su filosofía de código abierto y por su esencia multiplataforma. De esta forma, si el sistema operativo en el que se desarrollará el proyecto no es suficiente o sufre algún problema, siempre se podrá preparar otro equipo u otro sistema operativo, sin que esto suponga mucho cambio. Además de esto, también se ha elegido por la gran documentación existente, su gran calidad como software y por su gran integración con las dos tecnologías elegidas de antemano (PHP y MySql)

2.9.1.4.0 Nota

En fases más tempranas el alumno se planteó el uso de Ruby On Rails como lenguaje de programación (si hablamos con propiedad RoR no es un lenguaje de programación si no un framework escrito en Ruby) por su gran versatilidad, simplicidad y proyección en el mundo laboral, pero hubiera supuesto un incremento notable en la fase de Formación, así que finalmente se descartó. Aun así no se descarta un futuro portar el sistema a RoR, si los conocimientos alcanzados por el usuario así lo permiten.

2.9.2 Aplicación móvil

2.9.2.1.0 Android

Para este punto se han barajado como opciones, los dos sistemas operativos móviles con mayor impacto en el mercado que son Android y iOS pero finalmente se ha decidido que Android sea el elegido por los siguientes motivos.

2.9.2.1.1 *Mercado mayor y de mayor crecimiento:* El parqué de dispositivos Android en el mercado es significativamente superior al de iOS y las previsiones dicen que esta diferencia será más grande en el futuro.

2.9.2.1.2 *El lenguaje de programación:* A día de hoy existen varios frameworks que generan aplicaciones nativas para ambas plataformas, pero el rendimiento de dichas aplicaciones es sustancialmente inferior al de las aplicaciones realizadas con el SDK oficial de la plataforma, así que la decisión se resume en "Java Vs Objective-C". De esta forma, en caso de elegir desarrollar para iOS habría que replanificar la estimación de horas y aumentar la duración de la fase de formación, cosa que no sucedería en caso de elegir Android como plataforma.

2.9.2.1.3 *Desembolso económico:* En el momento en el que se realiza este proyecto, la única opción para desarrollar una aplicación iOS pasa por adquirir un ordenador de la marca Apple, sin embargo para desarrollar para Android podemos escoger cualquiera de los 3 sistemas operativos más importantes Windows, Mac OS X y Linux (por orden en cuanto a cuota de uso) y el alumno adquirió hace relativamente poco tiempo un portátil así que no se plantea la adquisición de otro equipo.

2.9.2.1.4 *Por otra parte, esto también plantea otros inconvenientes,* como por ejemplo la mayor fragmentación de la plataforma, y algunos estudios que indican que el usuario de iOS está más predispuesto a pagar una aplicación que uno de Android.

2.9.2.2.0 Sqlite

Se ha optado por un SGBD como SQLite para almacenar datos en el dispositivo móvil a modo de caché por su gran ligereza y

manipulabilidad desde cualquier sistema (Android, iOS, Windows Phone, Firefox, Chrome, etc)

2.9.3 Simulador del vehículo

2.9.3.1.0 Android

Por los mismos motivos que en la aplicación móvil

2.9.4 Comunicación

Para la comunicación entre los sistemas se han barajado dos opciones

2.9.4.1.0 XML

Lenguaje de marcas desarrollado por el World Wide Web Consortium (W3C) que permite definir la gramática de lenguajes específicos para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones se deben comunicar entre sí o integrar información. Además es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

2.9.4.2.0 JSON

Formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript que no requiere el uso de XML. La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX. Una de las ventajas de JSON sobre XML como formato de intercambio de datos en este contexto es que es mucho más sencillo escribir un analizador sintáctico (parser) de JSON. En JavaScript, un texto JSON se puede analizar fácilmente usando el procedimiento eval(), lo cual ha sido fundamental para que JSON haya sido aceptado por parte de la comunidad de desarrolladores AJAX, debido a la ubicuidad de JavaScript en casi cualquier navegador web.

2.9.4.3.0 Decisión final

La decisión final ha sido JSON por la gran cantidad de documentación disponible para el mismo, y la inmensa cantidad de recursos disponibles para el mismo.

2.9.5 Sistema operativo de desarrollo

De las dos opciones disponibles por el alumno (Windows y Linux, recordemos que se ha descartado el desarrollo para iOS, entre otras cosas, por no disponer de un sistema Mac OS X) se ha optado por utilizar linux por su mayor flexibilidad y por ser un sistema que el alumno utiliza con asiduidad.

2.9.6 IDE

En el desarrollo de aplicaciones para Android, Google (la desarrolladora del sistema operativo) aconseja utilizar Eclipse, y al ser este un IDE ya conocido por el usuario, la elección es obvia.

2.9.7 Generación de documentación

Para la redacción del proyecto se han barajado tres posibilidades (todas ellas disponibles en Linux, ya que va a ser el sistema operativo seleccionado para el desarrollo)

2.9.7.1.0 LibreOffice

Suite ofimática libre y gratuita, compatible con Microsoft Windows, Mac y GNU/Linux. Cuenta con un procesador de texto (Writer), un editor de hojas de cálculo (Calc), un creador de presentaciones (Impress), un gestor de bases de datos (Base), un editor de gráficos vectoriales (Draw), y un editor de fórmulas matemáticas (Math). LibreOffice fue creada por la fundación The Document Foundation y permite guardar los archivos en un formato estándar ISO (OpenDocument) además de importar y exportar documentos en varios formatos de archivo adicionales como por ejemplo los de Microsoft Office, Rich Text Format (.rtf), archivos de texto plano (.txt), Office Open XML y OpenOffice.org XML, Microsoft Works y WordPerfect. Además, puede exportar documentos directamente a los formatos PDF y SWF. LibreOffice también cuenta con la capacidad de importar documentos en modo de «solo lectura» en los formatos Unified Office Format, Data Interchange Format y los formatos propios de Lotus 1-2-3, entre otros.

2.9.7.2.0 Calligra Suite

Suite ofimática multiplataforma, libre y de código abierto para el proyecto KDE, aunque es independiente de este. Utiliza el formato de documento abierto y estándar OASIS OpenDocument de forma nativa. Además, incluye filtros de importación para poder trabajar con algunos formatos de fichero de otras suites ofimáticas. Calligra Suite fue diseñado inicialmente para funcionar en sistemas operativos tipo Unix, pero desde la versión 2.0 es posible la ejecución de Calligra Suite en Mac OS X así como también en Windows.

2.9.7.3.0 Google Drive

Servicio de almacenamiento de archivos en línea introducido por Google el 24 de abril de 2012. La principal diferencia es que no es un programa local si no que es una webapp (o aplicación web) por lo que compartir estos archivos a través de internet es una de las cualidades más interesantes. Google Drive nos ofrece la misma variedad de programas (procesador de textos, hoja de cálculo, presentaciones…) y es capaz de compatibilizar todas las extensiones de los dos suites anteriores (e incluso MS Office o iLife) sin embargo al ser una suite con mucho menor recorrido, las limitaciones son aún demasiado importantes. Sin embargo sí que se utilizará Google Drive (en combinación con Dropbox) como sistema de copias de seguridad.

2.9.7.4.0 Decisión final

Esta no está aún tomada, pero girará en torno a las dos primeras opciones, y al ser compatibles una con la otra, no afectará demasiado.

Ventas de smartphones a nivel mundial en el tercer cuatrimestre de 2012

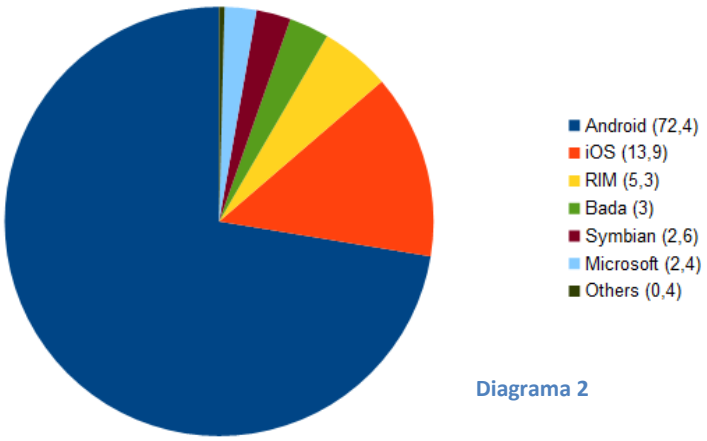


Diagrama 2

2.10 Estimación de tiempo

Tarea	Horas estimadas
Gestión del PFC	No estimado
Formación	160
Análisis	80
Diseño	116
Construcción	220
Pruebas	72
Otra documentación	8

Tabla 1

2.11 Calendario de trabajo

El alumno ahora mismo se encuentra desempleado por lo que podría dedicarle más horas a la realización del PFC pero debido a que también realiza otras actividades, como el aprendizaje de idiomas, deporte, etc, el horario real que le podrá dedicar se concentra por las mañanas. Además los fines de semana se dedicarán a comprobar si se está cumpliendo correctamente la previsión, y de no ser así se le dedicarán horas extraordinarias.

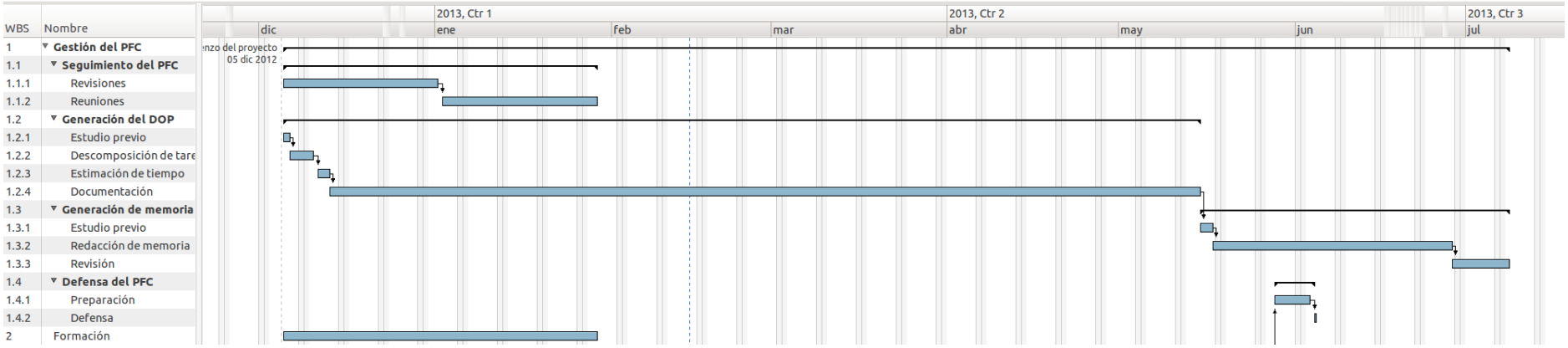
	Horario fijo
	Horas extras

	Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
08:00 09:00							
09:00 10:00							
10:00 11:00							
11:00 12:00							
12:00 13:00							
13:00 14:00							
14:00 15:00							
15:00 16:00							
16:00 17:00							
17:00 18:00							
18:00 19:00							
19:00 20:00							
20:00 21:00							
21:00 22:00							

Tabla 2

2.12 Ciclo de vida

Para la realización del proyecto, no se exige la utilización de ningún tipo de metodología y ya que se conocen de antemano casi la totalidad de los requisitos y funcionalidades, el alumno ha decidido seguir una metodología en cascada


WBS
Nombre
1 Gestión del PFC
1.1 Seguimiento del PFC
1.1.1 Revisiones
1.1.2 Reuniones
1.2 Generación del DOP
1.2.1 Estudio previo
1.2.2 Descomposición de tare
1.2.3 Estimación de tiempo
1.2.4 Documentación
1.3 Generación de memoria
1.3.1 Estudio previo
1.3.2 Redacción de memoria
1.3.3 Revisión
1.4 Defensa del PFC
1.4.1 Preparación
1.4.2 Defensa
2 Formación
2013, Ctr 1
2013, Ctr 2
2013, Ctr 3
dic
ene
feb
mar
abr
may
jun
jul
nzo del proyecto
05 dic 2012

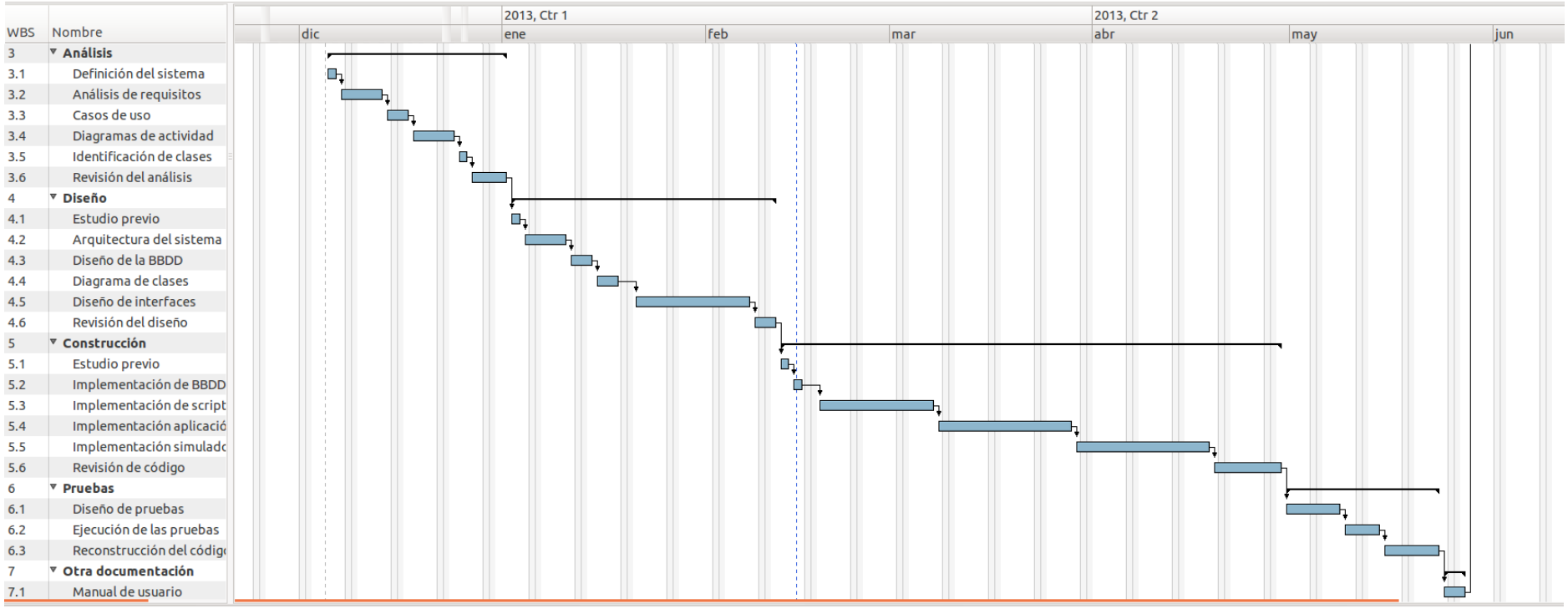

WBS
Nombre
3 Análisis
3.1 Definición del sistema
3.2 Análisis de requisitos
3.3 Casos de uso
3.4 Diagramas de actividad
3.5 Identificación de clases
3.6 Revisión del análisis
4 Diseño
4.1 Estudio previo
4.2 Arquitectura del sistema
4.3 Diseño de la BBDD
4.4 Diagrama de clases
4.5 Diseño de interfaces
4.6 Revisión del diseño
5 Construcción
5.1 Estudio previo
5.2 Implementación de BBDD
5.3 Implementación de script
5.4 Implementación aplicació
5.5 Implementación simulado
5.6 Revisión de código
6 Pruebas
6.1 Diseño de pruebas
6.2 Ejecución de las pruebas
6.3 Reconstrucción del códig
7 Otra documentación
7.1 Manual de usuario
2013, Ctr 1
2013, Ctr 2
dic
ene
feb
mar
abr
may
jun

Diagrama 3

3 Gestión del proyecto

La Gestión de Proyectos se puede describir como un proceso de planteamiento, ejecución y control de un proyecto, desde su comienzo hasta su conclusión, con el propósito de alcanzar un objetivo final en un plazo de tiempo determinado, con un coste y nivel de calidad determinados, a través de la movilización de recursos técnicos, financieros y humanos. Incorporando variadas áreas del conocimiento, su objetivo final es el de obtener el mejor resultado posible del trinomio coste-plazo-calidad.
En resumen, la gestión de proyectos suma áreas tan distintas como la incorporación del proyecto, la gestión de costes, la gestión de calidad, la gestión del tiempo, la gestión de recursos humanos o la gestión de la comunicación (entre los miembros y el exterior). Así, la gestión de proyectos forma un ciclo dinámico que transcurre del planteamiento a la ejecución y control.

3.1 Replanificación

Debido a la magnitud inicial del proyecto, durante el desarrollo del mismo se ha producido una reducción de funcionalidades que se explicarán a continuación.

La idea original del proyecto, como ya se explicó en el DOP, era una aplicación con la que poder gestionar un vehículo o flota de vehículos por completo, pero debido a la inexperiencia del alumno, y a los retrasos acumulados con anterioridad y las limitaciones físicas al no disponer de un vehículo con dichas características, se ha decidido recortar las siguientes funcionalidades.

- Posibilidad de realizar una foto o una grabación de audio del interior del vehículo
- Aviso en caso de que los acelerómetros hayan notado movimiento.
- Avisos sobre el mantenimiento del mismo (ITV, revisiones periódicas, etc)
- Mostrar información meteorológica del trayecto
- Mostrar información de tráfico
- Mostrar información de radares
- Control y reparto de carga en camiones
- Cliente web

Objetivos actuales del proyecto:

- Posibilidad de gestionar más de un vehículo
- Aviso en caso de que el coche haya sido abierto
- Comprobación de luces encendidas
- Comprobación de puertas abiertas (incluido maletero)
- Comprobación de ventanas abiertas (incluido posible techo solar)
- Comprobación de porcentaje de batería
- Comprobación de nivel de gasolina (y litros a poder ser)

- Comprobación de nivel de presión en las 4 ruedas
- Comprobación de nivel de aceite
- Comprobación de nivel de líquido de frenos
- Comprobación de nivel anticongelante
- Comprobación de kilómetros totales recorridos (Odómetro)
- Comprobación de geolocalización
- Acciones a realizar a distancia (bloqueo del coche, alarma, etc.)

Por tanto y teniendo en cuenta estas modificaciones y los retrasos actuales, la nueva planificación quedará de la siguiente forma.

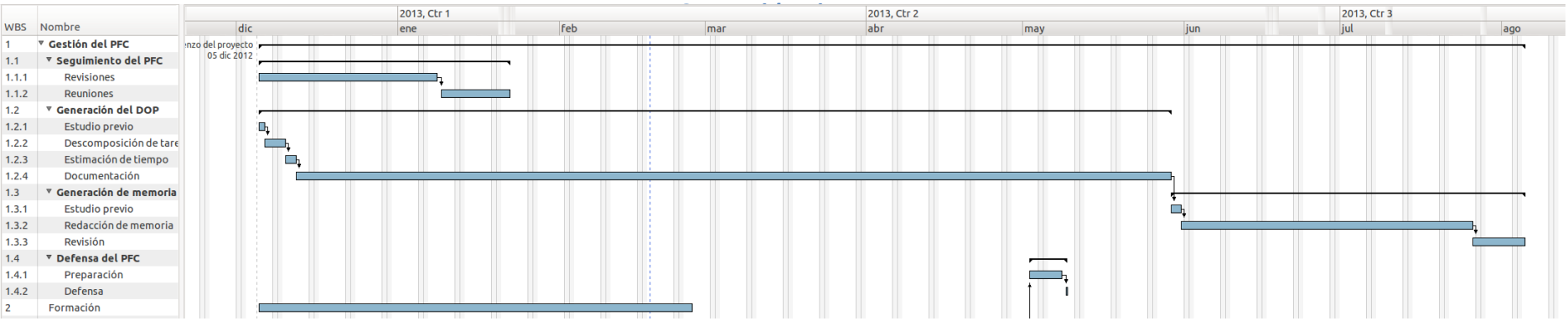
3.2 Nueva estimación de tiempo

Tarea	Horas reales
Gestión del PFC	110
Formación	240
Análisis	84
Diseño	120
Construcción	180
Pruebas	36
Otra documentación	4

Tabla 3

3.3 Nuevo ciclo de vida

Una vez modificadas las estimaciones de tiempo para la realización del proyecto, se realiza un nuevo diagrama que refleje el nuevo ciclo de vida.


WBS
Nombre
1 Gestión del PFC
1.1 Seguimiento del PFC
1.1.1 Revisiones
1.1.2 Reuniones
1.2 Generación del DOP
1.2.1 Estudio previo
1.2.2 Descomposición de tare
1.2.3 Estimación de tiempo
1.2.4 Documentación
1.3 Generación de memoria
1.3.1 Estudio previo
1.3.2 Redacción de memoria
1.3.3 Revisión
1.4 Defensa del PFC
1.4.1 Preparación
1.4.2 Defensa
2 Formación
2013, Ctr 1
2013, Ctr 2
2013, Ctr 3
dic
ene
feb
mar
abr
may
jun
jul
ago
05 dic 2012

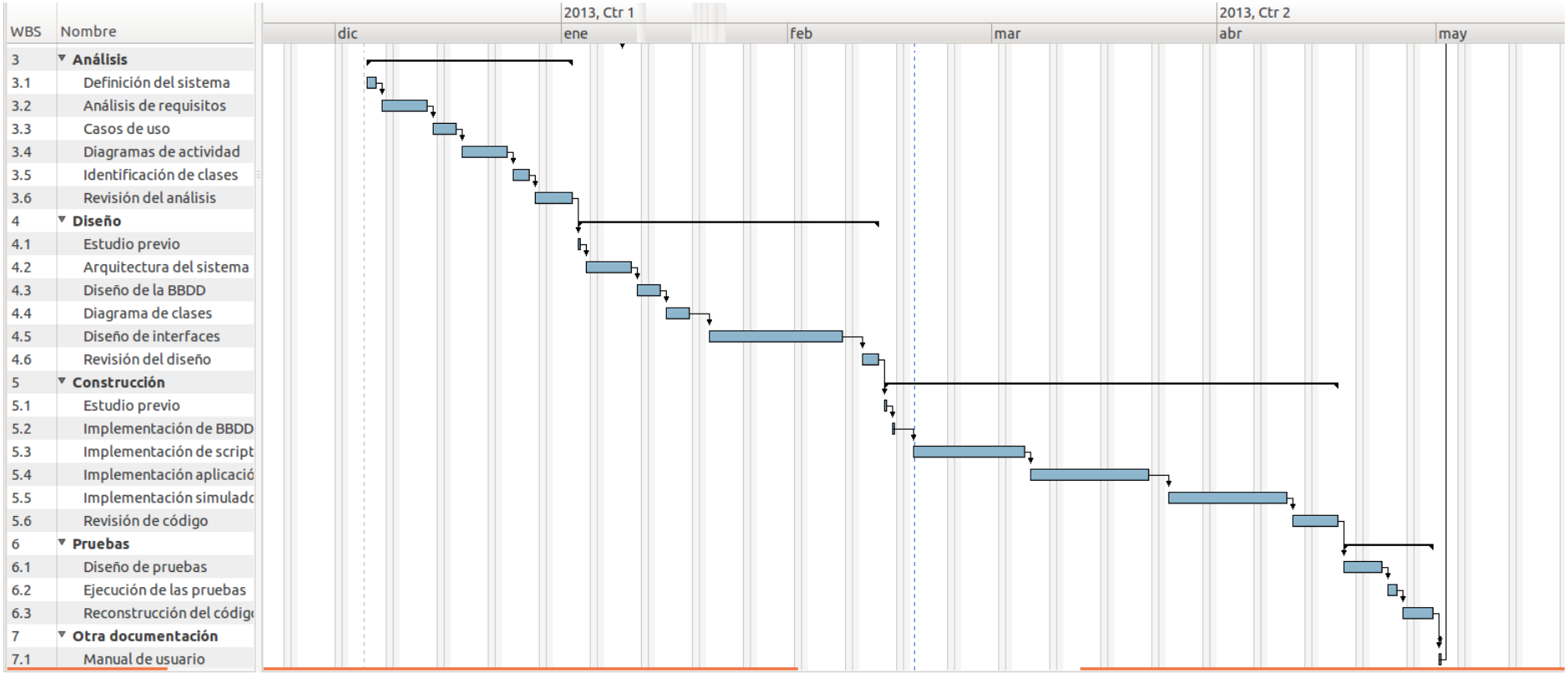

WBS
Nombre
3 Análisis
3.1 Definición del sistema
3.2 Análisis de requisitos
3.3 Casos de uso
3.4 Diagramas de actividad
3.5 Identificación de clases
3.6 Revisión del análisis
4 Diseño
4.1 Estudio previo
4.2 Arquitectura del sistema
4.3 Diseño de la BBDD
4.4 Diagrama de clases
4.5 Diseño de interfaces
4.6 Revisión del diseño
5 Construcción
5.1 Estudio previo
5.2 Implementación de BBDD
5.3 Implementación de script
5.4 Implementación aplicació
5.5 Implementación simulado
5.6 Revisión de código
6 Pruebas
6.1 Diseño de pruebas
6.2 Ejecución de las pruebas
6.3 Reconstrucción del códig
7 Otra documentación
7.1 Manual de usuario
2013, Ctr 1
2013, Ctr 2
dic
ene
feb
mar
abr
may

Diagrama 4

3.4 Comparativa de estimación de tiempo

Como se puede observar en la tabla, el tiempo destinado a cada una de las fases, en ningún caso coincide con la estimación inicial. También cabe resaltar, que a pesar de ser estimaciones, el desvío no ha sido especialmente significativo, reduciéndose alguna de dichas fases y aumentándose otras.

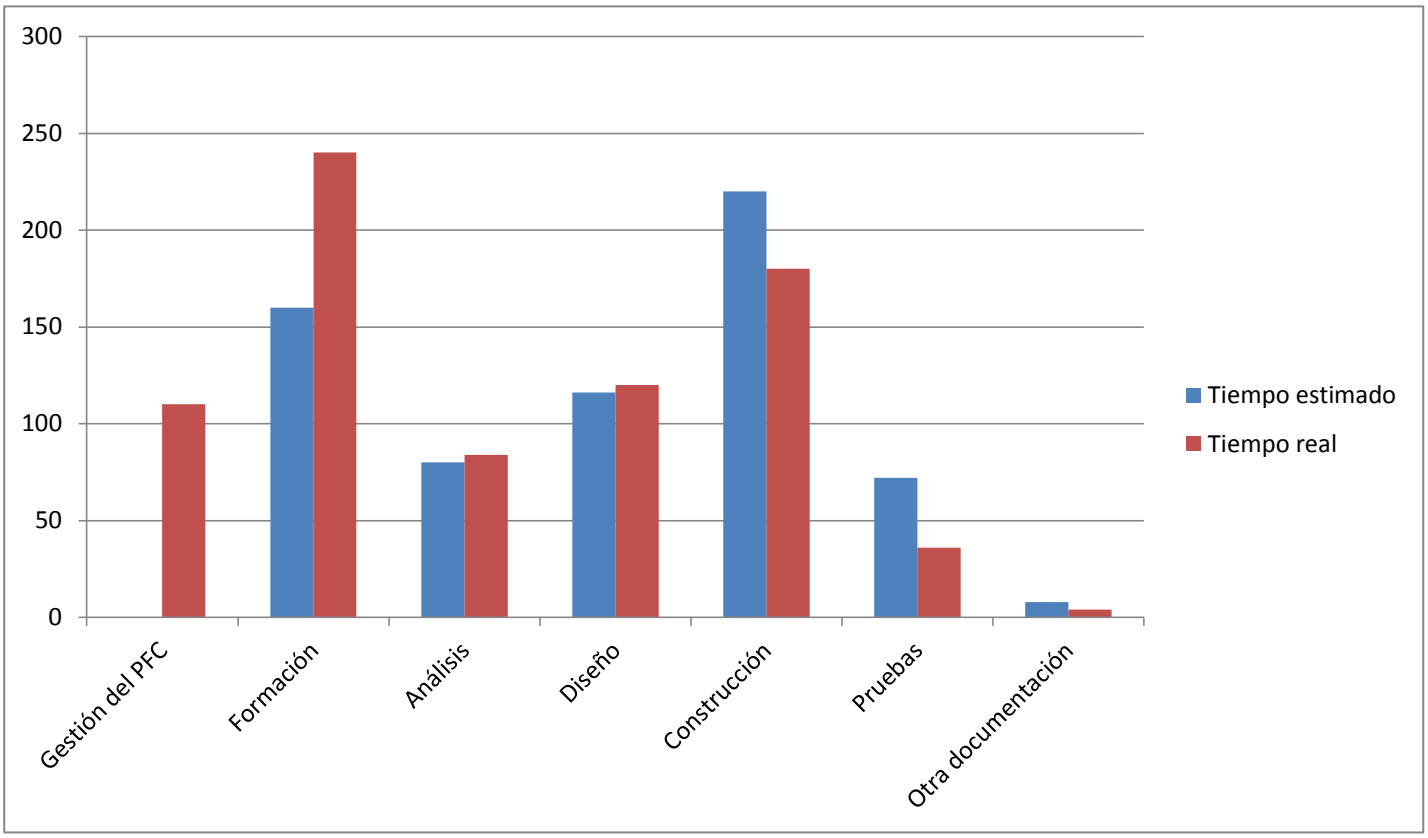


Diagrama 5

4 Análisis del proyecto

El objetivo del análisis del sistema es obtener las especificaciones necesarias para servir de apoyo y definir con exactitud qué tareas va a realizar el nuevo sistema y así poder satisfacer las necesidades de información de la parte del diseño.

Las secciones de este apartado son:

- Especificación de requisitos
- Roles
- Diagramas de casos de uso del sistema
- Identificación de clases provisionales y diagrama
- Diagramas de actividad
- Diagramas de secuencia

4.1 Especificación de requisitos

Como ya ha quedado reflejado con anterioridad, el objetivo básico de este proyecto es proporcionar a los usuarios una información real del estado de su vehículo móvil. Dicha tarea se descompondrá de tres equipos diferentes.

- El primero, el dispositivo instalado en el vehículo y que servirá como emisor de dicha información.
- El segundo, el dispositivo del usuario y en el que se podrá acceder a toda la información enviada por el primero.
- El tercero, un servidor central que se encargará de recoger la información del vehículo y almacenarla para futuras consultas por parte del segundo dispositivo.

4.2 Roles

4.2.1 Usuario

Dicho rol corresponderá al usuario que accederá desde tu terminal a los datos almacenados en el servidor

4.2.2 Vehículo

Dicho rol corresponderá al vehículo cuya función será la de enviar la información al servidor, cada determinado rango de tiempo.

4.3 Diagramas de casos de uso

4.3.1 Vehículo

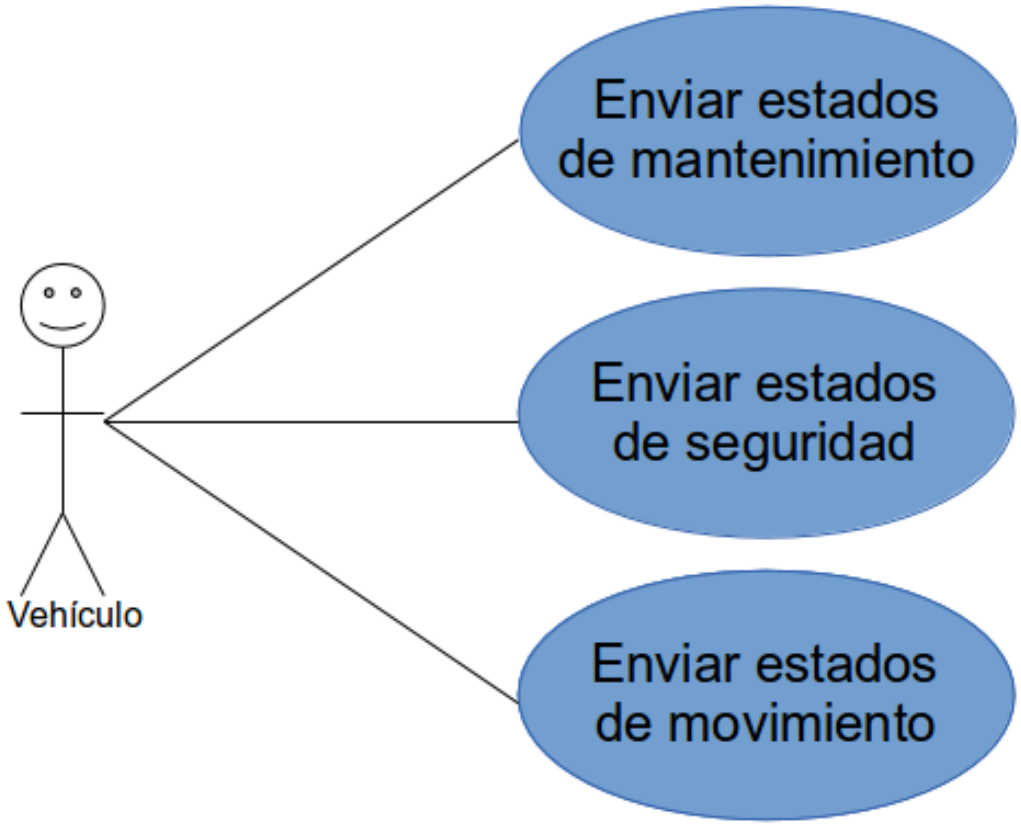


Diagrama 6

El vehículo deberá enviar su estado actual en todo momento para poder ser estos estados, almacenados en el servidor. Se han subdividido dichos estados en tres clases.

- Estados de mantenimiento: Aquellos estados que nos informen de los valores de los diferentes medidores del vehículo.
- Estados de seguridad: Aquellos estados que identifiquen el nivel de seguridad o de compromiso de la misma, del vehículo.
- Estados de movimiento: Aquellos estados que informen de la situación del vehículo y del desplazamiento del mismo durante un periodo determinado.

4.3.1.1 Estados de mantenimiento

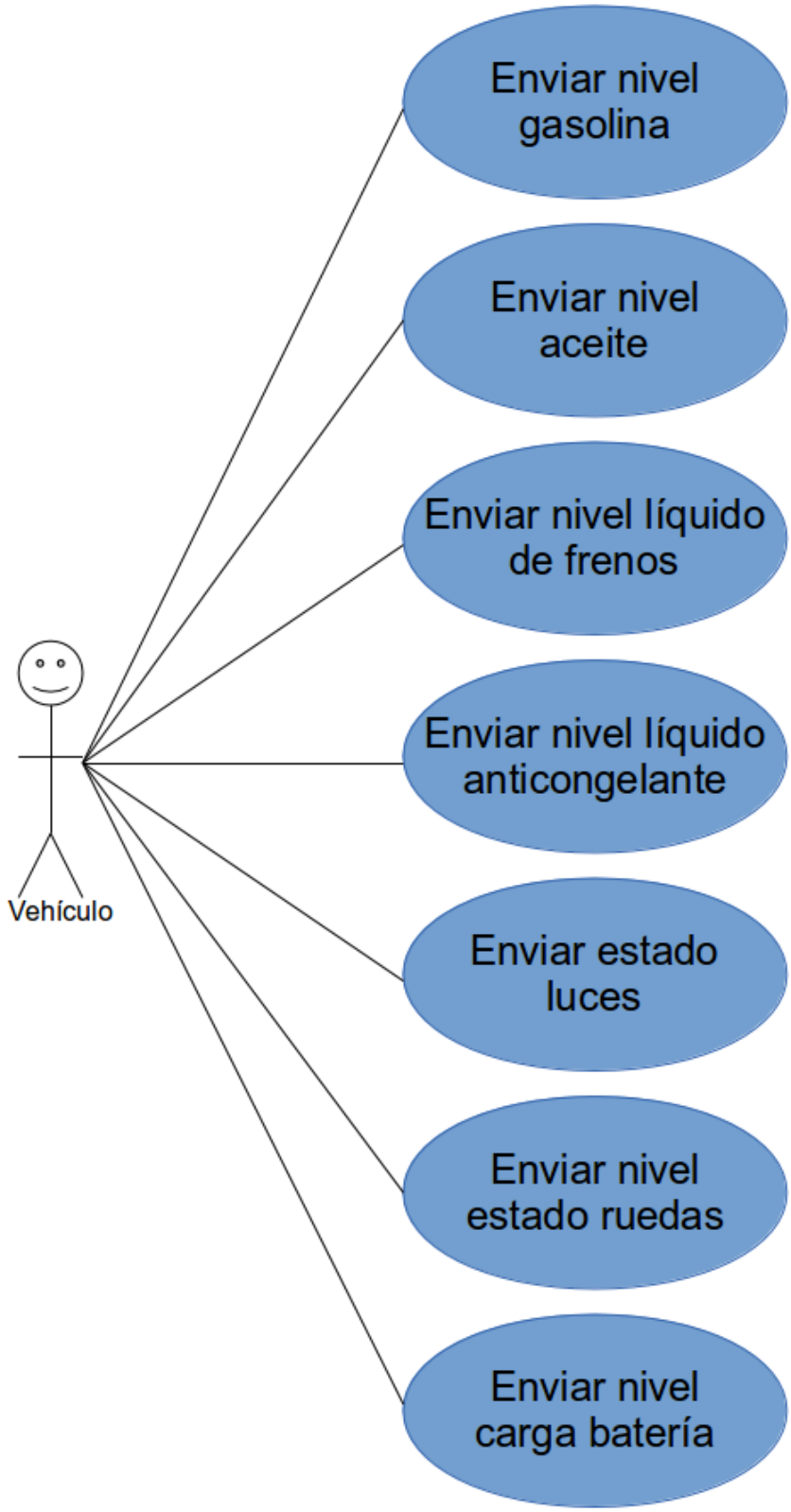


Diagrama 7

- Enviar nivel gasolina: El vehículo envía al servidor el porcentaje combustible disponible en su depósito correspondiente.
- Enviar nivel de aceite: El vehículo envía al servidor el porcentaje aceite disponible en su depósito correspondiente.
- Enviar nivel líquido de frenos: El vehículo envía al servidor el porcentaje líquido de frenos disponible en su depósito correspondiente.
- Enviar estado luces: El vehículo envía al servidor el estado actual de las luces del vehículo (si estas están o no encendidas)
- Enviar nivel estado ruedas: El vehículo envía al servidor la presión de las ruedas.
- Enviar carga batería: El vehículo envía al servidor el porcentaje de carga de la batería.

4.3.1.2 Estados de seguridad

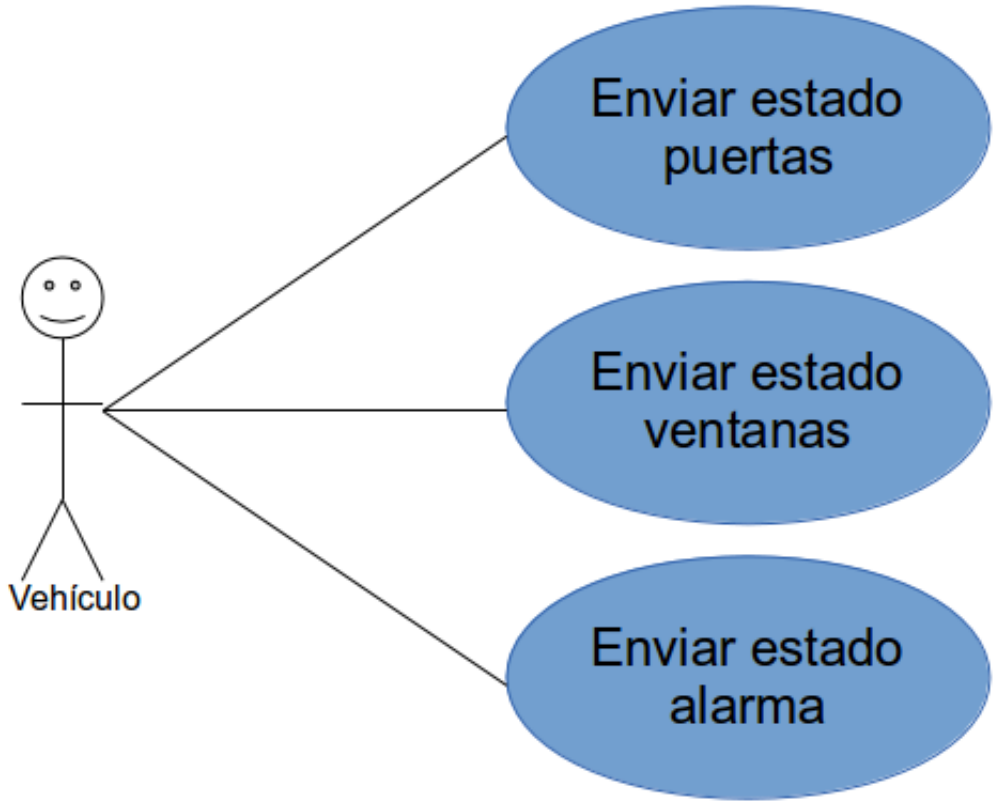


Diagrama 8

- Enviar estado puertas: El vehículo envía al servidor el estado actual de las puertas del mismo (si hay alguna puerta abierta)
- Enviar estado ventanas: El vehículo envía al servidor el estado actual de las ventanas del mismo (si hay alguna ventana abierta)

- Enviar estado alarma: El vehículo envía al servidor información sobre el estado de la alarma (si se encuentra bloqueado y con la alarma activada o no)

4.3.1.3 Estados de movimiento

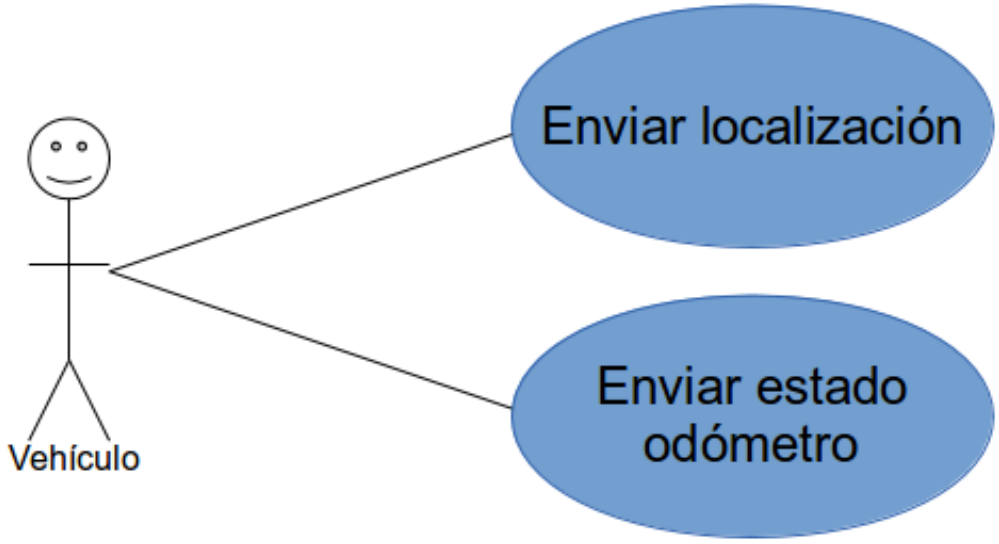


Diagrama 9

- Enviar localización: El vehículo envía al servidor su localización actual (obtenida mediante GPS, A-GPS, triangulación, etc)
- Enviar estado odómetro: El vehículo envía al servidor el valor actual del odómetro.

4.3.2 Usuario

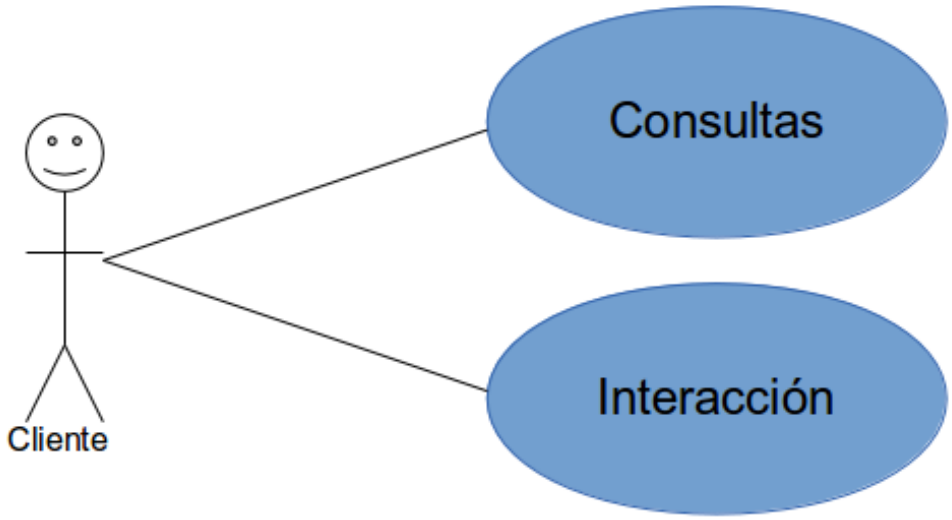


Diagrama 10

Por su parte, el usuario de la aplicación, deberá poder realizar dos tipos de interacciones con la aplicación. Se han subdividido estos a su vez, en dos categorías.

- Consultas: Todo tipo de interacciones que sirvan para que el usuario reciba información del estado del vehículo.
- Acción: Interacciones en las cuales el usuario envíe instrucciones a realizar por el vehículo.

4.3.2.1 Consultas

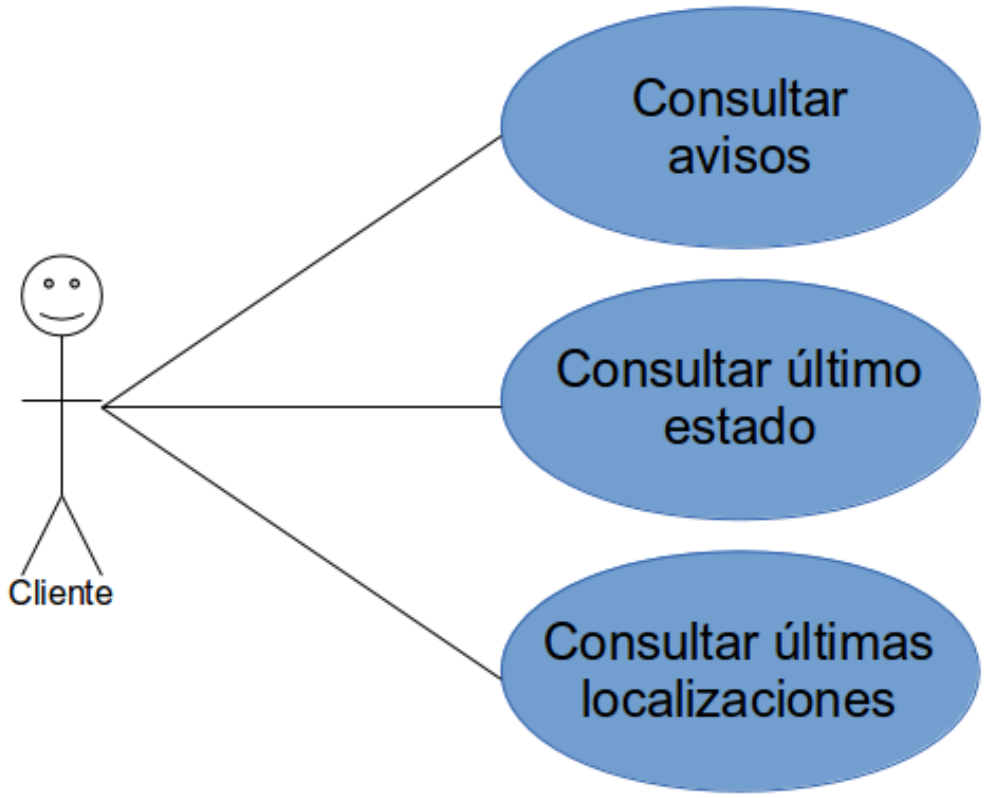


Diagrama 11

- Consultar avisos: El usuario podrá consultar los avisos que el sistema considere de mayor importancia
- Consultar último estado: El usuario podrá consultar todos los detalles del último estado enviado por el vehículo
- Consultar últimas localizaciones: El usuario podrá consultar un histórico de las últimas localizaciones del vehículo.

4.3.2.2 *Acción*

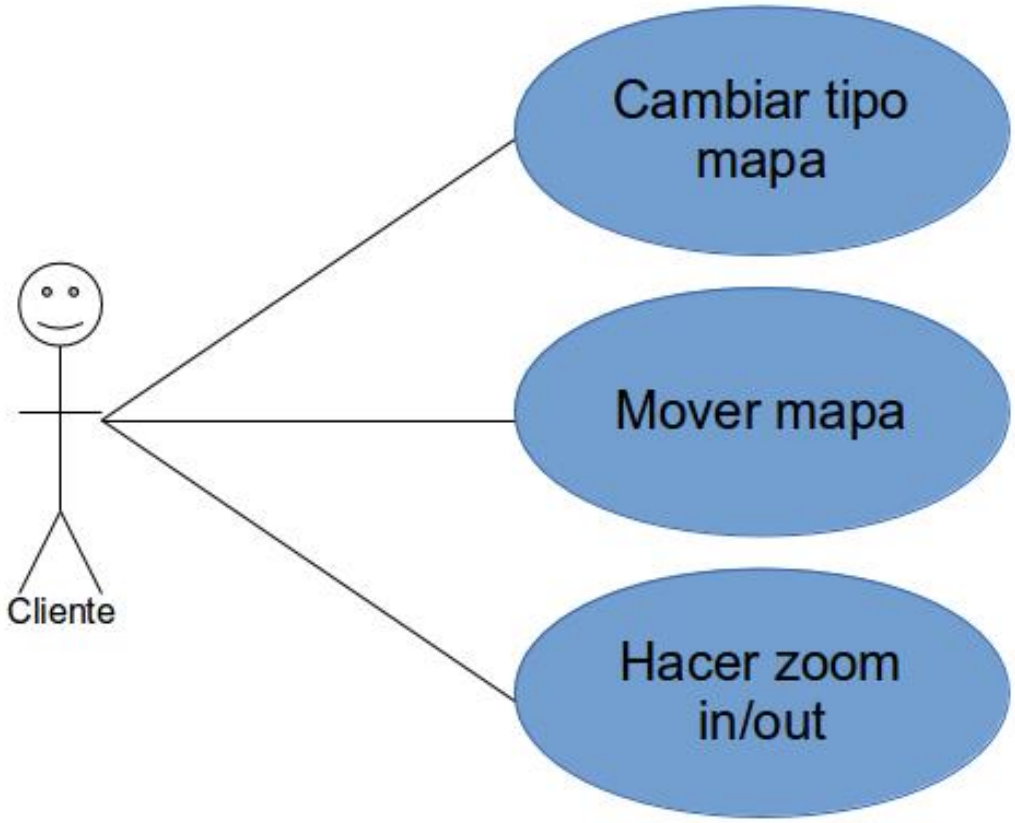


Diagrama 12

- Cambiar tipo de mapa: El usuario podrá en todo momento cambiar el tipo de mapa que desea visualizar (vista de mapa, vista de satélite, etc)
- Mover mapa: El usuario podrá desplazarse en el mapa.
- Hacer zoom in/out: El usuario podrá en todo momento ampliar o reducir el zoom del mapa.

4.3.2.3 Órdenes

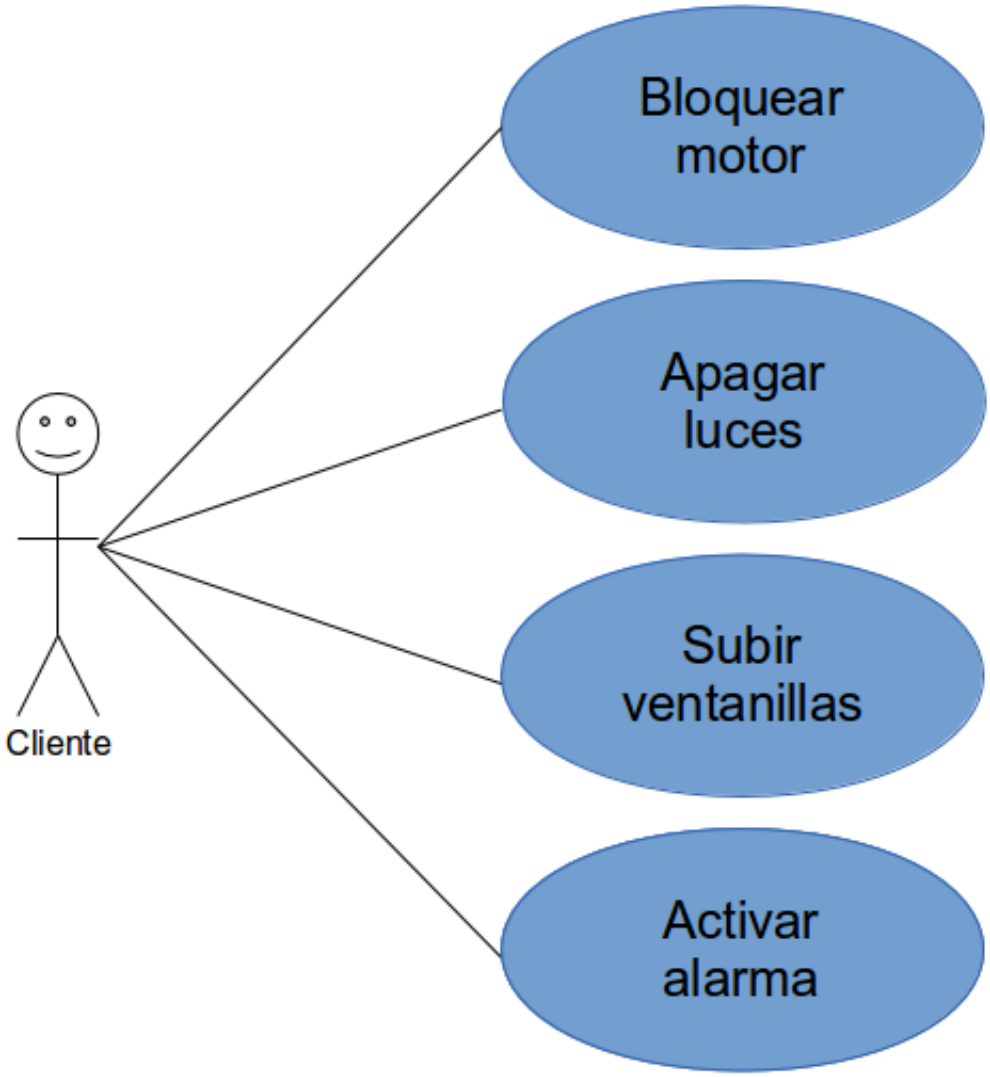


Diagrama 13

- Bloquear motor: El usuario podrá bloquear el motor en caso de que el vehículo haya sido sustraído. Este bloqueo se realizará cuando el coche esté parado.
- Apagar luces: Si el propietario se ha dejado las luces encendidas, podrá apagarlas desde el dispositivo móvil.
- Subir ventanillas: Si el propietario se ha dejado alguna ventana abierta, podrá cerrarla desde el dispositivo móvil.
- Activar alarma: Si al propietario se le ha olvidado activar la alarma, podrá hacerlo desde el dispositivo móvil.

4.3.3 Servidor

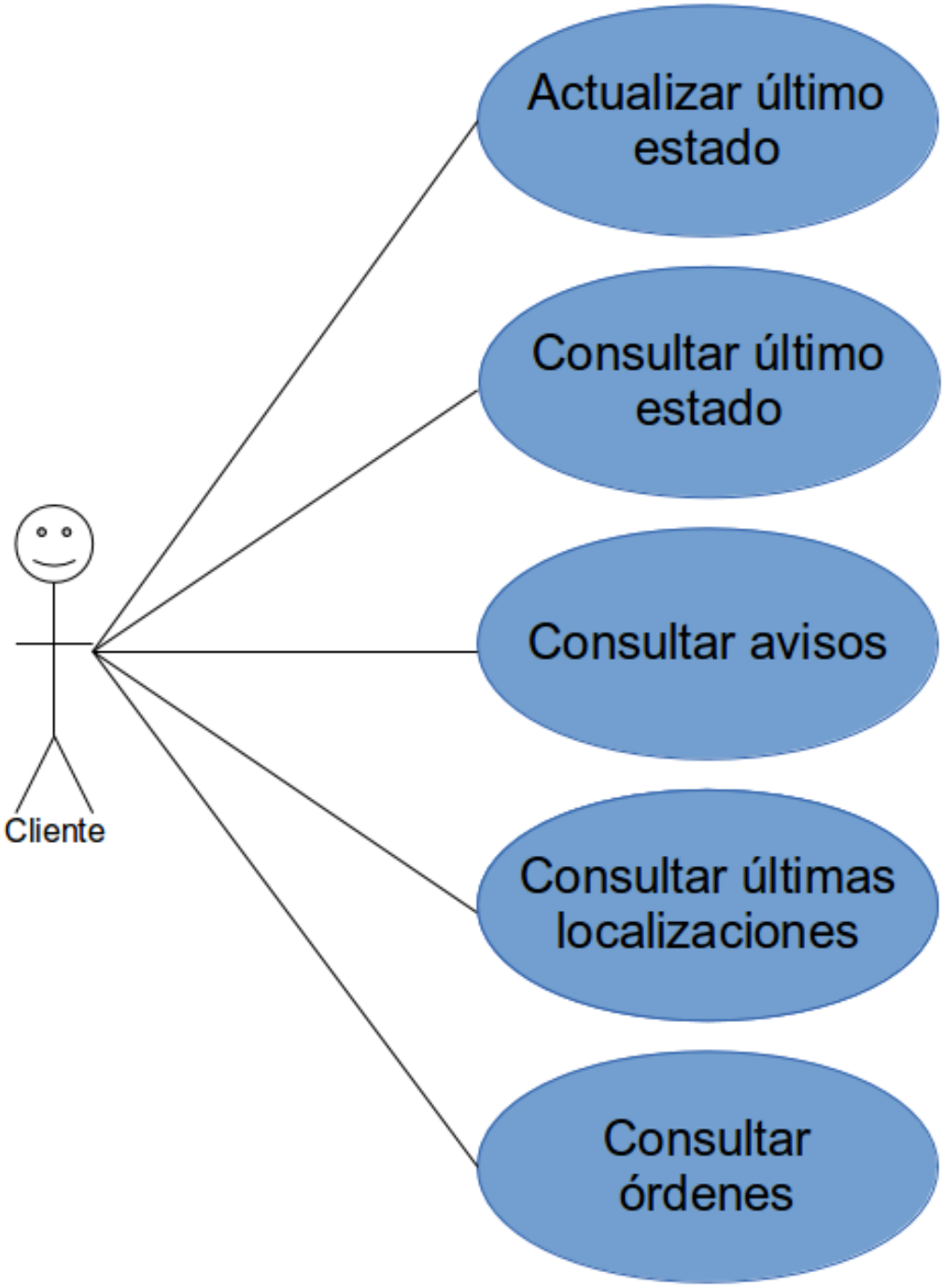


Diagrama 14

- Actualizar último estado: El servidor recibe y almacena la información del último estado, proporcionada por el vehículo

- Consultar último estado: El servidor muestra al usuario, la información correspondiente al último estado guardado
- Consultar avisos: El servidor mostrará al usuario los avisos que el sistema considere de mayor importancia
- Consultar últimas localizaciones: El servidor mostrará un historial de las últimas localizaciones.
- Consultar órdenes: El servidor mostrará las órdenes del dispositivo móvil hacia el vehículo

4.4 Identificación de capas

Teniendo siempre en mente que entre los objetivos que debe cumplir cualquier sistema tecnológico actual, está el de la sencillez, la reutilización y el mantenimiento del mismo, se ha optado por un patrón de capas Modelo-Vista-Controlador (MVC por sus siglas en inglés Model-View-Controller)

El Modelo Vista Controlador es un patrón para el desarrollo del software que se basa en separar los datos (por un lado), la interfaz del usuario (por otro) y la lógica interna (por un último lado). Es mayormente usado en aplicaciones web, dónde la vista es la página HTML, el modelo es el Sistema de Gestión de Base de Datos y la lógica interna, y el controlador es el responsable de recibir los eventos y darles solución.

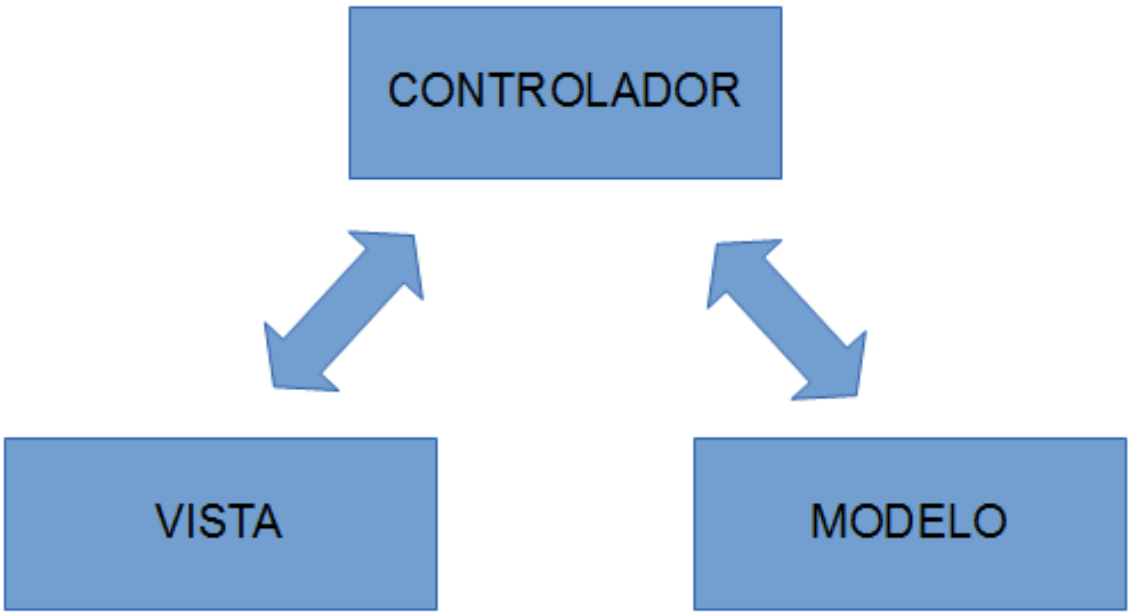


Diagrama 15

4.4.1 La capa Vista

Es la que presenta al modelo en un formato adecuado para que el usuario pueda interactuar con él, por tanto será donde se incluirán todas las clases que conformen la interfaz gráfica que responderá a las acciones el usuario y será la encargada de mostrar los elementos en pantalla.

4.4.2 La capa Controlador

Es el elemento más abstracto. Recibe, trata y responde los eventos enviados por el usuario o por la propia aplicación e interactúa tanto con el modelo como con la vista. De esta forma se puede establecer un canal de comunicación a través del cual las vistas puedan reconocer y responder a los cambios en los modelos.

4.4.3 La capa Modelo

Es la representación de la información en el sistema. Trabaja junto a la vista para mostrar la información al usuario y es accedido por el controlador para añadir, eliminar, consultar o actualizar datos. Dicha capa se encarga de mantener los datos de la aplicación, así como definir la lógica especial que los manipula.

4.5 Diagramas de actividad

4.5.1 Diagrama 1

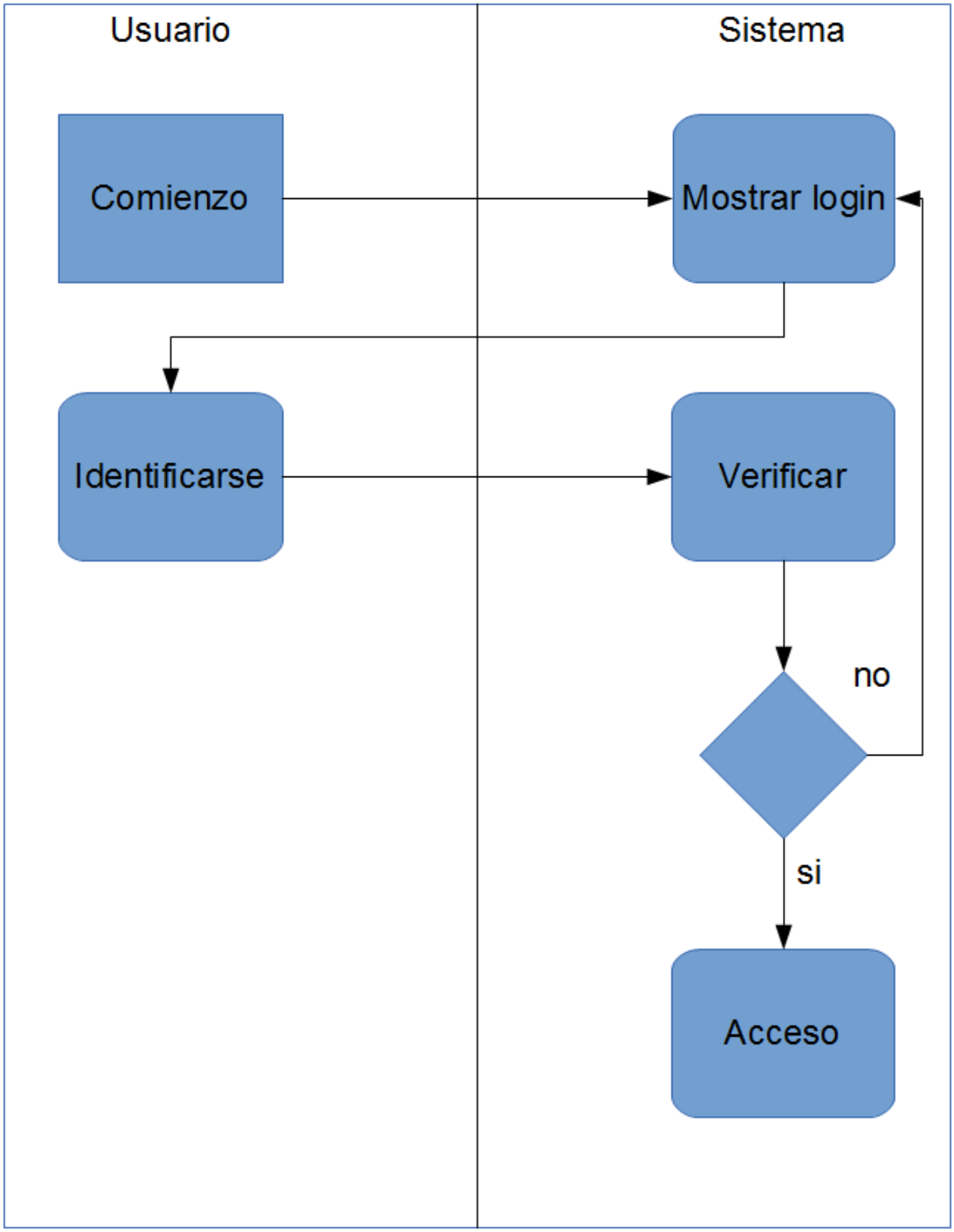


Diagrama 16

El diagrama anterior, representa el proceso que sigue la aplicación cuando un usuario quiere acceder a la misma. Nada más arrancar se mostrará una pantalla de acceso de

usuarios (pantalla de login) en la que se pedirá que el mismo se identifique (se seguirá para ello el método user-password). Una vez el usuario se ha identificado el sistema comprobará que es un usuario válido o no.

En caso de no serlo, se volverá a mostrar la pantalla de login mostrando además el motivo por el cual este usuario ha sido rechazado.

Por el contrario, en caso de que el usuario sea un usuario válido, se le dará acceso a la aplicación.

4.5.2 Diagrama 2

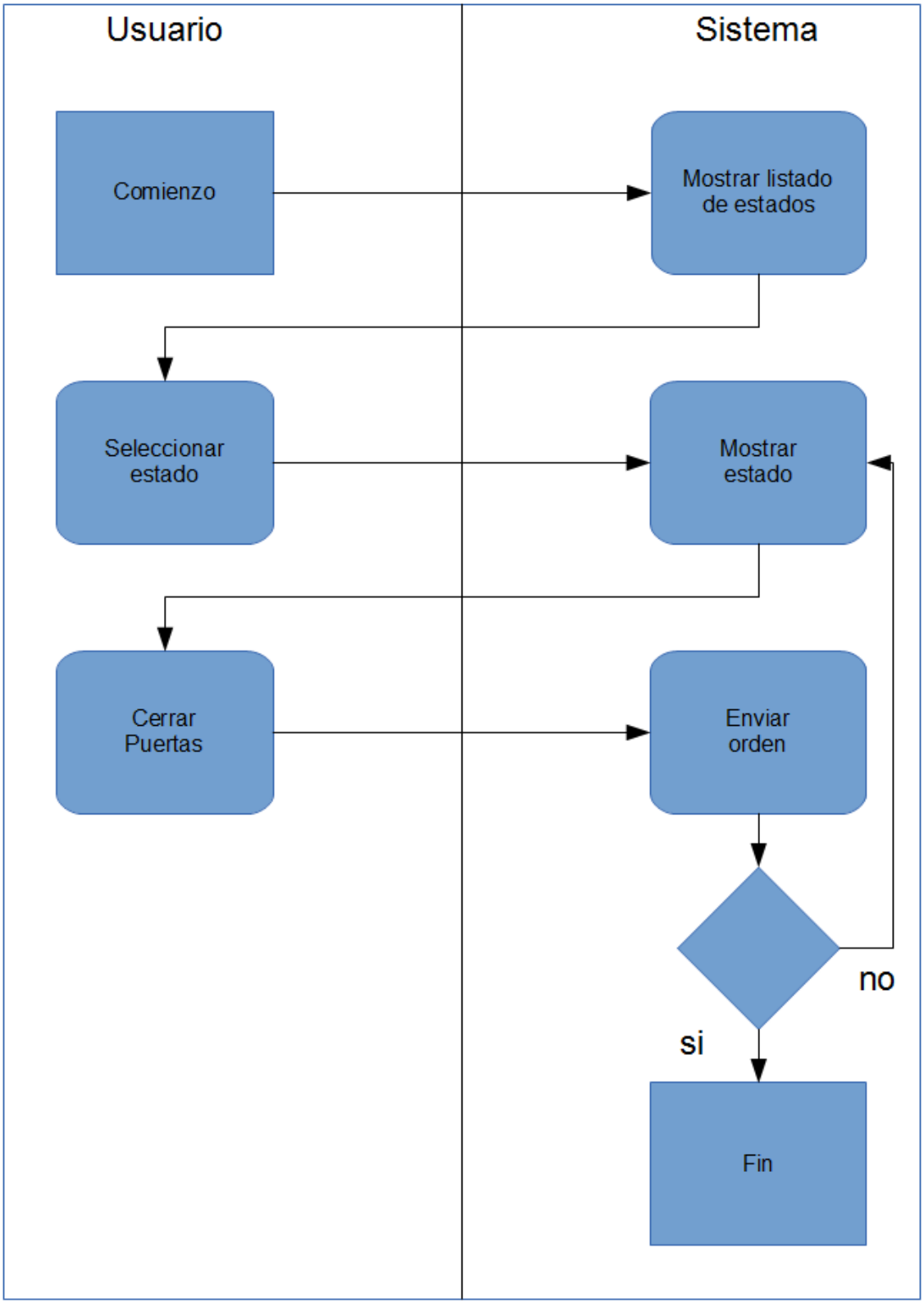


Diagrama 17

Una de las funciones de la aplicación es el envío de instrucciones desde el dispositivo del cliente al vehículo asociado, como por ejemplo la instrucción “Cerrar puertas”.

Al acceder el sistema nos mostrará una relación de coches que el propietario gestiona. El usuario tendrá que elegir el vehículo al cual le vamos a enviar dicha instrucción. Una vez seleccionado, el sistema nos mostrará información relativa como es el estado del mismo y una serie de posibles acciones a realizar. El usuario selecciona la opción correspondiente (cerrar puertas) y la envía al servidor.

Si la acción se ejecuta con normalidad la actividad finaliza. En caso de haber algún error, la aplicación volvería a mostrar al usuario el estado del mismo, en el que se podría comprobar como dicha acción no ha sido ejecutada. Se valorará también informarle activamente de dicho fallo con un mensaje.

El sistema es capaz de realizar muchas más funcionalidades, pero todas siguen el mismo esquema que esta última por tanto con este se dan por abarcados todos los tipos diferentes de diagramas de actividad.

Como se puede comprobar, la aplicación tiene un funcionamiento bastante simple que a su vez es una de las características fundamentales de las aplicaciones móviles, la sencillez de uso.

4.6 Diagramas de secuencia

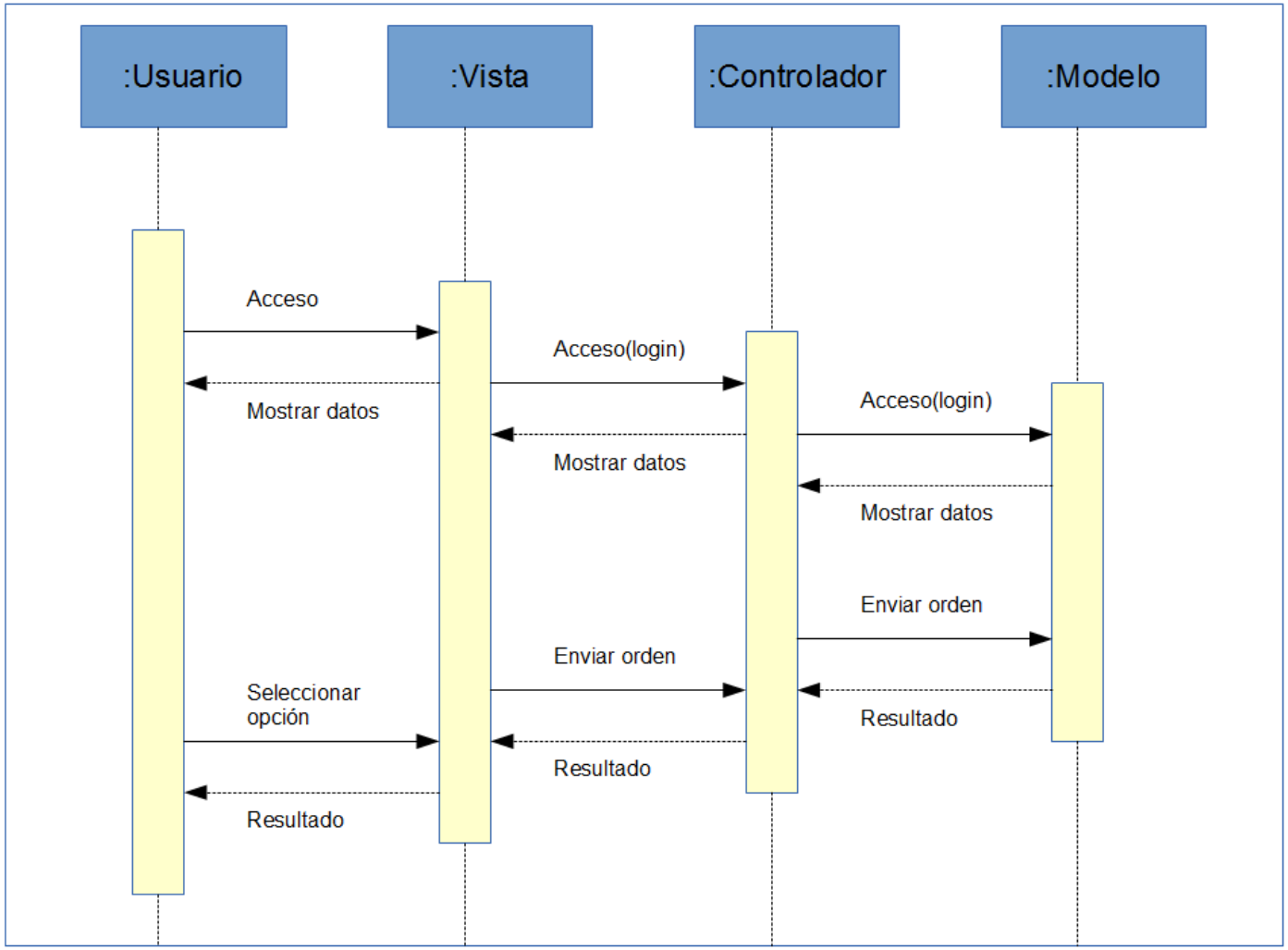


Diagrama 18

Cuando el usuario accede a la aplicación, la vista envía las credenciales al controlador. Este a su vez envía esas mismas credenciales al modelo que devuelve los datos a mostrar correspondientes a dicho usuario al controlador. El controlador por su parte devuelve estos datos a la vista que a su vez se lo muestra al usuario.

Posteriormente el usuario selecciona una opción de la pantalla, por lo que la vista envía una orden al controlador. Este a su vez envía dicha orden al modelo que devuelve el resultado de dicha orden a través del controlador y la vista. Esta última es la que se encarga de mostrarle de nuevo al usuario el nuevo estado.

Al igual que sucedía con el último diagrama de actividad, este último diagrama de secuencia es extrapolable a todos los posibles casos del sistema ya que el mecanismo de comunicación es el mismo en todos ellos, por tanto con este diagrama se dan por concluidos todos los posibles diagramas de secuencia.

4.7 Presupuesto

A continuación detallamos el presupuesto de este proyecto.

Para la realización de dicho proyecto, se han utilizado herramientas de software libr, por lo que no se incluye ningún gasto de licencias de uso. Por tanto, para la realización de dicho presupuesto, se tendrán en cuenta exclusivamente, las horas de realización del mismo.

Se aplicará a su vez un 6% de beneficio.

4.7.1 Listado de mano de obra

Código	Título	Precio
D101	Director	50
A101	Analista	45
D201	Diseñador	40
D301	Desarrollador	30
T101	Técnico	20

Tabla 4

4.7.2 Presupuesto y mediciones

Código	Título	Cantidad	Precio	Importe
Gestión del PFC				
D101	Director	40	50	2.000€
A101	Analista	35	45	1.575€
D201	Diseñador	20	40	800€
D301	Desarrollador	10	30	300€
T101	Técnico	5	20	100€
Total Gestión		110		**4.775€**
Análisis				
D101	Director	10	50	500€
A101	Analista	74	45	3.330€
Total Análisis		84		**3.830€**
Diseño				
D201	Diseñador	85	40	3.400€
D301	Desarrollador	35	30	1.050€
Total Diseño		120		**4.450€**
Construcción				
D301	Desarrollador	110	30	3.300€
T101	Técnico	70	20	1.400€
Total Construcción		180		**3.700€**
Pruebas				
A101	Analista	16	45	720€
D301	Desarrollador	20	30	600€
Total Pruebas		36		**1.320€**
Otra documentación				

D101	Director	1	50	50€
A101	Analista	3	45	135€
Total Otra documentación		4		**185€**
Implantación				
T101	Técnico	9	20	180€
A101	Analista	5	45	225€
Total Implantación		14		**405€**
TOTAL		548		**18.665€**

Tabla 5

4.7.3 Resumen del presupuesto

Apartado	Horas	Importe
Gestión del PFC	110	6.175€
Análisis	84	4.720€
Diseño	120	6.075€
Construcción	180	6.500€
Pruebas	36	1.680€
Otra documentación	4	230€
Implantación	14	545€

Tabla 6

TOTAL EJECUCIÓN MATERIAL - 18.665€

13,00% Gastos generales ---------- 3.370€

6,00% Beneficio industrial ---------- 1.120€

Suma de G.G. y B.I. 4.490€

21,00% I.V.A. ---------- 4.863€

TOTAL PRESUPUESTO 27.991€

Asciende el presupuesto general a la expresada cantidad de TREINTA Y SIETE MIL TRESCIENTOS VEINTINUEVE EUROS

5 Diseño del proyecto

El diseño en ingeniería es el proceso de idear un sistema, componente o proceso para satisfacer ciertas necesidades. Es un proceso de toma de decisiones (a menudo iterativo) en el que las ciencias básicas y las ciencias de la ingeniería se aplican para convertir recursos en forma óptima a fin de cumplir un objetivo estipulado. Entre los elementos fundamentales del proceso de diseño se encuentran el establecimiento de objetivos y criterios, síntesis, análisis construcción ensayos y evaluación.

Por tanto en esta etapa es donde se le dará forma a la información que se recolectó y analizó en apartados anteriores, conformando el nuevo sistema, en documentos, diagramas de flujo de procesos, diseños de entradas y salidas, selección de dispositivos de almacenamiento, la lógica que llevará el sistema, diseño de archivos maestros, de trabajo, flujo de los datos determinar volúmenes de información, pantallas, menús, submenús, mensajes, reportes, mantenimiento, ajustes...

Esta sección se compone de los siguientes apartados

- Arquitectura del Sistema
- Patrón de diseño
- Diseño de Clases
- Diseño de la Base de Datos

5.1 Arquitectura del sistema

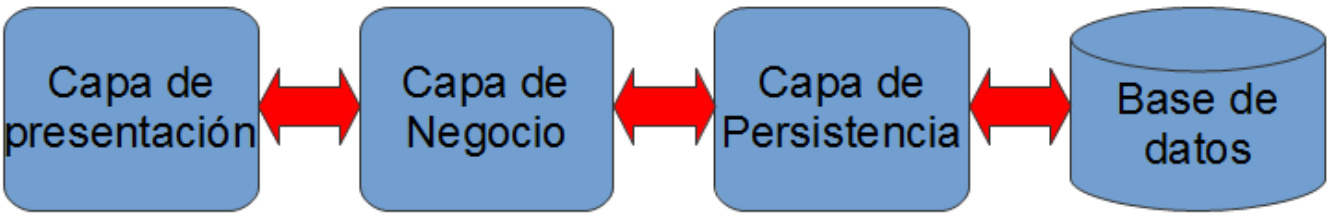


Diagrama 19

5.1.1 Capa de presentación: es la que ve el usuario (también se la denomina "capa de usuario"), presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). También es conocida como interfaz gráfica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario. Esta capa se comunica únicamente con la capa de negocio. Dicha capa estará compuesta por un conjunto de archivos en formato XML (el formato estándar para el diseño de interfaces en Android)

5.1.2 Capa de negocio: es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de persistencia, para solicitar al gestor de base de datos almacenar o recuperar datos de él.

5.1.3 Capa de persistencia: es la capa encargada de proporcionar una biblioteca de funcionalidades para el acceso a la base de datos, lugar en el que residen los datos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

5.2 Patrón de diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Un patrón de diseño resulta ser una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reutilizable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

En el apartado de Análisis se mencionó que se iba utilizar un patrón de diseño Modelo-vista-controlador por ser el más extendido en la actualidad y por su excepcional desempeño en todo tipo de proyectos. En realidad esto no es totalmente cierto puesto que muchos consideran que el patrón que mejor se adapta al desarrollo de aplicaciones para Android es el Modelo-Vista-Presentador (Model-View-Presenter) ya que la interfaz que haría de Controlador afecta en algunos momentos a la vista. Aún así, exceptuando estos casos, el diseño se adaptaría a un MVC por tanto vamos a asumir que este es el patrón escogido.

Las ventajas de usar dicho patrón son:

- Aislamiento por la separación de las capas.
- Sencillez para crear distintos tipos de datos o usar distintos protocolos.
- Posibilidad de utilización de distintas capas de persistencia.
- Reutilización de componentes.
- Simplicidad de mantenimiento.

5.3 Diseño de clases

Una vez definida la arquitectura que va a tener el sistema y realizado el análisis del mismo, procedemos a la realización del diagrama que servirá como base para la etapa posterior de "construcción". Como es lógico, durante esta etapa aún no ha comenzado el desarrollo, por tanto es muy probable que en etapas posteriores el siguiente diagrama sufra modificaciones tales como la creación o desaparición de clases o métodos auxiliares. Sin embargo las clases principales de la aplicación que se especifican a continuación sí que son las definitivas.

5.3.1 Diagramas

A continuación se detallan los diagramas de clases, tanto de la aplicación del usuario, como de la aplicación del vehículo.

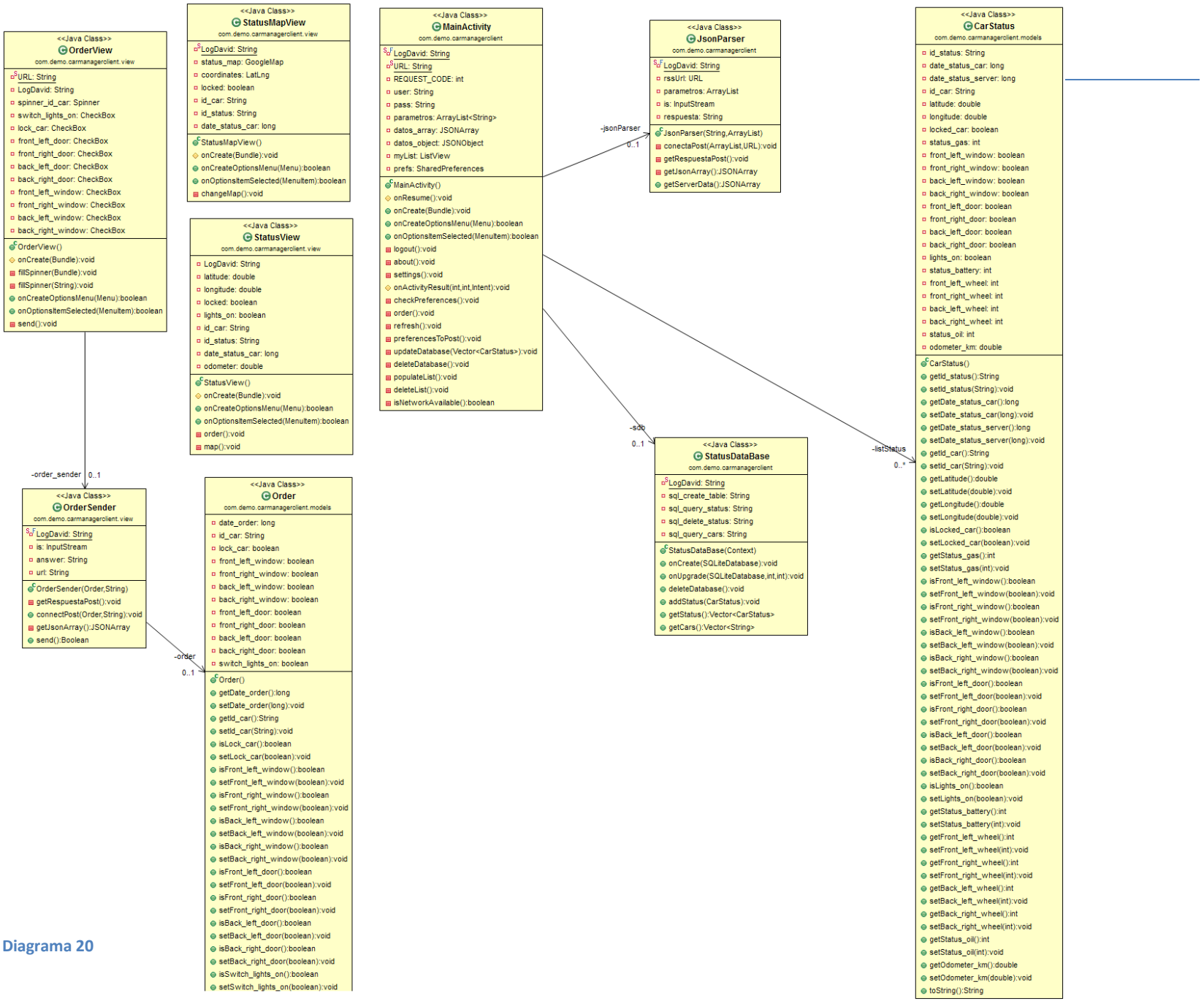

<<Java Class>>
OrderView
com.demo.carmanagerclient.view
URL: String
LogDavid: String
spinner_id_car: Spinner
switch_lights_on: CheckBox
lock_car: CheckBox
front_left_door: CheckBox
front_right_door: CheckBox
back_left_door: CheckBox
back_right_door: CheckBox
front_left_window: CheckBox
front_right_window: CheckBox
back_left_window: CheckBox
back_right_window: CheckBox
OrderView()
onCreate(Bundle):void
fillSpinner(Bundle):void
fillSpinner(String):void
onCreateOptionsMenu(Menu):boolean
onOptionsItemSelected(MenuItem):boolean
send():void
<<Java Class>>
StatusMapView
com.demo.carmanagerclient.view
LogDavid: String
status_map: GoogleMap
coordinates: LatLng
locked: boolean
id_car: String
id_status: String
date_status_car: long
StatusMapView()
onCreate(Bundle):void
onCreateOptionsMenu(Menu):boolean
onOptionsItemSelected(MenuItem):boolean
changeMap():void
<<Java Class>>
StatusView
com.demo.carmanagerclient.view
LogDavid: String
latitude: double
longitude: double
locked: boolean
lights_on: boolean
id_car: String
id_status: String
date_status_car: long
odometer: double
StatusView()
onCreate(Bundle):void
onCreateOptionsMenu(Menu):boolean
onOptionsItemSelected(MenuItem):boolean
order():void
map():void
-order_sender
0..1
<<Java Class>>
OrderSender
com.demo.carmanagerclient.view
LogDavid: String
is: InputStream
answer: String
url: String
OrderSender(Order,String)
getRespuestaPost():void
connectPost(Order,String):void
getJsonArray():JSONArray
send():Boolean
-order
0..1
<<Java Class>>
Order
com.demo.carmanagerclient.models
date_order: long
id_car: String
lock_car: boolean
front_left_window: boolean
front_right_window: boolean
back_left_window: boolean
back_right_window: boolean
front_left_door: boolean
front_right_door: boolean
back_left_door: boolean
back_right_door: boolean
switch_lights_on: boolean
Order()
getDate_order():long
setDate_order(long):void
getId_car():String
setId_car(String):void
isLock_car():boolean
setLock_car(boolean):void
isFront_left_window():boolean
setFront_left_window(boolean):void
isFront_right_window():boolean
setFront_right_window(boolean):void
isBack_left_window():boolean
setBack_left_window(boolean):void
isBack_right_window():boolean
setBack_right_window(boolean):void
isFront_left_door():boolean
setFront_left_door(boolean):void
isFront_right_door():boolean
setFront_right_door(boolean):void
isBack_left_door():boolean
setBack_left_door(boolean):void
isBack_right_door():boolean
setBack_right_door(boolean):void
isSwitch_lights_on():boolean
setSwitch_lights_on(boolean):void
<<Java Class>>
MainActivity
com.demo.carmanagerclient
LogDavid: String
URL: String
REQUEST_CODE: int
user: String
pass: String
parametros: ArrayList<String>
datos_array: JSONArray
datos_object: JSONObject
myList: ListView
prefs: SharedPreferences
MainActivity()
onResume():void
onCreate(Bundle):void
onCreateOptionsMenu(Menu):boolean
onOptionsItemSelected(MenuItem):boolean
logout():void
about():void
settings():void
onActivityResult(int,int,Intent):void
checkPreferences():void
order():void
refresh():void
preferencesToPost():void
updateDatabase(Vector<CarStatus>):void
deleteDatabase():void
populateList():void
deleteList():void
isNetworkAvailable():boolean
-jsonParser
0..1
<<Java Class>>
JsonParser
com.demo.carmanagerclient
LogDavid: String
rssUrl: URL
parametros: ArrayList
is: InputStream
respuesta: String
JsonParser(String,ArrayList)
conectaPost(ArrayList,URL):void
getRespuestaPost():void
getJsonArray():JSONArray
getServerData():JSONArray
-sdb
0..1
<<Java Class>>
StatusDataBase
com.demo.carmanagerclient
LogDavid: String
sql_create_table: String
sql_query_status: String
sql_delete_status: String
sql_query_cars: String
StatusDataBase(Context)
onCreate(SQLiteDatabase):void
onUpgrade(SQLiteDatabase,int,int):void
deleteDatabase():void
addStatus(CarStatus):void
getStatus():Vector<CarStatus>
getCars():Vector<String>
-listStatus
0..*
<<Java Class>>
CarStatus
com.demo.carmanagerclient.models
id_status: String
date_status_car: long
date_status_server: long
id_car: String
latitude: double
longitude: double
locked_car: boolean
status_gas: int
front_left_window: boolean
front_right_window: boolean
back_left_window: boolean
back_right_window: boolean
front_left_door: boolean
front_right_door: boolean
back_left_door: boolean
back_right_door: boolean
lights_on: boolean
status_battery: int
front_left_wheel: int
front_right_wheel: int
back_left_wheel: int
back_right_wheel: int
status_oil: int
odometer_km: double
CarStatus()
getId_status():String
setId_status(String):void
getDate_status_car():long
setDate_status_car(long):void
getDate_status_server():long
setDate_status_server(long):void
getId_car():String
setId_car(String):void
getLatitude():double
setLatitude(double):void
getLongitude():double
setLongitude(double):void
isLocked_car():boolean
setLocked_car(boolean):void
getStatus_gas():int
setStatus_gas(int):void
isFront_left_window():boolean
setFront_left_window(boolean):void
isFront_right_window():boolean
setFront_right_window(boolean):void
isBack_left_window():boolean
setBack_left_window(boolean):void
isBack_right_window():boolean
setBack_right_window(boolean):void
isFront_left_door():boolean
setFront_left_door(boolean):void
isFront_right_door():boolean
setFront_right_door(boolean):void
isBack_left_door():boolean
setBack_left_door(boolean):void
isBack_right_door():boolean
setBack_right_door(boolean):void
isLights_on():boolean
setLights_on(boolean):void
getStatus_battery():int
setStatus_battery(int):void
getFront_left_wheel():int
setFront_left_wheel(int):void
getFront_right_wheel():int
setFront_right_wheel(int):void
getBack_left_wheel():int
setBack_left_wheel(int):void
getBack_right_wheel():int
setBack_right_wheel(int):void
getStatus_oil():int
setStatus_oil(int):void
getOdometer_km():double
setOdometer_km(double):void
toString():String

Diagrama 20

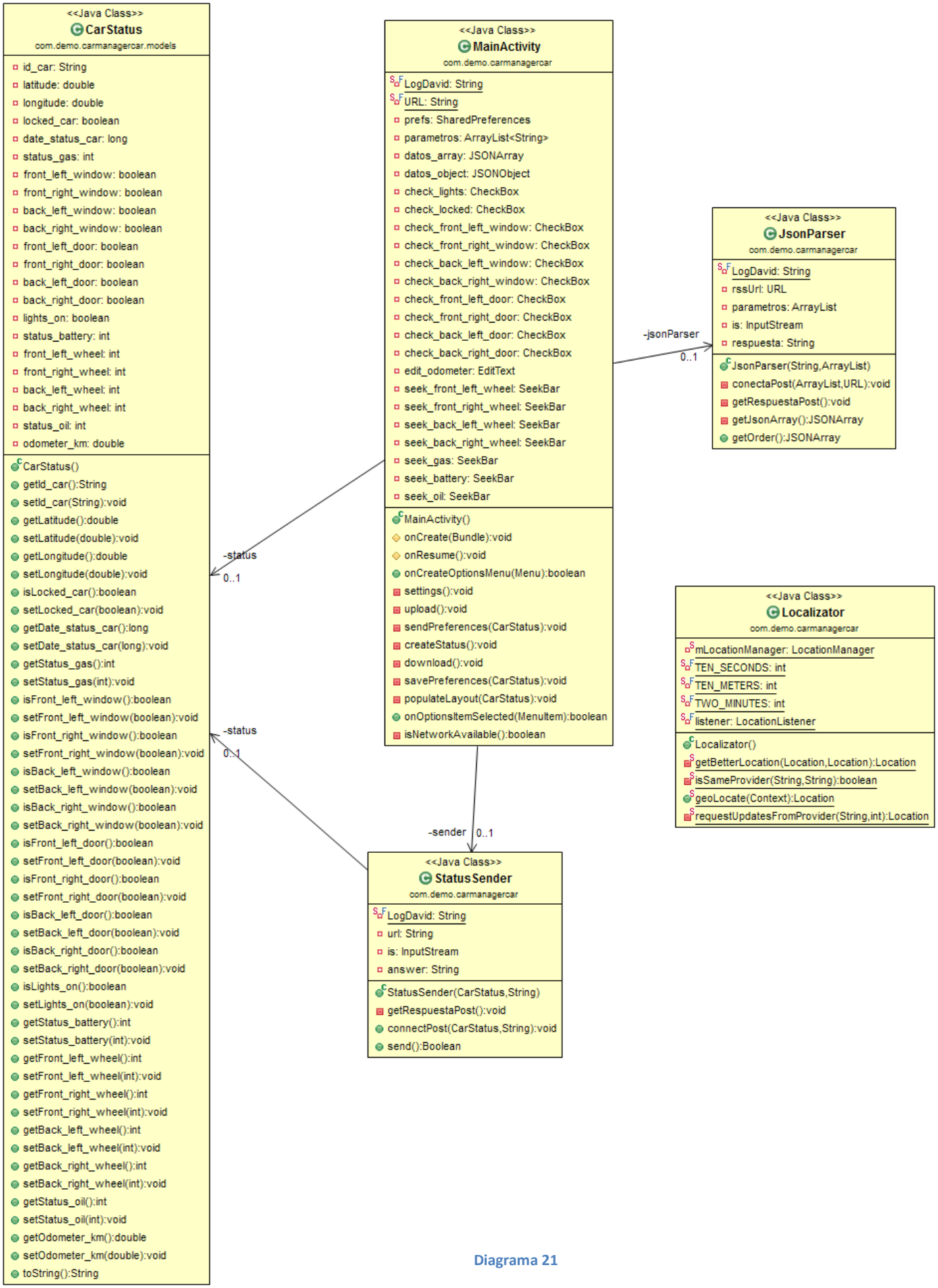

<<Java Class>>
CarStatus
com.demo.carmanagercar.models
id_car: String
latitude: double
longitude: double
locked_car: boolean
date_status_car: long
status_gas: int
front_left_window: boolean
front_right_window: boolean
back_left_window: boolean
back_right_window: boolean
front_left_door: boolean
front_right_door: boolean
back_left_door: boolean
back_right_door: boolean
lights_on: boolean
status_battery: int
front_left_wheel: int
front_right_wheel: int
back_left_wheel: int
back_right_wheel: int
status_oil: int
odometer_km: double
CarStatus()
getId_car():String
setId_car(String):void
getLatitude():double
setLatitude(double):void
getLongitude():double
setLongitude(double):void
isLocked_car():boolean
setLocked_car(boolean):void
getDate_status_car():long
setDate_status_car(long):void
getStatus_gas():int
setStatus_gas(int):void
isFront_left_window():boolean
setFront_left_window(boolean):void
isFront_right_window():boolean
setFront_right_window(boolean):void
isBack_left_window():boolean
setBack_left_window(boolean):void
isBack_right_window():boolean
setBack_right_window(boolean):void
isFront_left_door():boolean
setFront_left_door(boolean):void
isFront_right_door():boolean
setFront_right_door(boolean):void
isBack_left_door():boolean
setBack_left_door(boolean):void
isBack_right_door():boolean
setBack_right_door(boolean):void
isLights_on():boolean
setLights_on(boolean):void
getStatus_battery():int
setStatus_battery(int):void
getFront_left_wheel():int
setFront_left_wheel(int):void
getFront_right_wheel():int
setFront_right_wheel(int):void
getBack_left_wheel():int
setBack_left_wheel(int):void
getBack_right_wheel():int
setBack_right_wheel(int):void
getStatus_oil():int
setStatus_oil(int):void
getOdometer_km():double
setOdometer_km(double):void
toString():String
<<Java Class>>
MainActivity
com.demo.carmanagercar
LogDavid: String
URL: String
prefs: SharedPreferences
parametros: ArrayList<String>
datos_array: JSONArray
datos_object: JSONObject
check_lights: CheckBox
check_locked: CheckBox
check_front_left_window: CheckBox
check_front_right_window: CheckBox
check_back_left_window: CheckBox
check_back_right_window: CheckBox
check_front_left_door: CheckBox
check_front_right_door: CheckBox
check_back_left_door: CheckBox
check_back_right_door: CheckBox
edit_odometer: EditText
seek_front_left_wheel: SeekBar
seek_front_right_wheel: SeekBar
seek_back_left_wheel: SeekBar
seek_back_right_wheel: SeekBar
seek_gas: SeekBar
seek_battery: SeekBar
seek_oil: SeekBar
MainActivity()
onCreate(Bundle):void
onResume():void
onCreateOptionsMenu(Menu):boolean
settings():void
upload():void
sendPreferences(CarStatus):void
createStatus():void
download():void
savePreferences(CarStatus):void
populateLayout(CarStatus):void
onOptionsItemSelected(MenuItem):boolean
isNetworkAvailable():boolean
<<Java Class>>
JsonParser
com.demo.carmanagercar
LogDavid: String
rssUrl: URL
parametros: ArrayList
is: InputStream
respuesta: String
JsonParser(String,ArrayList)
conectaPost(ArrayList,URL):void
getRespuestaPost():void
getJsonArray():JSONArray
getOrder():JSONArray
-jsonParser
0..1
-status
0..1
-status
0..1
-sender
0..1
<<Java Class>>
Localizator
com.demo.carmanagercar
mLocationManager: LocationManager
TEN_SECONDS: int
TEN_METERS: int
TWO_MINUTES: int
listener: LocationListener
Localizator()
getBetterLocation(Location,Location):Location
isSameProvider(String,String):boolean
geoLocate(Context):Location
requestUpdatesFromProvider(String,int):Location
<<Java Class>>
StatusSender
com.demo.carmanagercar
LogDavid: String
url: String
is: InputStream
answer: String
StatusSender(CarStatus,String)
getRespuestaPost():void
connectPost(CarStatus,String):void
send():Boolean

Diagrama 21

5.3.2 Descripción

5.3.2.1 Usuario

Diagrama 22

- CarStatus:

Esta clase será la encargada de almacenar el estado del coche, en un momento determinado. Dicho estado está compuesto de los valores esenciales que son, el id del vehículo, la hora en el que el mismo se envía al servidor, el instante en el que este último lo recibe, la localización exacta del vehículo, y una serie de parámetros fundamentales para conocer su estado real como son, el nivel de carga del depósito, de aceite, de batería, si este se encuentra abierto o cerrado, si tiene alguna ventana o puerta abierta, etc.

Al realizar la llamada al servidor, este nos devolverá un array de estados, correspondientes a todos los estados de todos los coches del usuario.

- Order:

Esta clase será la encargada de gestionar las órdenes que el usuario de dicho vehículo envía al mismo. La esencia es la misma que en la clase CarStatus, solo que con menos valores. Solo aquellos que puedan ser modificados a distancia.

Por ejemplo podremos ordenar al coche a apagar las luces o a cerrar una ventanilla, pero no podremos ordenarle llenar el depósito de combustible.

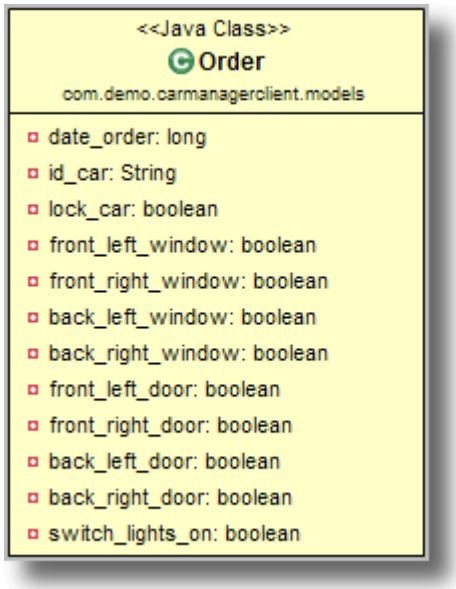


Diagrama 23

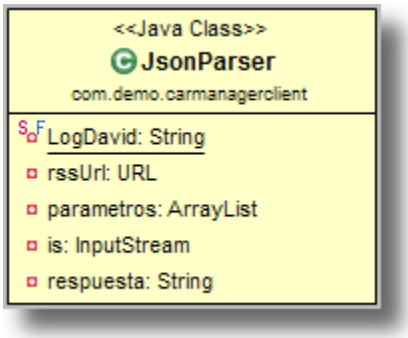


Diagrama 24

- JsonParser:

Esta clase es la encargada de formatear el fichero json recibido desde el servidor y a devolver una respuesta en formato String al módulo encargado de gestionarlo.

Dicho fichero json contendrá el array de CarStatus explicado en el apartado anterior.

- OrderSender:

Al igual que la clase anterior, este se encargará de formatear la respuesta obtenida del servidor, en formato json, después del envío de una orden. Como está explicado en el apartado anterior, esta orden es un objeto de la clase Order

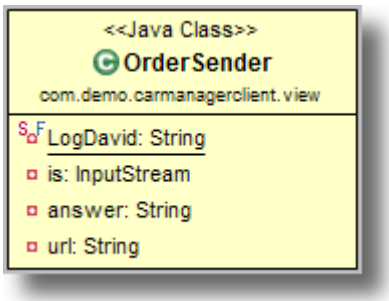


Diagrama 25

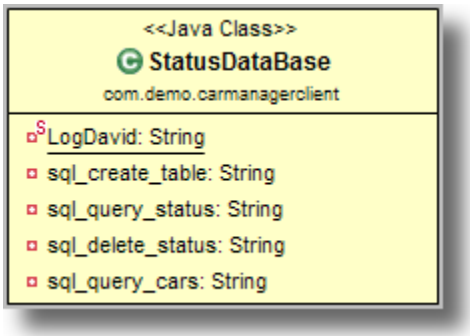


Diagrama 26

- StatusDatabase:

Esta será la clase encargada de gestionar la BBDD local de la aplicación. Como ya se ha comentado en anteriores apartados, la aplicación contará con una ligera base de datos en modo offline, para poder consultar la información aún cuando el usuario no disponga de conexión a internet.

- StatusView:

Clase encargada de mostrar al usuario, un estado concreto del vehículo. Este estado se compone de prácticamente los mismos valores de la clase CarStatus, puesto que va a ser una representación gráfica de la misma, exceptuando aquellos en los que se necesita realizar algún ajuste manual, y por características de la plataforma, se ha optado

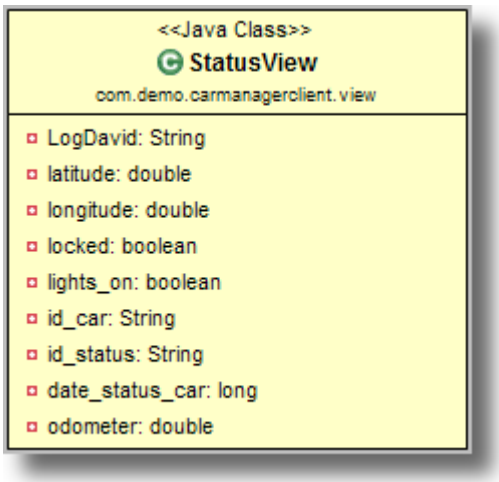
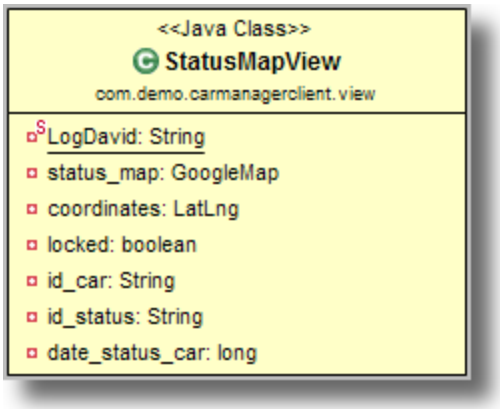


Diagrama 27

Diagrama 28



por usar variables locales, y no globales.

- StatusMapView:

Esta será la clase encargada de mostrar al usuario, de la forma más gráfica posible, la localización del vehículo, en un estado concreto. Además de ello, también mostrará si el mismo está cerrado o no.

Se ha considerado que el valor del parámetro "locked" que es el que muestra si el coche se encuentra cerrado o abierto, es el valor más importante, por tanto a pesar de tratarse en la clase StatusView, también se tratará en esta clase, y por tanto se le mostrará al usuario.

- OrderView:

De la misma forma que la clase StatusView está relacionada con la clase CarStatus, esta clase está relacionada con la clase Order. Es por tanto la representación gráfica al usuario, de dicha clase.

La clase OrderView será la encargada de recolectar todos los parámetros que el usuario quiera ajustar para enviar la orden al vehículo.

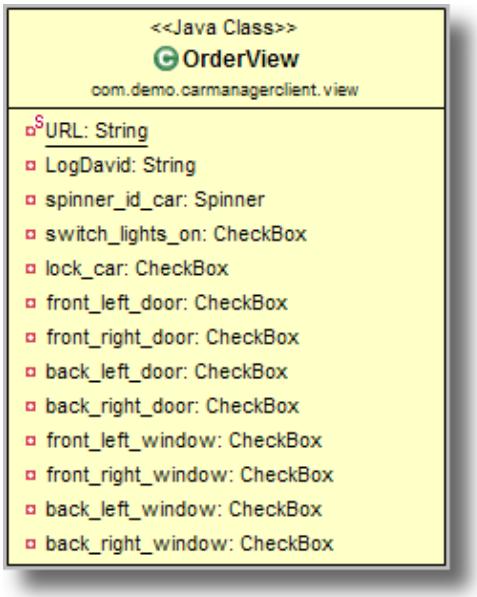


Diagrama 29

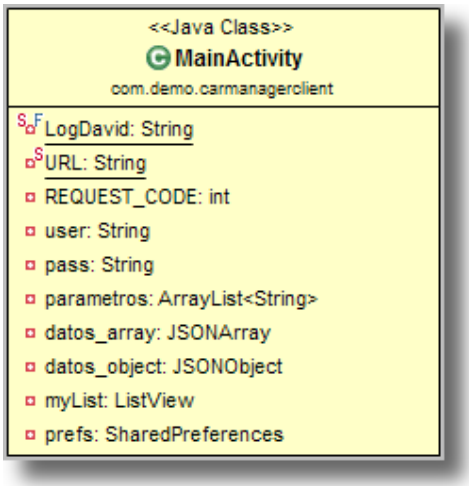
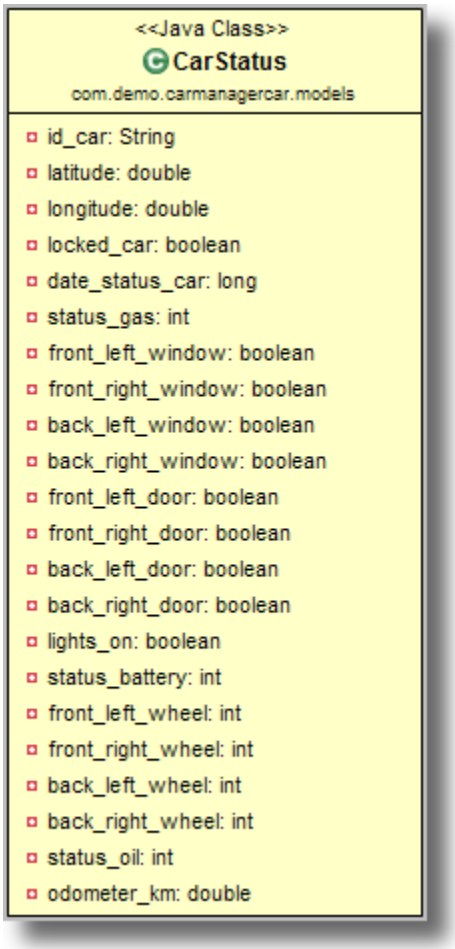


Diagrama 30

- MainActivity:

Como su propio nombre indica, esta es la clase principal de la aplicación. Es la primera clase que se muestra al abrir la aplicación y es gracias a la cual, podremos acceder a las diferentes secciones de la misma. A su vez se encarga de gestionar la identificación del usuario, y la encargada de lanzar las diferentes peticiones que se realizan al servidor.

5.3.2.2 Vehículo



- CarStatus:

Al igual que en la versión del cliente, esta va a ser la clase que nos va a servir de representación de un estado concreto del vehículo.

Los valores son los mismos que en la versión anterior, a excepción de la hora del servidor, puesto en este momento, aún este estado no ha sido enviado y por tanto no se le ha asignado una hora. Si tiene por otro lado la hora local del vehículo, que será rellenada en el momento en el que este estado se cree.

Diagrama 31

- JsonParser:

Esta clase será la encargada de parsear la respuesta recibida del servidor, en forma de json. Esta respuesta estará compuesta de un solo valor, verdadero o falso, dependiendo de si el servidor ha recogido y almacenado correctamente el estado en la base de datos o por el contrario ha habido algún problema imprevisto.

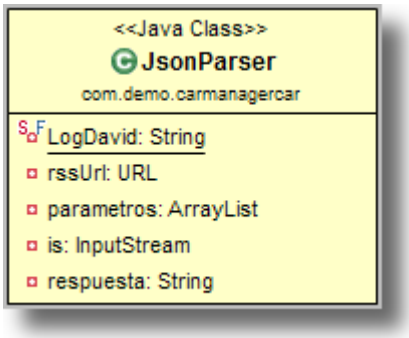
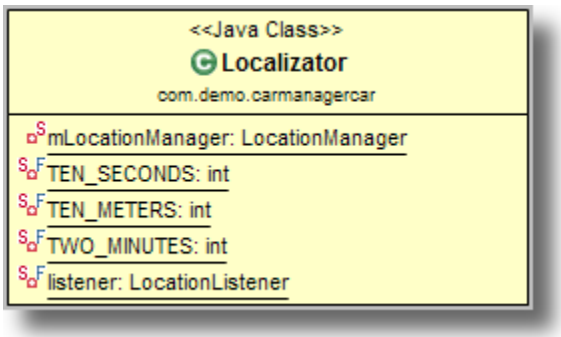


Diagrama 32



- Localizator:

Esta clase auxiliar tendrá una sola función, que es recibir en un momento determinado la localización exacta del vehículo. Esta será llamada al crear la

clase StatusServer, y justo antes de que esta sea enviada al servidor.

- StatusSender:

Clase encargada de, una vez creada la clase CarStatus, enviar dicho estado al servidor mediante llamadas post al servidor.

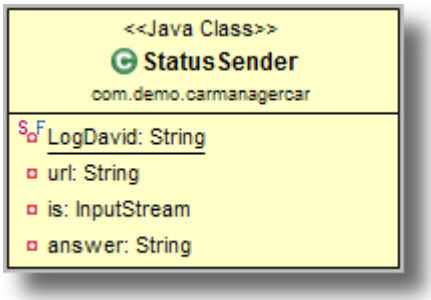


Diagrama 33

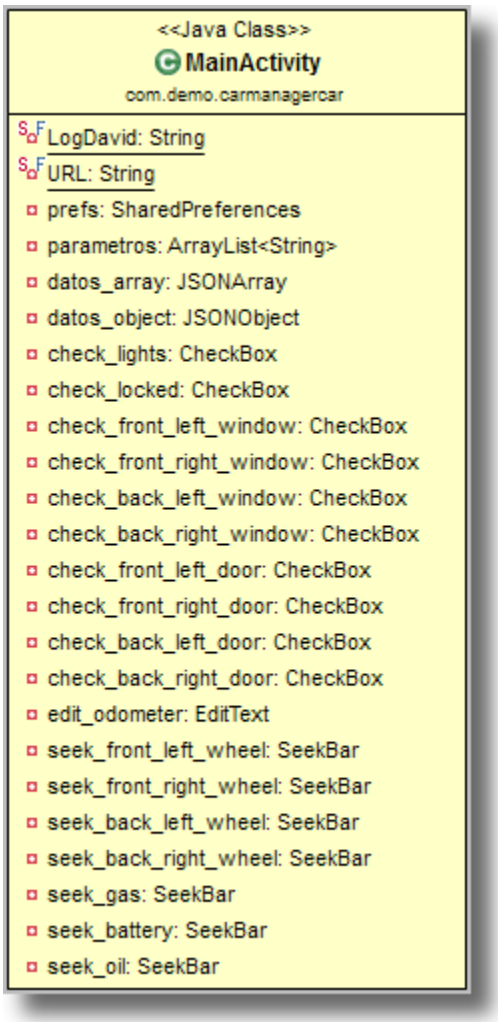


Diagrama 34

- MainActivity:

Al igual que con la versión del cliente, esta será la clase principal de la aplicación. Tendrá asociada una interfaz gráfica en la que el usuario podrá ajustar los valores a su gusto, simulando la recepción de los mismos, desde los diferentes sensores del vehículo. Desde la misma también se podrá realizar dos funciones, que son las de envío del estado al servidor, y la de recepción de la última orden enviada por el usuario y almacenada temporalmente en el servidor.

5.4 Diseño de la base de datos

5.4.1 Diagrama

A continuación se detalla el diagrama de tablas de la base de datos implementada en el servidor.

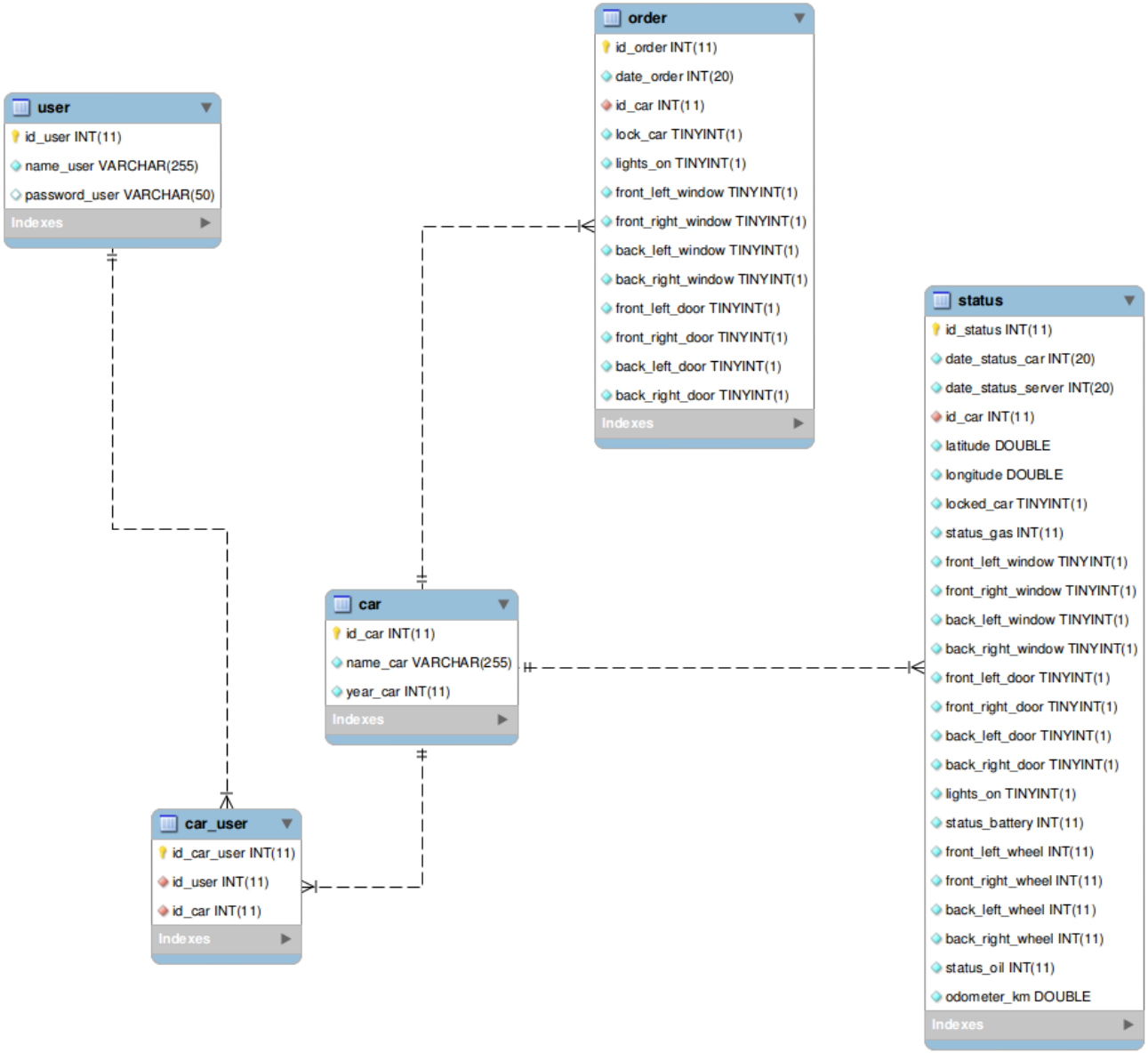

order
id_order INT(11)
date_order INT(20)
id_car INT(11)
lock_car TINYINT(1)
lights_on TINYINT(1)
front_left_window TINYINT(1)
front_right_window TINYINT(1)
back_left_window TINYINT(1)
back_right_window TINYINT(1)
front_left_door TINYINT(1)
front_right_door TINYINT(1)
back_left_door TINYINT(1)
back_right_door TINYINT(1)
Indexes
user
id_user INT(11)
name_user VARCHAR(255)
password_user VARCHAR(50)
Indexes
status
id_status INT(11)
date_status_car INT(20)
date_status_server INT(20)
id_car INT(11)
latitude DOUBLE
longitude DOUBLE
locked_car TINYINT(1)
status_gas INT(11)
front_left_window TINYINT(1)
front_right_window TINYINT(1)
back_left_window TINYINT(1)
back_right_window TINYINT(1)
front_left_door TINYINT(1)
front_right_door TINYINT(1)
back_left_door TINYINT(1)
back_right_door TINYINT(1)
lights_on TINYINT(1)
status_battery INT(11)
front_left_wheel INT(11)
front_right_wheel INT(11)
back_left_wheel INT(11)
back_right_wheel INT(11)
status_oil INT(11)
odometer_km DOUBLE
Indexes
car
id_car INT(11)
name_car VARCHAR(255)
year_car INT(11)
Indexes
car_user
id_car_user INT(11)
id_user INT(11)
id_car INT(11)
Indexes

Diagrama 35

5.4.2 Descripción

La base de datos más importante, será la almacenada en el servidor de la aplicación, y será la encargada de almacenar tanto los estados de los diferentes vehículos, como las órdenes que sus usuarios que ambos se intercambien.

Además de estos datos, hay una tabla específica para los datos principales del vehículo. En este caso y por simplificar se ha optado por almacenar únicamente el nombre del vehículo y su año de fabricación. No se han añadido otros por no ser de utilidad, pero a estos datos se les podrían sumar datos para tener una descripción más completa del vehículo, como la marca, el modelo, etc.

Al igual que con los vehículos, también se almacenan datos referentes a los usuarios. Los más importantes son el nombre de usuario y la contraseña, que nos servirán para realizar la autenticación del mismo. De la misma forma que el vehículo, en caso de interesarnos se podrían añadir datos complementarios como la fecha de nacimiento del usuario, la dirección de correo electrónico, etc.

Además de las tablas comentadas, se ha añadido una quinta tabla que nos servirá de nexo entre las tablas “car” y “user” y que nos servirá para discernir qué vehículos pertenecen a qué usuarios.

Por otra parte, en el lado del usuario también se ha creado una pequeña base de datos local, para poder consultar la información sin necesidad de conexión.

Esta base de datos es mucho más simple y sólo consta de una tabla “CarStatus”, en la que se almacenarán, al igual que en el servidor, todos los estados de los diferentes coches, pero a diferencia de esta última, en el dispositivo local, sólo se almacenarán los estados asociados a vehículos que pertenecen a un determinado usuario.

6 Construcción del proyecto

En esta fase se identificarán y plasmarán, tanto los objetivos como las principales características ya concretadas en apartados anteriores, como el análisis y todas las decisiones técnicas que las mismas han supuesto.

Además se concretarán los diferentes recursos que el alumno ha utilizado, tales como tecnologías, librerías, etc. También se realizará un estudio de las principales características de los mismos.

En este apartado se complementarán los diagramas introducidos en el apartado anterior, con las nuevas clases auxiliares necesarias para el correcto desarrollo del sistema.

Además, se incluirán fragmentos resumidos de código para explicar y ejemplificar las diferentes decisiones y metodologías seguidas durante la construcción del mismo.

Este apartado se subdivide en las siguientes secciones.

- Tecnologías empleadas.
- Librerías utilizadas
- Estructura del proyecto
- Interfaz

Como ya se especificó en anteriores apartados del proyecto, los lenguajes oficiales para desarrollo de aplicaciones y sistemas bajo Android, son Java y XML. El segundo queda relegado a cuestiones eminentemente gráficas como son el diseño de interfaces, diccionario de idiomas, estilos, etc, mientras que el primero es el encargado de llevar el control del sistema en todo momento y realizar las funciones del mismo.

La gran ventaja de la elección de Android como plataforma, es que el alumno ya tiene cierto conocimiento del lenguaje Java, puesto que es el lenguaje al que más importancia se le da durante la carrera y por tanto no era necesario que el alumno lo aprendiera de cero, como sí que hubiera sucedido de haber elegido alguna otra plataforma.

Por contra, sí que ha sido necesario que el alumno, estudie y aprenda las características específicas de que esta plataforma dispone, ya que el desarrollo de aplicaciones móviles sí que era algo nuevo para el alumno.

6.1 Tecnologías empleadas

6.1.1 Entorno de desarrollo

- Hardware:
  - El portátil del alumno. Asus, Intel i7
  - El teléfono móvil del alumno. Samsung Galaxy Nexus
- Software
  - Sistemas operativos
    - Ubuntu 12.10 - Quantal Quetzal
    - Android 4.2 - Jelly Bean
  - Base de datos
    - Mysql para el servidor
    - Sqlite para el dispositivo móvil
  - Servidor
    - Lamp (Apache, mysql, PHP)
  - IDE
    - Eclipse 4.2.2 – Juno
  - Otros
    - SqliteManager

Cabe añadir, que para el diseño de los diagramas no se han utilizado herramientas específicas, si no que se ha optado por utilizar plugins y extensiones o interfaces gráficas de las anteriores herramientas, tales como.

- Base de datos
  - Mysql Workbenc
  - PhpMyAdmin
- Diagramas de clases
  - ObjectAid para Eclipse
- Realización de la memoria
  - LibreOffice 3 y LibreOffice 4
- Codificación del proyecto
  - SDK oficial de Android para Eclipse

6.1.2 Alternativas de desarrollo

Para el desarrollo de aplicaciones Android, han surgido multitud de alternativas al SDK oficial, con dos objetivos principales en mente, que son facilitar su desarrollo, o bien permitir de manera sencilla la creación de aplicaciones portables a otra plataforma. Estos frameworks, se caracterizan por usar aplicaciones ejecutables sobre una máquina virtual, o por utilizar lenguajes de programación web.

El alumno se planteó el uso de una de estas alternativas, pero supondría un costo extra de tiempo el aclimatarse a estas plataformas, y demás el resultado no suele ser el óptimo puesto que la fluidez de estas aplicaciones es generalmente inferior, por lo que se acabó optando por el SDK oficial.

Las alternativas más comunes son:

- Phonegap
- Flex
- Appcelerator Titanium

6.1.3 Alternativas para la base de datos

Respecto al almacenamiento de datos en el servidor, se ha optado por utilizar un gestor mysql por su flexibilidad, por ser código libre y por la familiaridad que tiene el alumno con este gestor. Recientemente (en 2005) Oracle adquirió MySql y aunque en principio no se notó mucho la injerencia de una en otra, poco a poco MySql está siendo abandonado por la comunidad, optando por otras alternativas, también valoradas por el alumno en su momento, como por ejemplo:

- PostgreSQL
- MariaDB

En el lado del cliente, Android incorpora de serie una serie de librerías para utilizar de forma sencilla bases de datos de tipo SqLite por tanto ese ha sido el motivo fundamental de su elección. Sin embargo cabe destacar que aplicaciones como navegadores web (Chrome, Firefox, Opera), sistemas operativos móviles (Android, iOS, Windows Phone, Symbian) e incluso aplicaciones de escritorio (Skype, Photoshop, etc) lo han escogido como gestor de bases de datos por lo que se puede suponer su gran rendimiento y flexibilidad.

La otra opción para el desarrollo pasaba por almacenar los datos utilizando el otro mecanismo que Android dispone por defecto, que son los archivos xml. Este mecanismo es el ideal para poca cantidad de datos como por ejemplo, credenciales, preferencias, etc. Sin embargo mostraba un rendimiento inferior con cantidades grandes de datos, por lo que se acabó descartando.

6.2 Librerías utilizadas

6.2.1 CardsUI

En un principio el alumno optó por librerías gráficas como CardsUI para la representación de los datos en pantalla, y de hecho en fases tempranas del desarrollo, la aplicación tenía un aspecto similar a este.

Ilustración 1

Sin embargo, a pesar de haber realizado ya las primeras pruebas gráficas con dicha librearía, con la última actualización del SDK de Android, esta librería sufrió ciertos problemas de compatibilidad y se optó por utilizar el componente “listview” estándar, pero con una ligera personalización. De esta forma la aplicación seguiría siendo ligeramente diferente a la mayoría de aplicaciones para Android, aunque perdiendo gran parte del atractivo que hubiera tenido, de haber podido usar dicha librería.

6.2.2 Google Maps API V2

Otra librería que forma parte del proyecto es la librería oficial de Google Maps V2. Creada por Google, sirve para la utilización y manipulación de mapas Google Maps, en aplicaciones Android.

Cabe destacar que dicha librería fue presentada por Google en diciembre del 2012 y que el alumno había estado estudiando el funcionamiento de la versión anterior, por otra parte mucho más limitada, por lo que se tuvo que rediseñar parte del código de la aplicación para poder utilizar la nueva versión de la librería.

Dicha librería tiene bastantes ventajas en comparación con la anterior, como la utilización de gráficos vectoriales, la integración con los servicios de Google Play, mejoras en el sistema de caché y visualización 3D.

Por otra parte, tiene algunos inconvenientes, como la menor compatibilidad con dispositivos más antiguos.

Ilustración 2

6.3 Estructura del proyecto

Para cubrir un mayor rango de dispositivos aceptados, se comenzó realizando el proyecto con compatibilidad para Android 2.2 (Froyo) sin embargo, al no tener ningún objetivo comercial, finalmente se optó por asegurar únicamente la compatibilidad con dispositivos Android 4.1 en adelante.

Los beneficios de esta decisión, son la posibilidad de utilizar las nuevas funcionalidades introducidas en las últimas versiones, como son ActionBar, OpenGL, etc. Dado que se ha optado por las últimas versiones del sistema operativo, la estructura de dicho proyecto varia ligeramente respecto a versiones anteriores.

6.3.1 Cliente

La siguiente estructura, es la estructura típica de un proyecto de Android bajo IDE Eclipse. El proyecto consta de las siguientes partes.

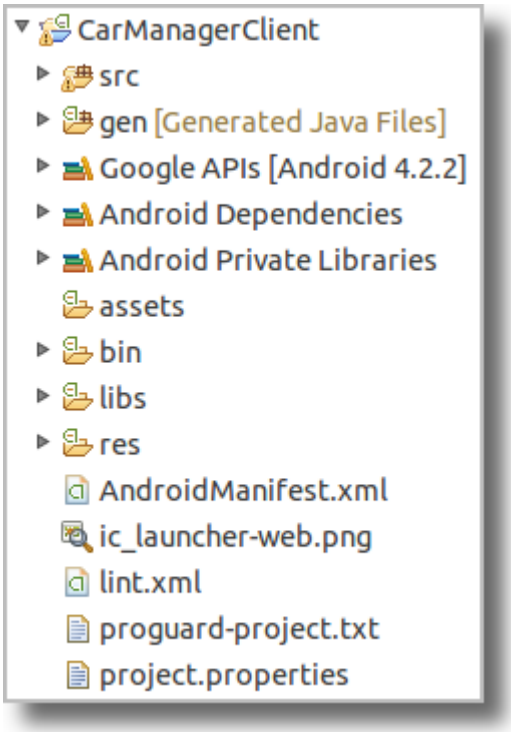
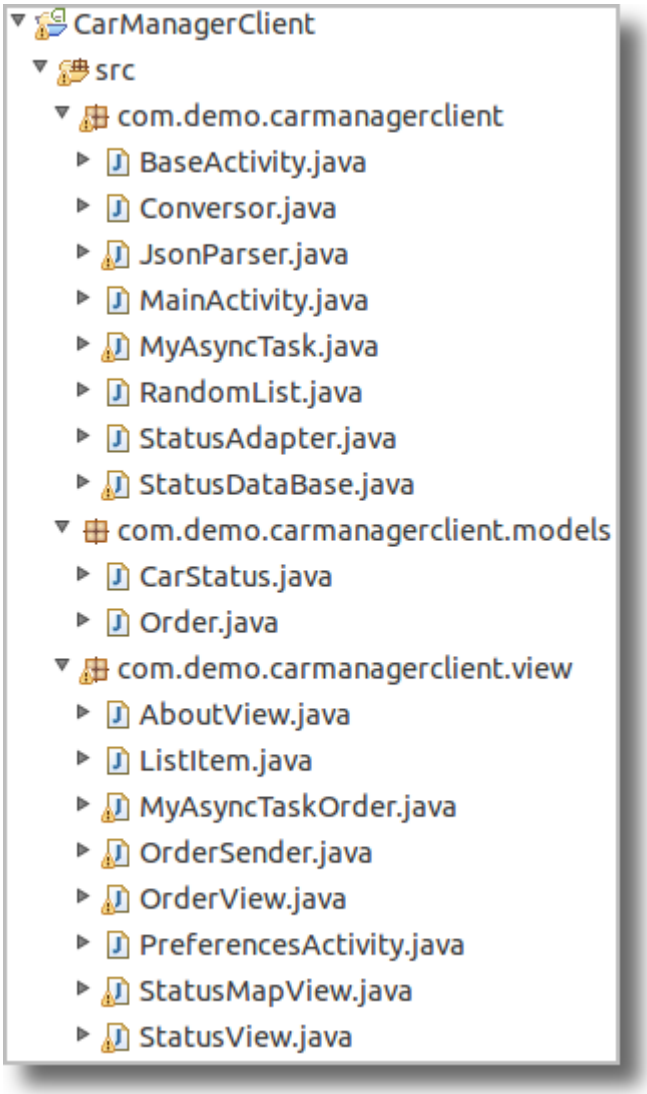


Diagrama 36

A continuación se explica detalladamente el significado y utilidad de cada una de las carpetas anteriores.



La carpeta SRC contiene todas las clases generadas durante el proyecto, cada una en sus respectivos paquetes.

El paquete models es en el que se almacenan las clases que representan los dos tipos de datos que se van a intercambiar con el servidor. El estado del vehículo, y la orden del usuario.

En el paquete view se almacenan las clases referentes a la representación gráfica de la aplicación, junto con las funcionalidades que están requieren.

En el paquete raíz, se almacenan las clases que harán funcionar la aplicación cuando el usuario la abra.

Diagrama 37

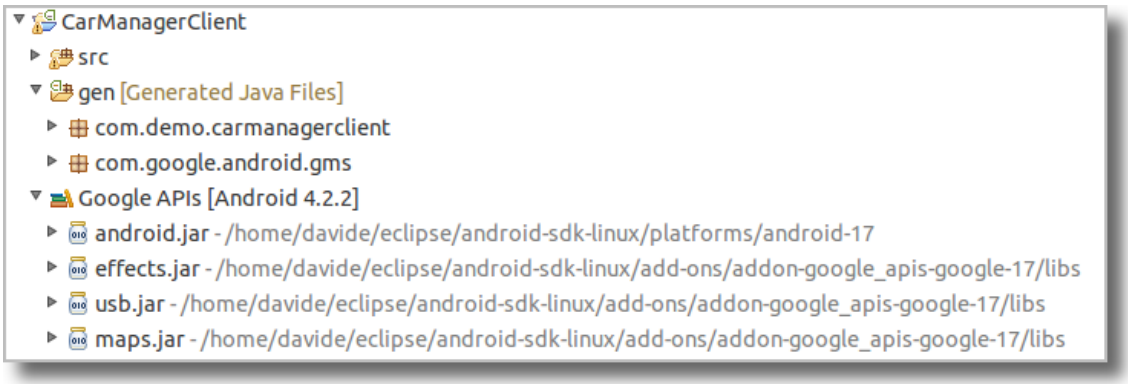
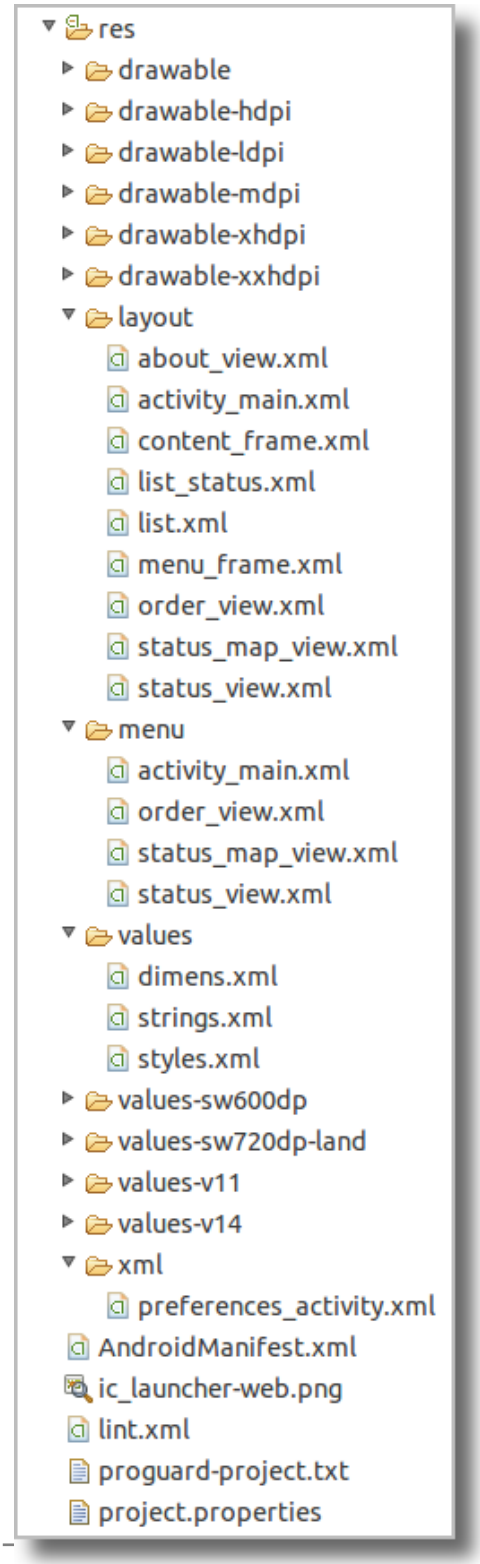


Diagrama 38

La carpeta gen Contiene una serie de elementos de código generados automáticamente al compilar el proyecto. Cada vez que generamos nuestro proyecto, la maquinaria de compilación de Android genera por nosotros una serie de ficheros fuente java dirigidos al control de los recursos de la aplicación. Importante: dado que

estos ficheros se generan automáticamente tras cada compilación del proyecto es importante que no se modifiquen manualmente bajo ninguna circunstancia.

En la carpeta Google APIs se guardan aquellas librerías que se han ido adjuntando al proyecto. En concreto la librería android.jar es una librería que se adjunta automáticamente en cualquier proyecto, mientras que las otras tres librerías, se han adjuntado al haber necesitado hacer uso de los servicios de Google. En este caso para la manipulación de los mapas de Google Maps.



La carpeta res contiene todos los ficheros de recursos necesarios para el proyecto: imágenes, vídeos, cadenas de texto, etc. Los diferentes tipos de recursos se distribuyen entre las siguientes subcarpetas:

drawable: Contiene las imágenes [y otros elementos gráficos] usados en por la aplicación. Para definir diferentes recursos dependiendo de la resolución y densidad de la pantalla del dispositivo se suele dividir en varias subcarpetas, dependiendo del dispositivo en el que se vaya a ejecutar la aplicación.

layout: Contiene los ficheros de definición XML de las diferentes pantallas de la interfaz gráfica. Para definir distintos layouts dependiendo de la orientación del dispositivo se puede dividir en dos subcarpetas, dependiendo de como se muestre la aplicación (vertical y horizontal), pero en este proyecto no ha sido necesario.

menu: Contiene la definición XML de los menús que se mostrarán en las diferentes pantallas de la aplicación.

values: Contiene otros ficheros XML de recursos de la aplicación, como por ejemplo cadenas de texto (strings.xml), estilos (styles.xml), colores (colors.xml), arrays de valores (arrays.xml), etc.

Diagrama 39

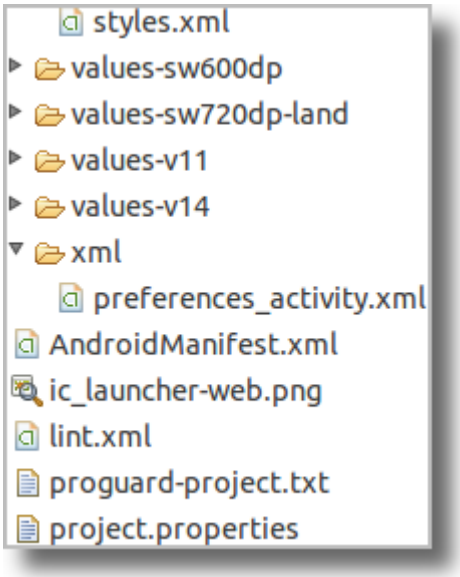


Diagrama 40

xml: Contiene otros ficheros XML de datos utilizados por la aplicación. En el caso que nos ocupa, se ha utilizado para definir la actividad referente a las preferencias de la aplicación.

AndroidManifest.xml: Contiene la definición en XML de los aspectos principales de la aplicación, como por ejemplo su identificación (nombre, versión, icono, …), sus componentes (pantallas, mensajes, …), las librerías auxiliares utilizadas, o los permisos necesarios para su ejecución.

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

    <uses-sdk
        android:minSdkVersion="11"
        android:targetSdkVersion="17" />

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

    <permission
        android:name="org.example.ejemplogooglemaps.permission.MAPS_RECEIVE"
        android:protectionLevel="signature" />

    <uses-permission android:name="org.example.ejemplogooglemaps.permission.MAPS_RECEIVE" />
    <uses-permission android:name="com.google.android.providers.gsf.permission.READ_GSERVICES" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />

    <uses-feature
        android:glEsVersion="0x00020000"
        android:required="true" />

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <meta-data
            android:name="com.google.android.maps.v2.API_KEY"
            android:value="AIzaSyBrpVUJ4F1mU1DGMbrdPNY_axOC2cqevDI" />

        <activity
            android:name="com.demo.carmanagerclient.MainActivity"
            android:label="@string/app_name"
            android:screenOrientation="portrait" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name="com.demo.carmanagerclient.view.StatusView" />
        <activity android:name="com.demo.carmanagerclient.view.StatusMapView" />
        <activity android:name="com.demo.carmanagerclient.view.OrderView" />
        <activity android:name="com.demo.carmanagerclient.view.AboutView" />
        <activity
            android:name="com.demo.carmanagerclient.view.PreferencesActivity"
            android:theme="@android:style/Theme.Holo" />
    </application>

</manifest>
```

Diagrama 41

Para el correcto funcionamiento de la aplicación, se han añadido a este archivo los siguientes parámetros.

- Permisos:
  - Acceso a Google Maps
  - Utilización de los servicios de Google
  - Acceso a internet
  - Comprobación del estado de la red
  - Almacenamiento de datos en caché
  - Acceso a los dos tipos de geolocalización que existen en Android (GPS y triangulación GPRS/WIFI)

- Credenciales
  - Credenciales de acceso al API de Google Maps
- Pantallas de la aplicación
  - MainActivity
    - Nombre de aplicación
    - Definición como pantalla principal
    - Icono de la aplicación
    - Orientación vertical
  - StatusView
  - StatusMapView
  - OrderView
  - AboutView
- Otros
  - Uso de aceleración gráfica OpenGL
  - Tema y diseño de la aplicación
  - Versión mínima para su ejecución (versión mínima del SDK)
  - Versión de la aplicación
  - etc

6.3.2 Vehículo

En la parte de la aplicación del vehículo, la estructura varía ligeramente puesto que el funcionamiento no es el mismo, pero la definición de clases y paquetes es muy similar.

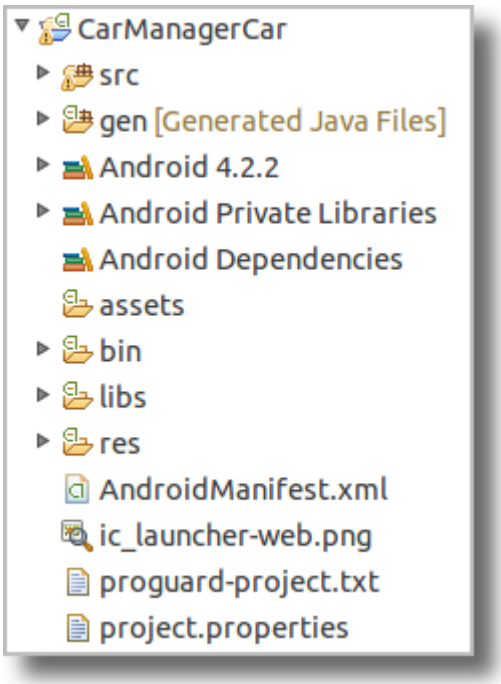


Diagrama 42

En este caso, la única diferencia con el proyecto anterior, es el nombre del mismo.

La carpeta src sigue conteniendo las clases definidas para la aplicación.

En este caso sólo hay un modelo, puesto que se trata de forma idéntica, el estado del coche, como la orden del usuario, ya que esta se convertirá en estado una vez recibida.

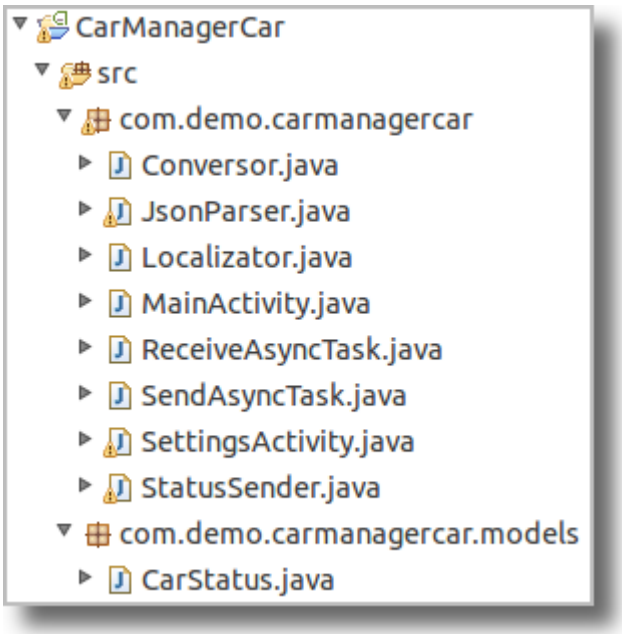


Diagrama 43

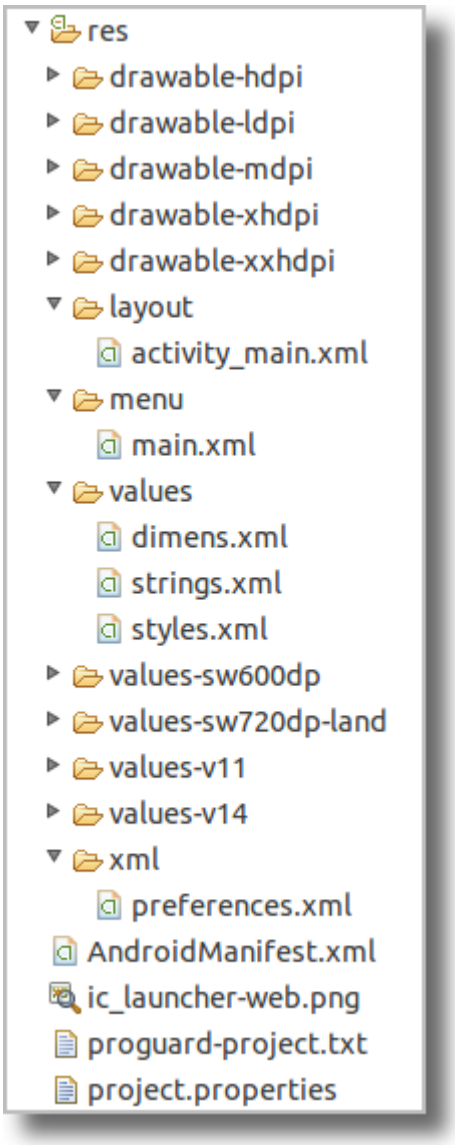


Diagrama 44

En este caso se puede comprobar que la aplicación requiere de menos pantallas (sólo se muestra una, que es la encargada de recibir los datos del vehículo y transmitirlos)

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

    <uses-sdk
        android:minSdkVersion="14"
        android:targetSdkVersion="17" />

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

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

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

</manifest>
```

Diagrama 45

En este caso se ha añadido los siguientes parámetros:

- Permisos
  - Acceso a internet
  - Comprobación del estado de la red
  - Acceso a los dos tipos de geolocalización que existen en Android (GPS y triangulación GPRS/WIFI)
- Pantallas de la aplicación
  - MainActivity
    - Nombre de aplicación
    - Definición como pantalla principal
    - Icono de la aplicación
- Otros
  - Tema y diseño de la aplicación
  - Versión mínima para su ejecución (versión mínima del SDK)
  - Versión de la aplicación
  - etc

6.3.3 Servidor

Scripts de creación de las tablas y referencias de la base de datos del servidor.

6.3.3.1 CAR

```
CREATE TABLE IF NOT EXISTS `car` (
  `id_car` int(11) NOT NULL DEFAULT '0',
  `name_car` varchar(255) COLLATE latin1_spanish_ci NOT NULL,
  `year_car` int(11) NOT NULL,
  PRIMARY KEY (`id_car`)
)
```

Diagrama 46

6.3.3.2 USER

```
CREATE TABLE IF NOT EXISTS `user` (
  `id_user` int(11) NOT NULL DEFAULT '0',
  `name_user` varchar(255) COLLATE latin1_spanish_ci NOT NULL,
  `password_user` varchar(50) COLLATE latin1_spanish_ci DEFAULT NULL,
  PRIMARY KEY (`id_user`)
)
```

Diagrama 47

6.3.3.3 CAR_USER

```
CREATE TABLE IF NOT EXISTS `car_user` (
  `id_car_user` int(11) NOT NULL,
  `id_user` int(11) NOT NULL,
  `id_car` int(11) NOT NULL,
  PRIMARY KEY (`id_car_user`),
  KEY `fk_id_car` (`id_car`),
  KEY `fk_id_user` (`id_user`)
)
ALTER TABLE `car_user`
  ADD CONSTRAINT `fk_id_car` FOREIGN KEY (`id_car`) REFERENCES `car` (`id_car`) ON DELETE NO ACTION ON UPDATE NO ACTION,
  ADD CONSTRAINT `fk_id_user` FOREIGN KEY (`id_user`) REFERENCES `user` (`id_user`) ON DELETE NO ACTION ON UPDATE NO ACTION;
```

Diagrama 48

6.3.3.4 STATUS

```
CREATE TABLE IF NOT EXISTS `status` (
  `id_status` int(11) NOT NULL AUTO_INCREMENT,
  `date_status_car` int(20) NOT NULL,
  `date_status_server` int(20) NOT NULL,
  `id_car` int(11) NOT NULL,
  `latitude` double NOT NULL,
  `longitude` double NOT NULL,
  `locked_car` tinyint(1) NOT NULL,
  `status_gas` int(11) NOT NULL,
  `front_left_window` tinyint(1) NOT NULL,
  `front_right_window` tinyint(1) NOT NULL,
  `back_left_window` tinyint(1) NOT NULL,
  `back_right_window` tinyint(1) NOT NULL,
  `front_left_door` tinyint(1) NOT NULL,
  `front_right_door` tinyint(1) NOT NULL,
  `back_left_door` tinyint(1) NOT NULL,
  `back_right_door` tinyint(1) NOT NULL,
  `lights_on` tinyint(1) NOT NULL,
  `status_battery` int(11) NOT NULL,
  `front_left_wheel` int(11) NOT NULL,
  `front_right_wheel` int(11) NOT NULL,
  `back_left_wheel` int(11) NOT NULL,
  `back_right_wheel` int(11) NOT NULL,
  `status_oil` int(11) NOT NULL,
  `odometer_km` double NOT NULL,
  PRIMARY KEY (`id_status`),
  KEY `fk_id_car` (`id_car`)
)
ALTER TABLE `status`
  ADD CONSTRAINT `status_ibfk_1` FOREIGN KEY (`id_car`) REFERENCES `car` (`id_car`);
```

Diagrama 49

6.3.3.5 ORDER

```
CREATE TABLE IF NOT EXISTS `order` (
  `id_order` int(11) NOT NULL AUTO_INCREMENT,
  `date_order` int(20) NOT NULL,
  `id_car` int(11) NOT NULL,
  `lock_car` tinyint(1) NOT NULL,
  `lights_on` tinyint(1) NOT NULL,
  `front_left_window` tinyint(1) NOT NULL,
  `front_right_window` tinyint(1) NOT NULL,
  `back_left_window` tinyint(1) NOT NULL,
  `back_right_window` tinyint(1) NOT NULL,
  `front_left_door` tinyint(1) NOT NULL,
  `front_right_door` tinyint(1) NOT NULL,
  `back_left_door` tinyint(1) NOT NULL,
  `back_right_door` tinyint(1) NOT NULL,
  PRIMARY KEY (`id_order`),
  KEY `fk_id_car` (`id_car`)
)
ALTER TABLE `order`
  ADD CONSTRAINT `order_ibfk_1` FOREIGN KEY (`id_car`) REFERENCES `car` (`id_car`);
```

Diagrama 50

6.4 Interfaz

6.4.1 Usuario

Ilustración 3

Al entrar en la aplicación, el usuario verá un listado de los últimos estados que se han enviado desde los vehículos que él tiene asignados. En cada estado se verá el identificador de dicho estado, y la fecha/hora a la que se ha enviado.

A su vez tendrá un botón en la parte superior para poder actualizar dichos estados, realizando una consulta al servidor. Otro botón con el cual podrá enviar una orden a uno de los vehículos, y un menú desde el cual poder entrar a los ajustes y borrar sus credenciales.

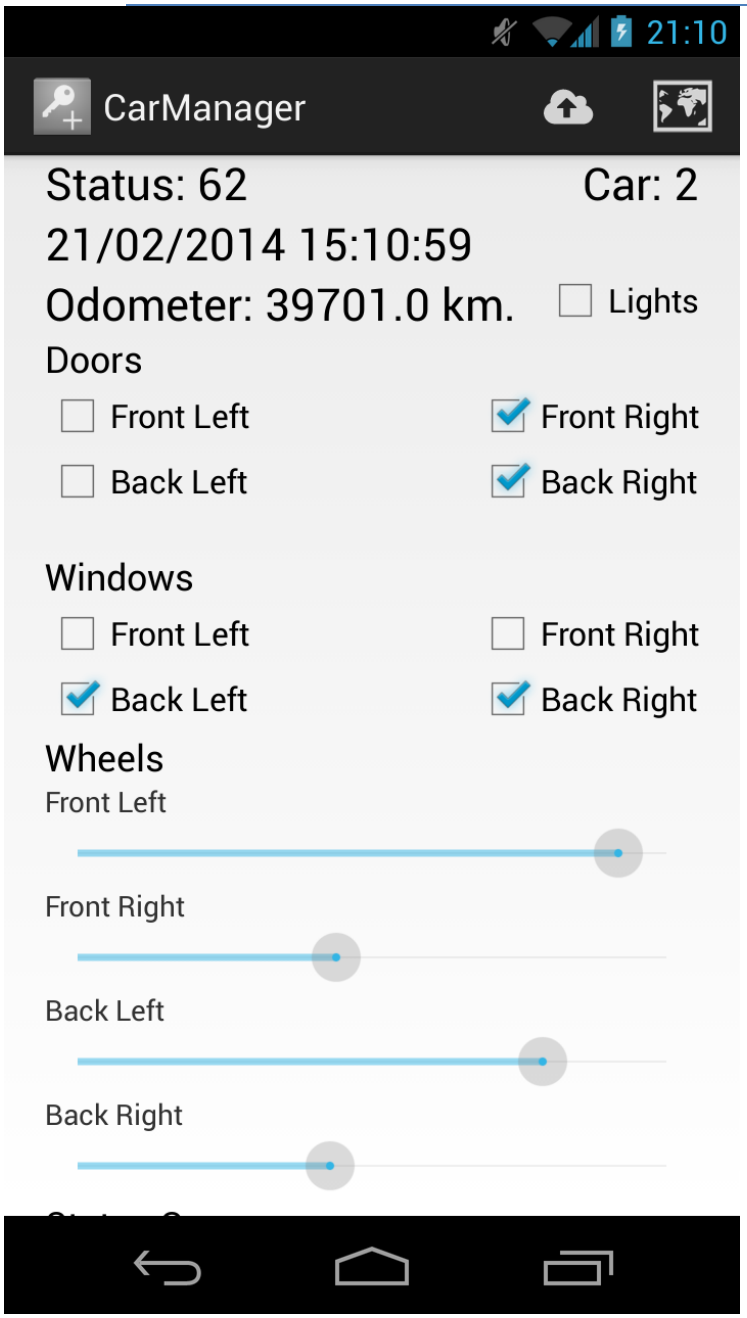


Ilustración 4

Cuando el cliente seleccione uno de los estados del listado, verá una pantalla en la cual se le mostrarán mediante indicadores, los valores de dicho estado.

Estos valores son por ejemplo, qué puertas o ventanas se encontraban abiertas, el valor del odómetro, el nivel de presión de las ruedas, nivel de carga de gasolina, aceite y batería, si las luces se encuentran o no encendidas, la fecha de dicho estado y a qué coche pertenece. Importante destacar que dichos controles estarán totalmente desactivados, puesto que solo son informativos, no se pueden modificar.

A su vez, tendrá dos botones para poder interactuar. Con el primero de ellos, podrá enviar una orden a dicho vehículo. Con el segundo, podrá ver en un mapa, la localización exacta del vehículo, cuando se envió dicho estado.

Por otra parte, para mostrar si este vehículo se encontraba abierto cuando se envió el estado, se ha optado por cambiar el color de la barra de acciones en vez de mostrar otro control, para darle mayor importancia. Se mostrará un ejemplo a continuación.

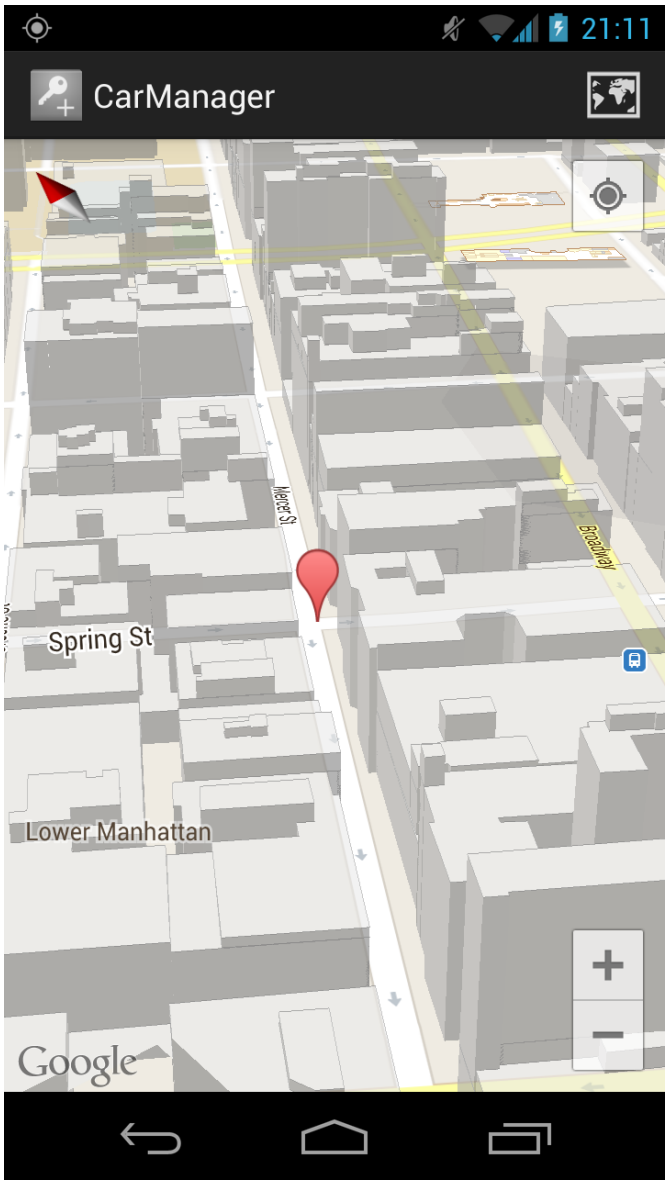


Ilustración 5

Para mostrar la localización del vehículo, se ha optado por incluir un mapa de Google Maps con un puntero que nos indica dicha localización. Este mapa, como se puede observar, se puede colocar en modo 3D para que le sea más fácil al usuario, conocer dicha localización.

También se ha añadido un botón que muestra la localización actual del usuario, para que este pueda hacerse una idea más exacta de dónde se encuentra dicho vehículo con relación a su situación.

Por último, se ha añadido un botón en la barra de acciones, con el cual el usuario podrá cambiar las vista, alternando entre una vista de mapa, y otra con fotografías de satélite.

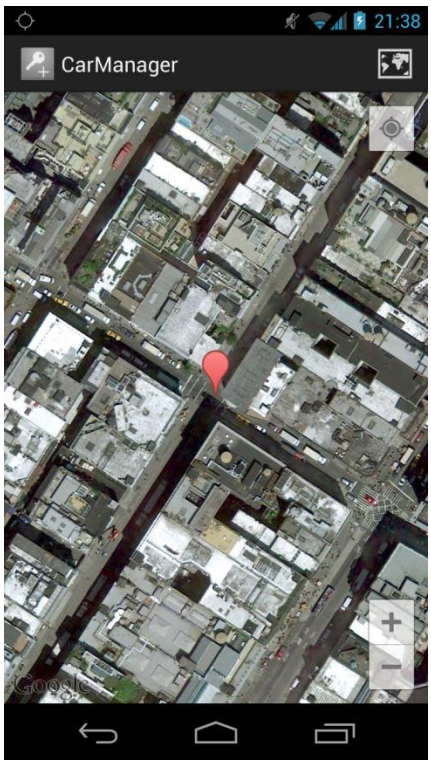


Ilustración 6

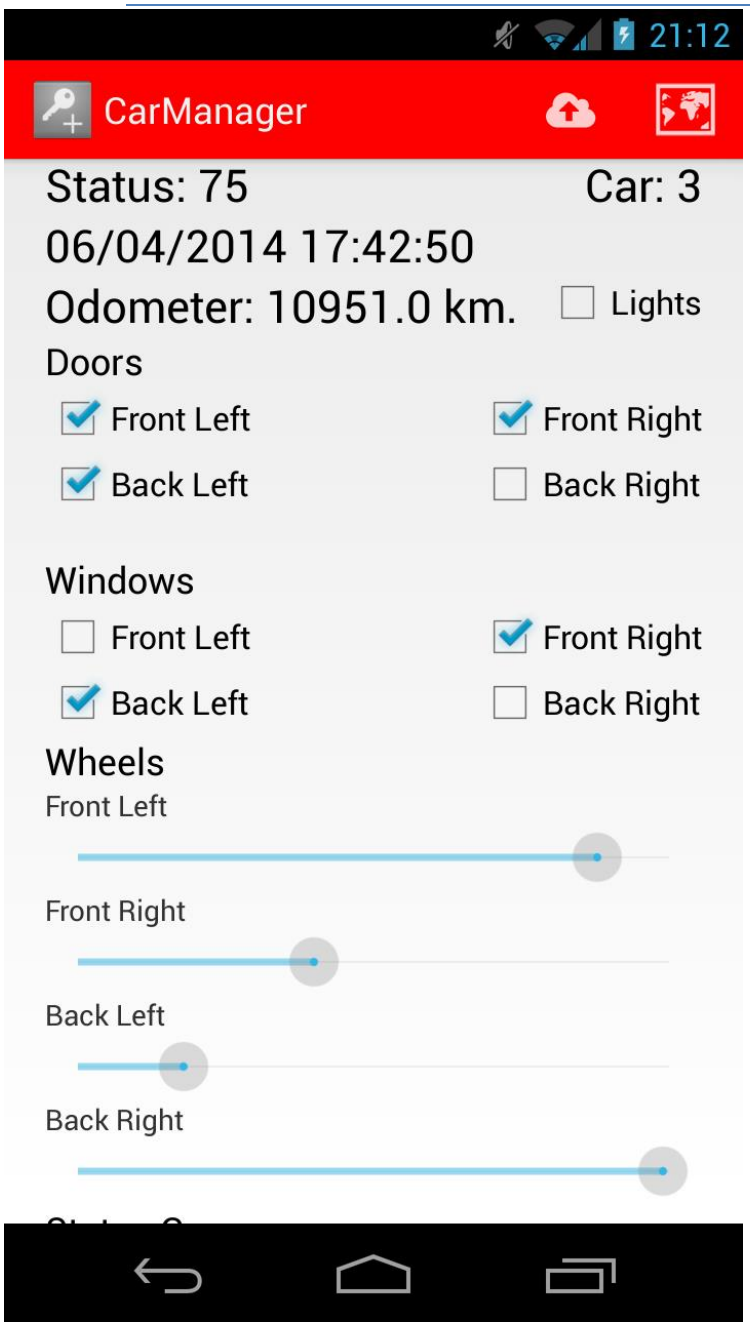


Ilustración 7

Como ya se ha comentado anteriormente, cuando el vehículo se encuentre abierto, dicho estado se representará con un color rojo, para darle mayor importancia.

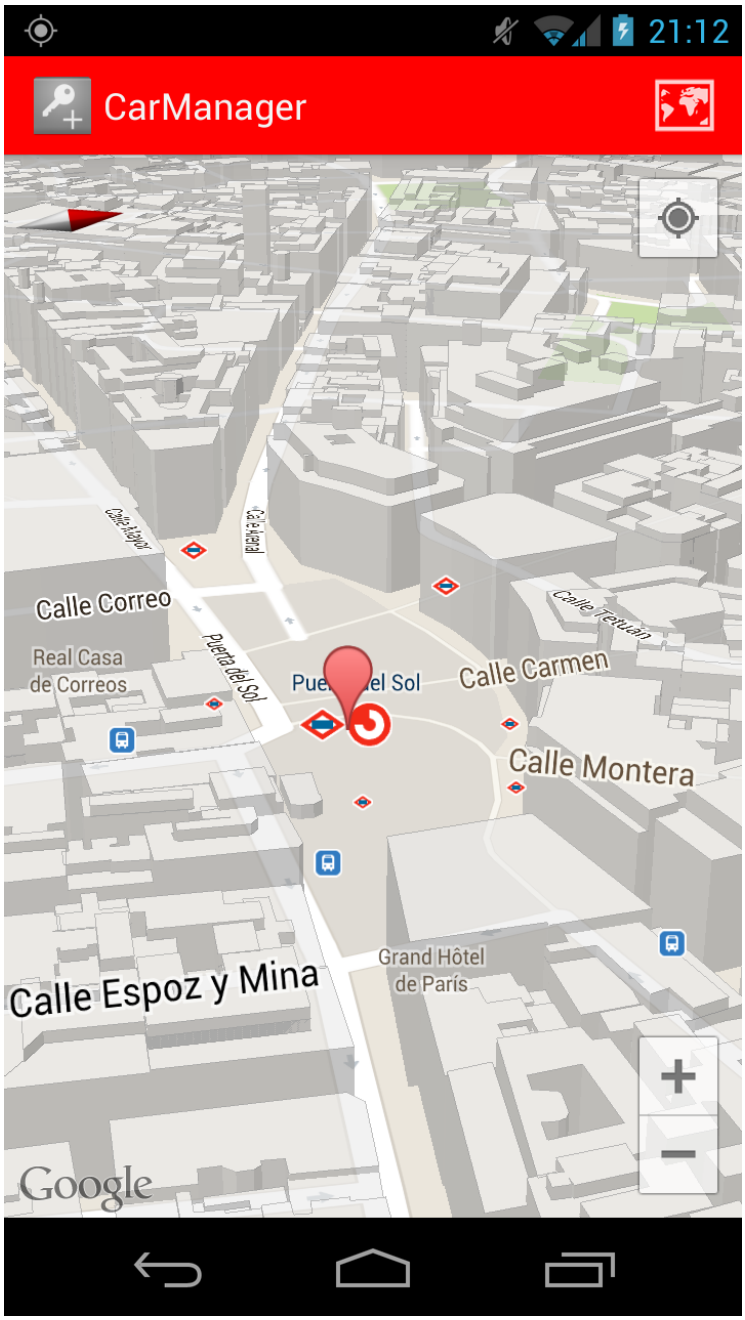


Ilustración 8

Igualmente con la representación con mapa de dicho estado.

El funcionamiento de dicha pantalla no difiere en nada, con respecto a la anterior. Simplemente se ha cambiado el color para darle mayor énfasis.

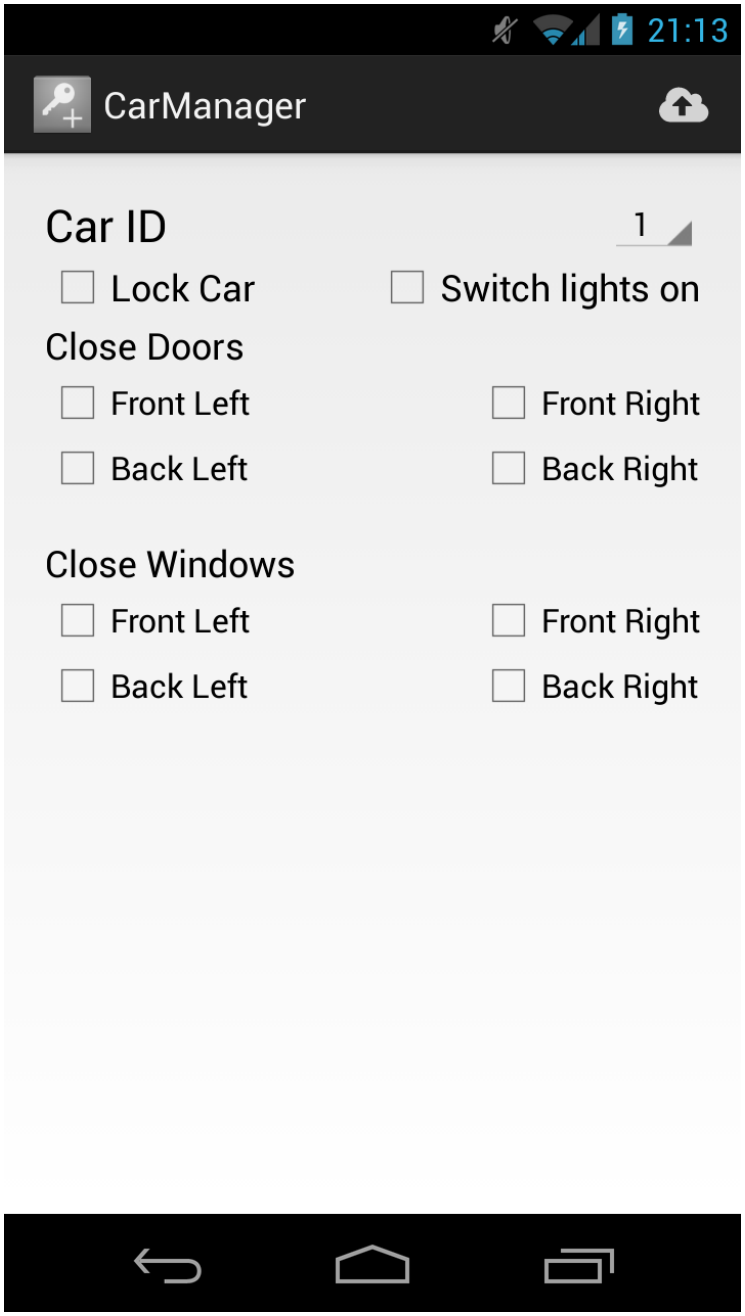


Ilustración 9

La pantalla de enviar órdenes, es una simplificación de la pantalla de estado, dejando disponibles únicamente aquellos parámetros que se pueden modificar a distancia.

Así por ejemplo, el usuario puede bloquear el coche, cerrar las ventanillas, apagar las luces, etc a distancia. Sin embargo jamás podrá cambiar el nivel del depósito de gasolina o de la carga de batería.

Una vez rellenados dichos valores, el usuario dispone de un botón para enviar dicha orden al vehículo.

Cabe destacar, que a esta pantalla se puede acceder desde el listado principal, como desde un estado concreto, sin embargo esto afectará ligeramente al comportamiento de la misma.

En la captura de la izquierda, se muestra la pantalla de órdenes, accedida desde un estado concreto, y por tanto en la parte superior derecha, ya se nos queda marcado el identificador del vehículo al que queremos enviar dicha orden.

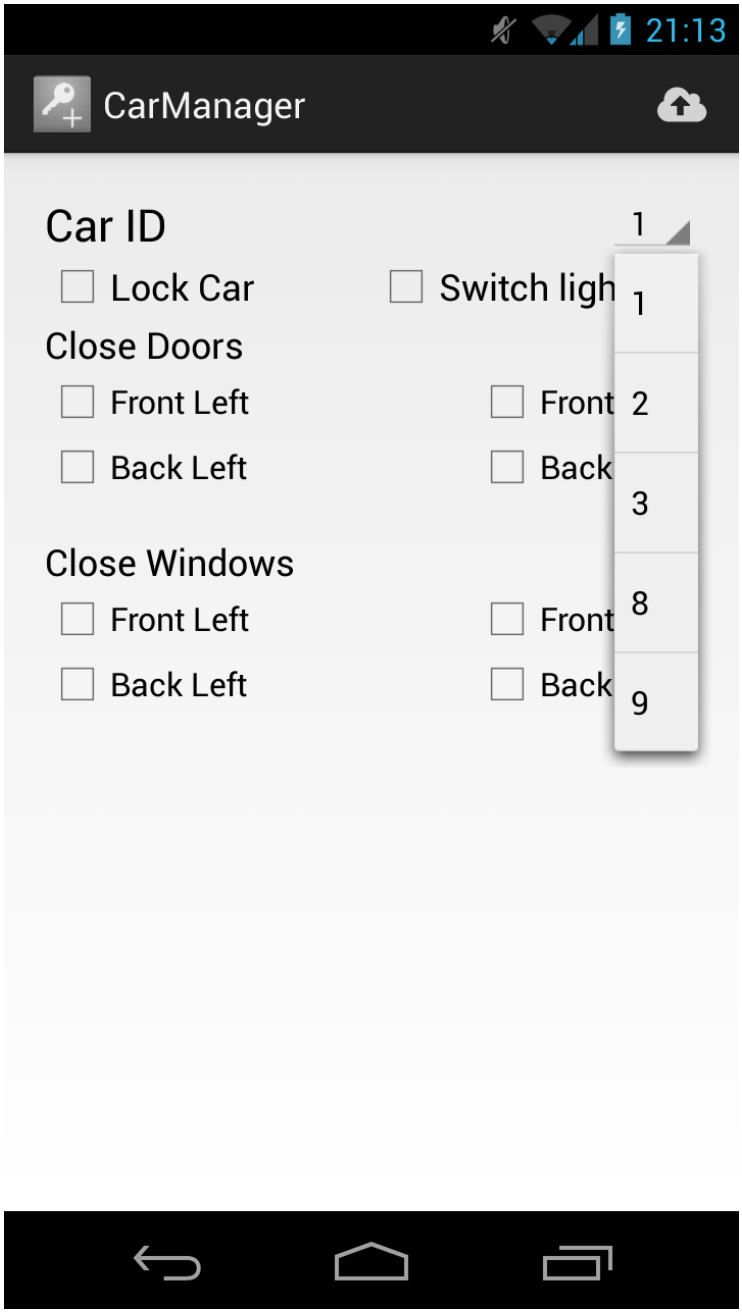


Ilustración 10

En este caso, se nos muestra la pantalla de órdenes, accedida desde el listado principal. En este caso se observa como el identificador del vehículo está habilitado para seleccionar el vehículo al que queramos enviar dicha orden.

En este selector, no se podrá elegir cualquier vehículo, sólo estarán disponibles aquellos vehículos asociados al usuario concreto.

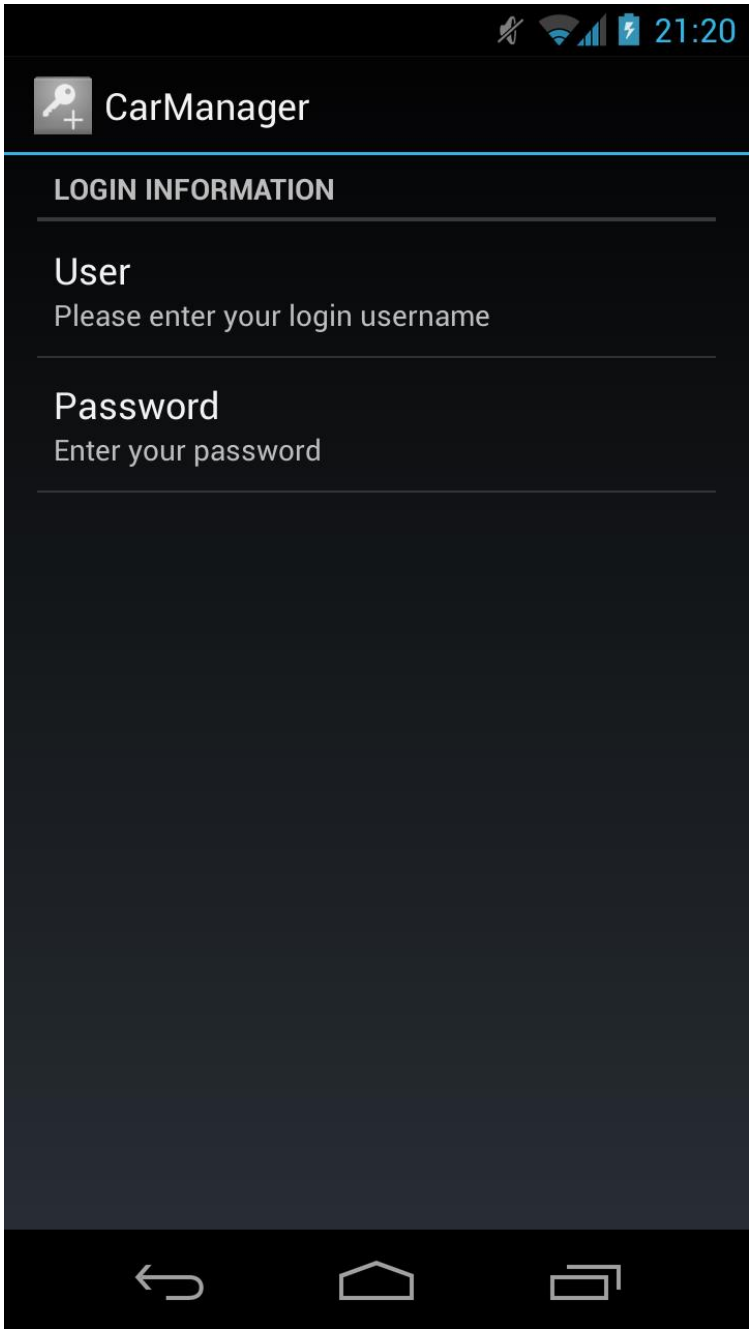


Ilustración 11

Por último, la pantalla de ajustes es accesible desde el menú y sirve para que el usuario se identifique con su nombre y contraseña.

Si el usuario y contraseña introducido es incorrecto, la aplicación borrará de la base de datos, todos lo que tenga almacenado, evitando de esta forma que dicho usuario tenga acceso a datos a los que no debería poder tener.

En caso de ser correctos, al introducirlos, dicho usuario podrá volver a la pantalla principal y actualizar los datos desde el botón correspondiente.

6.4.2 Vehículo

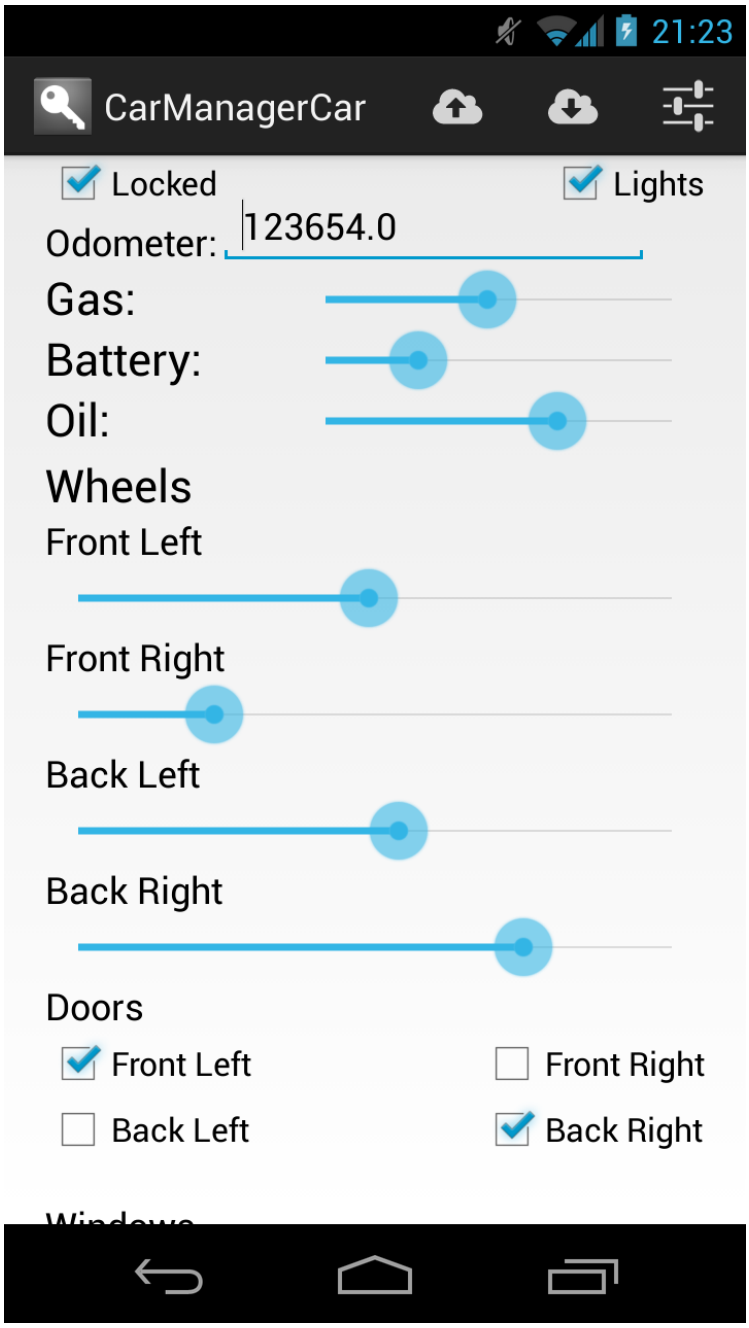


Ilustración 12

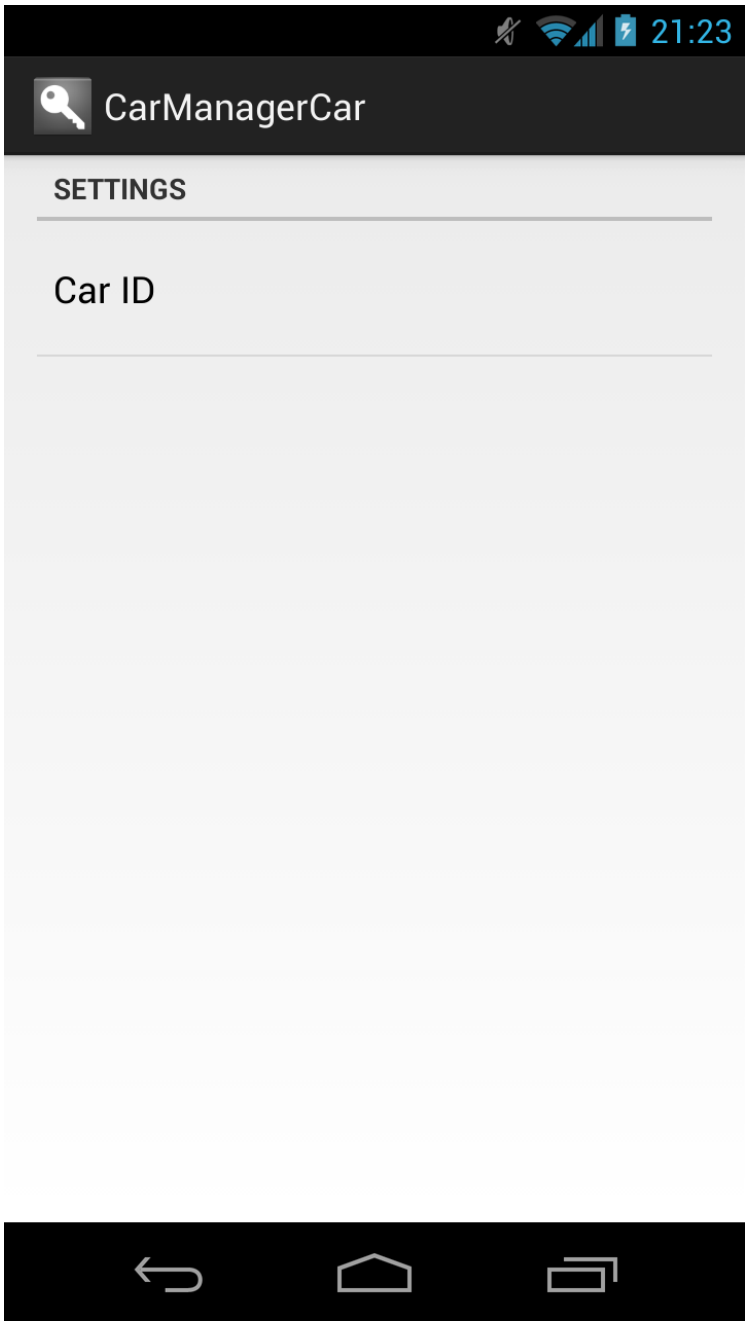
Por otra parte, la pantalla principal del vehículo consta únicamente de los controles en los cuales, se indicará el estado de dicho vehículo.

Además de esto, se dispone de 3 botones en la barra de acciones.

El primero es para enviar dicho estado al servidor.

El segundo, para recibir la última orden enviada por el usuario a dicho vehículo y por tanto actualizar los datos con los nuevos.

El tercero nos lleva a la pantalla de ajustes.



Esta pantalla de ajustes, sólo dispone de un campo, que es donde indicaremos el id del vehículo.

Ilustración 13

7 Pruebas del proyecto

Basándonos en el análisis de casos de uso que se puede encontrar en el apartado correspondiente, en esta sección se van a ir detallando las pruebas realizadas con la aplicación para comprobar si el comportamiento es el esperado y deseado.

Dichas pruebas se fueron realizando durante las sucesivas etapas de la fase de implementación, para comprobar el correcto funcionamiento de cada una de las funcionalidades. Además se estableció la realización de algunas de ellas, con carácter más general, al final de dicho proceso, con el fin de comprobar el comportamiento general del sistema.

PRUEBA 1	
Tarea	Lectura y parseo de un recurso online
Descripción	Búsqueda de un recurso online que ofrezca una serie de noticias en formato JSON para poder realizar una conexión al mismo, recoger dicho JSON, analizarlo, y poder mostrar dicha información en pantalla
Realización	Se buscaron concretamente periódicos, webs de noticias, o blogs y no se encontró ninguna que ofreciera las distintas entradas en formato JSON. Al final se optó por realizar la conexión al mismo, pero leer dichas noticias en formato XML y posteriormente parsearlas y mostrarlas
Resultado esperado	Un listado de noticias en el que cada elemento muestre el título y las primeras palabras del artículo completo
Resultado obtenido	Correcto

Tabla 7

PRUEBA 2	
Tarea	Parseo de archivo JSON
Descripción	Generación manual de un archivo en formato JSON con unos campos definidos con anterioridad para la realización de parseador del mismo, en lenguaje JAVA
Realización	Siguiendo un formato previamente definido, y gracias a algunas herramientas online de verificación JSON, se generó un archivo JSON de pruebas. Una vez generado, se procedió a generar las clases y métodos necesarios para la lectura de dicho archivo, y la instanciación de varios objetos de una clase, creada previamente para almacenarlos.
Resultado esperado	Un array de objetos de la clase definida para dicho archivo
Resultado obtenido	Correcto

Tabla 8

PRUEBA 3	
Tarea	Pruebas de integridad de la base de datos
Descripción	Realización de las pruebas pertinentes con la base de datos del servidor, para verificar la integridad de los mismos
Realización	Una vez establecida la estructura de datos del servidor, y las relaciones entre las diferentes tablas, se realizaron varios intentos de inserción de datos para intentar saltarse alguna de las diferentes relaciones.
Resultado esperado	Errores de todos los intentos de inserción de datos
Resultado obtenido	Correcto

Tabla 9

PRUEBA 4	
Tarea	Comprobación de base de datos del cliente
Descripción	Comprobación de la inserción de datos, descargados desde la base de datos del servidor, en la base de datos local, almacenada en el dispositivo del usuario.
Realización	Una vez conseguidos los mecanismos de conexión al servidor y de parseo de archivos JSON, se realizó el proceso con el servidor de la aplicación, y posteriormente el almacenamiento de los datos descargados, de manera local, en una base de datos sqlite en el dispositivo del usuario. Posteriormente se extrajo dicha base de datos, y se realización pruebas para asegurar que los datos almacenados, coincidían con los datos previamente insertados en la base de datos del servidor.
Resultado esperado	Coincidencia total entre los datos de la base de datos local, con los datos de la base de datos del servidor.
Resultado obtenido	Correcto

Tabla 10

PRUEBA 5	
Tarea	Comprobación de uso de los mapas de Google
Descripción	Realización de una pantalla en la aplicación, en la cual mostrar un mapa de Google, marcando una localización
Realización	Inicialmente se realizó un proyecto aparte con el único objetivo de insertar en la pantalla principal, un mapa en el cual marcar con un puntero una localización fija. Posteriormente se incluyó dicha funcionalidad en la aplicación, utilizando para la localización, las coordenadas de un estado concreto.
Resultado esperado	Un mapa totalmente manipulable, en el que quede reflejada una localización concreta.
Resultado obtenido	Correcto

Tabla 11

PRUEBA 6	
Tarea	Envío de datos de la aplicación al servidor
Descripción	Conexión mediante http con un servidor y envío de datos mediante POST
Realización	Se creó una aplicación sencilla en la que se enviaban dos datos (usuario y contraseña) al servidor, este realizaba una consulta en la base de datos, y respondía con un booleano indicando si esta combinación existía o no en la base de datos. Una vez realizado, se integró esta funcionalidad tanto en la consulta de estados al servidor (indicando el usuario y la contraseña para recibir únicamente los datos de los vehículos de dicho usuario) como en el envío de órdenes del usuario al vehículo (enviado todos los datos correspondientes a dicha orden)
Resultado esperado	Envío de dichos datos, y recepción de los mismos en el servidor de la aplicación.
Resultado obtenido	Correcto

Tabla 12

PRUEBA 7	
Tarea	Identificación del usuario
Descripción	Comprobación del usuario y muestra de sus datos correspondientes
Realización	La primera vez que se ejecute la aplicación, o cuando el usuario borre sus credenciales, se exigirá al usuario que vuelva a introducir su identificación en el sistema.
Resultado esperado	Forzar al usuario a identificarse en el sistema en caso de que no lo haya hecho aún. Además, si el mismo borra sus credenciales, un borrado completo de los datos descargados con anterioridad.
Resultado obtenido	Correcto

Tabla 13

PRUEBA 8	
Tarea	Detalles del estado
Descripción	Muestra en detalle del estado seleccionado.
Realización	En el listado de los diferentes estados de la pantalla principal, cada uno de los diferentes elementos, es a su vez un botón que nos muestra una nueva ventana en la cual se pueden ver todos los detalles de dicho estado, de forma visual para el usuario.
Resultado esperado	Mostrar todos los detalles del estado en una nueva pantalla de manera correcta.
Resultado obtenido	Correcto

Tabla 14

PRUEBA 9	
Tarea	Cambiar configuración
Descripción	Al pulsar el botón de configuración se abrirá la pantalla correspondiente
Realización	En la pantalla principal de la aplicación se ha colocado un botón para acceder a la pantalla de configuración. En esta pantalla se podrá cambiar tanto el usuario como la contraseña. Una vez realizado este cambio y dependiendo de dichos datos, la aplicación borra o no los datos descargados con anterioridad
Resultado esperado	Cambio correcto de dichos datos y borrado de la base de datos local
Resultado obtenido	Correcto

Tabla 15

PRUEBA 10	
Tarea	Recepción de órdenes desde el servidor
Descripción	Recepción desde el vehículo, de la última orden enviada por el usuario al servidor
Realización	El vehículo envía mediante http y POST su identificador, recibe desde el servidor la última orden que el usuario ha enviado dirigida a dicho vehículo, y la configuración del mismo cambia, reflejando dicha orden.
Resultado esperado	Se reciben los datos de la orden, y tanto la pantalla, como los datos almacenados en las preferencias varían en función de dicha orden.
Resultado obtenido	Correcto

Tabla 16

PRUEBA 11	
Tarea	Cambiar configuración del vehículo
Descripción	Al pulsar el botón de configuración se abrirá la pantalla correspondiente
Realización	En la pantalla principal de la aplicación se ha colocado un botón para acceder a la pantalla de configuración. En esta pantalla se podrá tan solo cambiar el identificador del vehículo.
Resultado esperado	Cambio correcto de dicho identificador
Resultado obtenido	Correcto

Tabla 17

PRUEBA 12	
Tarea	Envío del estado actual al servidor
Descripción	Recolección de los datos del estado actual, y envío de los mismos al servidor
Realización	Cuando el usuario pulse el botón de envío de datos, la aplicación comprobará el estado de cada uno de los controles de la pantalla, y los enviará al servidor mediante HTTP y POST. A cambio recibirá una respuesta positiva en caso de que dichos datos hayan sido correctamente insertados, o una respuesta negativa en el caso de que haya habido algún problema en ello.
Resultado esperado	Inserción correcta de los datos en la base de datos del servidor
Resultado obtenido	Correcto

Tabla 18

PRUEBA 13	
Tarea	Localización del vehículo
Descripción	Adquisición de manera correcta de las coordenadas del vehículo
Realización	Se creó una aplicación sencilla, en la cual el usuario, al pulsar un botón, el sistema debía consultar la localización del mismo. Para ello se comprueban qué fuentes de ubicación tiene el dispositivo activadas, se captura las coordenadas de cada una de ellas, y se comprueba cual de ellas es más precisa. Una vez realizada esta aplicación, se reutilizó el código para integrarlo en la aplicación del vehículo.
Resultado esperado	Coordenadas correctas del vehículo
Resultado obtenido	Correcto

Tabla 19

8 Conclusiones finales

En este apartado del proyecto, se expondrán las conclusiones y retos que ha supuesto la realización de este proyecto por parte del alumno, así como una serie de funcionalidades que han ido surgiendo durante la realización del mismo, pero que por falta de tiempo o por no realizar una replanificación del mismo, se ha decidido finalmente no incluir.

Este apartado se compone de las siguientes secciones.

- Evaluación de objetivos y conclusión
- Tiempo dedicado
- Mejoras y ampliaciones
- Conocimientos adquiridos

8.1 Evaluación de objetivos y conclusión

Antes de la realización de este proyecto, el alumno ya se mostraba interesado en las plataformas móviles, puesto que es la forma más cómoda de llevar la tecnología y las utilidades de software a todas partes. Incluso hace ya unos años, y con conocimientos mucho más limitados, ya intentó realizar sencillas herramientas de software para la plataforma Symbian, con el objetivo de realizar ciertos cálculos sin necesidad de un ordenador completo.

Con esto en mente, y tras la popularización de plataformas móviles como Android e iOS, el alumno ya tenía claro que para el desarrollo de su proyecto fin de carrera iba a enfocarse al desarrollo de aplicaciones móviles. Incluso antes de elegir cual iba a ser su proyecto.

Por estos motivos entre otros, la conclusión de este proyecto se considera un reto muy interesante y con el que el alumno se siente muy realizado.

Si bien es cierto que los objetivos iniciales eran demasiado optimistas, y que durante la realización del mismo, se han rebajado parte de dichos objetivos, también hay que decir que el alcance del mismo ha sido superado ampliamente la idea inicial que tenía el alumno cuando se planteó su ejecución.

De cara al futuro se plantea un futuro realmente incierto, en el cual hay dos plataformas muy fuertes que son Android e iOS, con otras dos con muchísimo menos impacto pero que igualmente hay que tener en cuenta, como son Windows Phone y Blackberry. Además, y por si no fuera suficiente, hay otro sector también muy importante de desarrolladores que apuestan por aplicaciones web (en vez de aplicaciones nativas) multiplataforma. Por tanto, una vez terminado este proyecto, el alumno se enfrenta a una muy difícil decisión. Por un lado puede seguir profundizando en el sistema Android, sistema utilizado para el desarrollo del proyecto, comenzar el aprendizaje de otro sistema (de los destacados anteriormente) o decantarse por tecnologías web estándar.

A pesar de todo esto, el alumno considera que la realización de este proyecto, ha supuesto un ejercicio muy interesante de planificación, estructuración y desarrollo, además de una muy buena lección de aprendizaje. Aprendizaje por otra parte, que puede abrir muchas puertas en el mercado laboral que se plantea, nada más acabar la carrera.

8.2 Tiempo dedicado

Aunque este apartado ya ha sido desarrollado en la revisión del DOP, cabe destacar una serie de conclusiones del mismo.

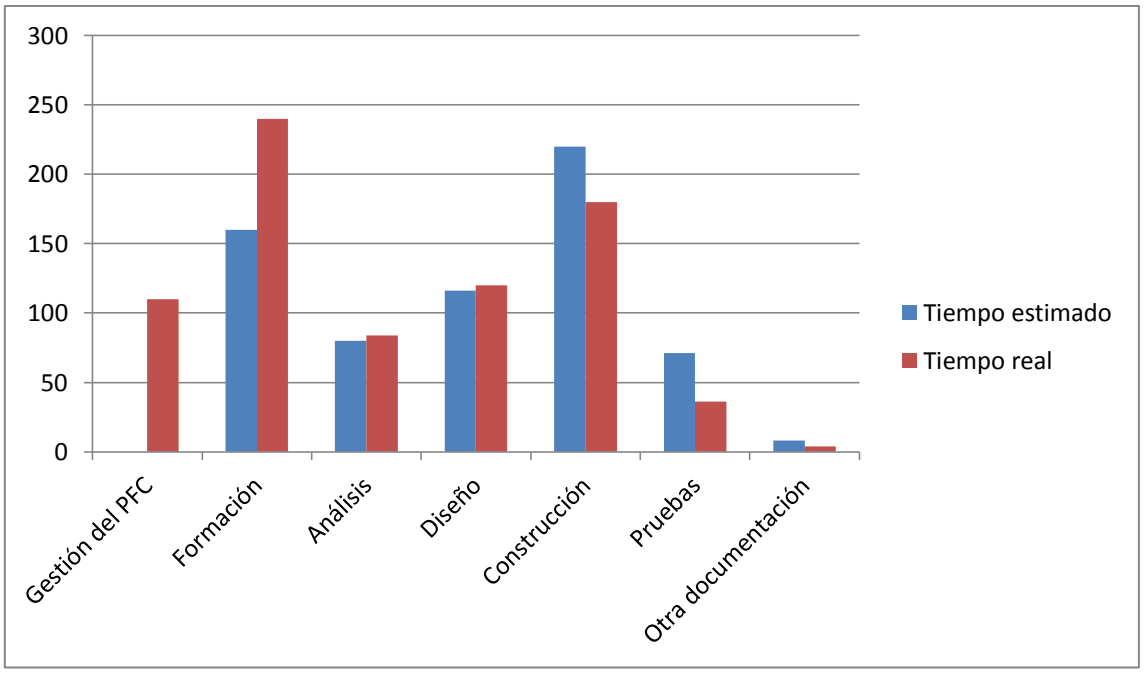


Diagrama 51

Como se puede observar en la gráfica, tanto la fase de Análisis, como la fase de diseño, han superado las previsiones que se habían creado en un principio, aunque no de una forma demasiado pronunciada. Eso no pasa con la fase de formación, ya que el alumno creía que con los conocimientos de Java adquiridos durante la carrera, iba a ser una fase mucho más corta de lo que en realidad ha acabado siendo. De hecho como se puede comprobar, con motivo de las particularidades del sistema operativo (Android) y otros conocimientos que el alumno no tenía, esta se ha convertido en la fase que más ha ocupado en la realización del proyecto.

Sin embargo cabe destacar, que una vez superado estas tres fases, se ha conseguido reducir el tiempo previsto para las fases de construcción y pruebas de forma, notoria. No muy pronunciada pero suficiente como para aligerarlas de forma interesante.

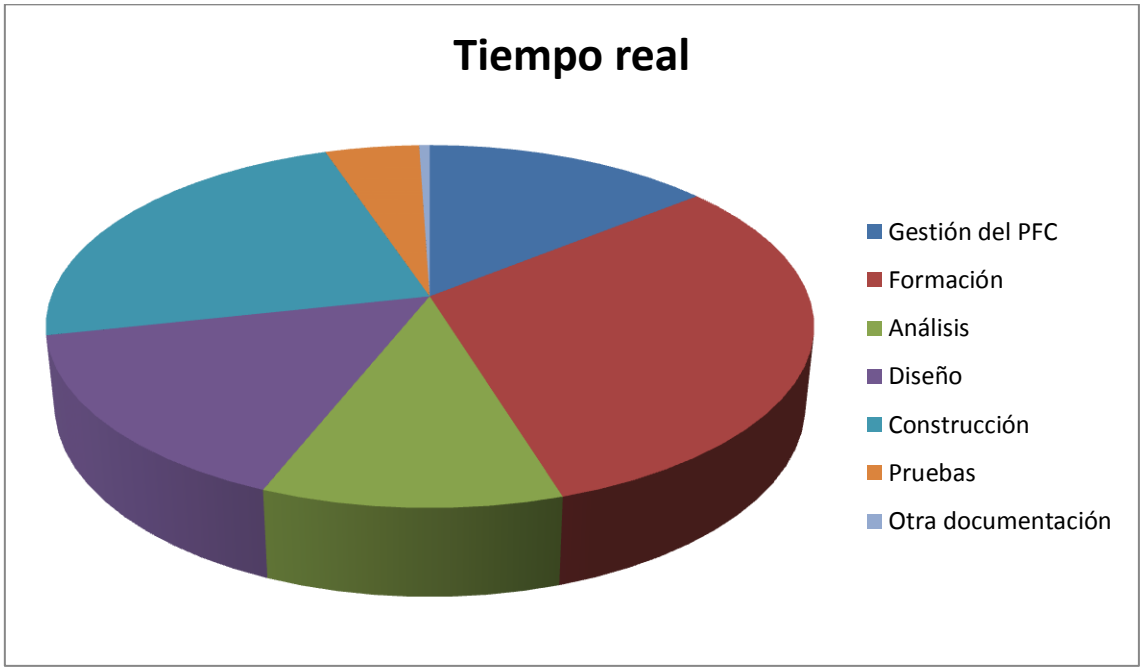


Diagrama 52

En el gráfico anterior se puede comprobar de forma más visual, la diferencia real entre el tiempo invertido en cada una de las etapas.

Cabe destacar que el tiempo indicado en la fase "Otra documentación" no corresponde a la documentación total del proyecto, puesto que esta está ya incluida en las diferentes etapas. Esta sólo representa a la documentación extra, que no había sido incluida en ninguna de las anteriores etapas.

8.3 Mejoras y ampliaciones

Como ya se ha comentado con anterioridad, durante el desarrollo del proyecto han ido surgiendo ideas que han sido integradas en el mismo, y que por tanto forman parte de las fases de análisis, diseño y construcción del mismo. Sin embargo hay otra serie de ideas se han descartado por no suponer una reestructuración demasiado grande del mismo, o por no disponer del tiempo deseado para implementarlas.

A continuación se destacan algunas de ellas.

- Notificaciones instantáneas al usuario
- Filtrado por vehículo en la pantalla principal

- Vista de mapa con el recorrido seguido por un vehículo concreto (mostrando los últimos 20 estados, por ejemplo)
- Automatización del envío del estado del vehículo (esta no ha sido por falta de tiempo, si no por la imposibilidad de realizarlo con una simulación del mismo)
- Medidas de seguridad más exhaustivas
- Mayor personalización en la interfaz
- Mayor descripción para el usuario, de los fallos durante su uso. Por ejemplo, un aviso más descriptivo en caso de que el servidor no haya almacenado correctamente los datos enviados desde uno de los dispositivos

8.4 Conocimientos adquiridos

Si bien ya había participado en la creación y desarrollo de un proyecto desde cero en la asignatura de Proyectos, la experiencia de realizarlo sobre una plataforma real y con resultados visibles, ha sido más intensa y estimulante. Gracias a ello, el alumno se ha adquirido un mayor grado de conocimiento en el desarrollo y gestión de proyectos.

Además de esto, conocer un poco más la programación en Android me hace ver todo el camino que me queda por recorrer pero lo satisfactorio que puede resultar. En caso de ser posible, sería muy gratificante para el alumno, poder continuar aumentando los conocimientos en este campo y ampliarlos con sistemas móviles diferentes.

También destacar, que la realización de este proyecto, ha resultado de un gran interés para el alumno, por motivos como:

- Ampliar y poner en práctica conocimientos adquiridos durante la carrera
- Aprender los pasos para la realización de forma correcta, de un proyecto de una entidad considerable.
- Conocer los mecanismos de comunicación cliente-servidor y las tecnologías XML y JSON
- Estudiar y comprender todo lo relacionado con las tecnologías móviles y en particular Android.
- Mejorar las capacidades de auto aprendizaje, y la búsqueda de recursos en diferentes medios (libros, web, etc.)

Y ya para finalizar y en modo de resumen, cabría indicar que si bien se trata de una apelación con una interfaz sencilla, hace falta una aproximación más profunda de la

misma para poder identificar todos los conocimientos de diversas disciplinas que ha debido asimilar el alumno para conclusión del proyecto.

9 Bibliografía

9.1 Libros

- Ableson F., Collins C., and Sen R., 2009. Unlocking Android A Developer's Guide. Manning publications Co.
- Burnette E., 2009. Hello, Android: Introducing Google's Mobile Development Platform. 3rd ed. Pragmatic Bookshelf
- Murphy M., 2001. Beginning Android 3. Apress

9.2 Recursos web

- The official site for Android developers. [online] Disponible en: <http://developer.android.com/index.html> [Última consulta 3 de Junio 2013]
- Curso de Programación Android | sgoliver.net blog. [online] Disponible en: <http://www.sgoliver.net/blog/?page_id=2935> [Última consulta 26 de Mayo 2013]
- Blog de Javielinux [online] Disponible en: <http://www.javielinux.com/> [Última consulta 13 de Marzo 2013]
- Blog Androideity [online] Disponible en: <http://androideity.com/> [Última consulta 25 de Mayo 2013]
- Foro Stackoverflow [online] Disponible en: <http://stackoverflow.com/> [Última consulta 2 de Junio 2013]
- Android Developers Blog [online] Disponible en: <http://android-developers.blogspot.com/> [Última consulta 5 de Marzo 2013]
- Blog de Cyril Mottier [online] Disponible en: <http://cyrilmottier.com/> [Última consulta 17 de Abril 2013]
- Comunidad Android Development en Google+ [online] Disponible en: <https://plus.google.com/communities/105153134372062985968/stream/7db53d71-5bab-46fd-a3f2-a321fee334af> [Última consulta 7 de Junio 2013]
- Foro DesarrolloWeb [online] Disponible en: <http://www.desarrolloweb.com/> [Última consulta 20 de Abril 2013]
- Lista de correo Android Developers en Google Groups [online] Disponible en: <https://groups.google.com/forum/#!forum/android-developers> [Última consulta 13 de Mayo 2013]
- Lista de correo desarrolladores-android en Google Groups [online] Disponible en: <https://groups.google.com/forum/#!forum/desarrolladores-android> [Última consulta 9 de Mayo 2013]

10 Anexos

En este apartado, se incluirán tanto la documentación que no ha tenido cabida en ninguno de los apartados anteriores, como aquellas imágenes o gráficos que por culpa del espacio que necesitan, se han ido añadiendo de forma reducida.

Las secciones son las siguientes.

- Otra documentación
- Actas de reunión

10.1 Otra documentación

10.1.1 Arquitectura de Android

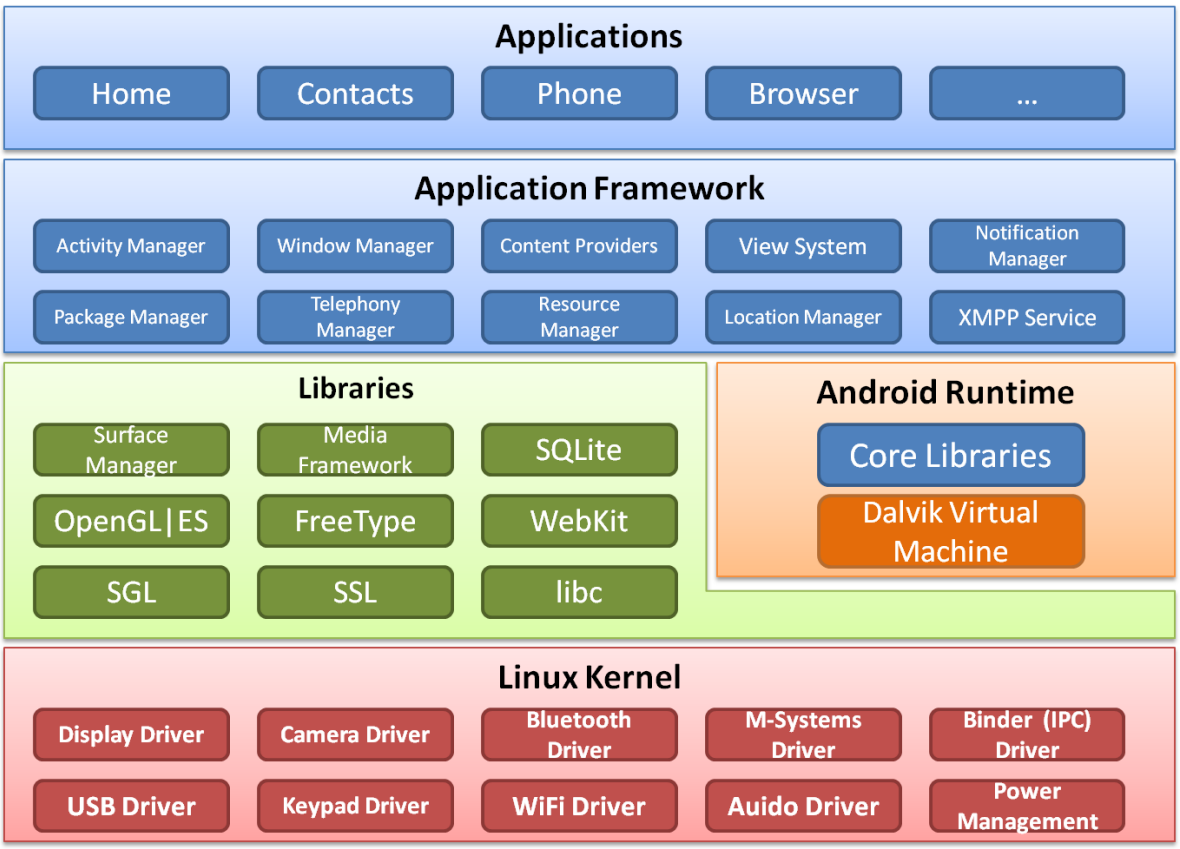


Diagrama 53

En la imagen se observa la estructura general del sistema Android. Este se subdivide en las siguientes 5 capas

10.1.1.1 Linux Kernel:

El núcleo del sistema operativo Android está basado en el kernel de Linux, adaptado a las características del hardware en el que se ejecutará Android. Este actúa como una capa de abstracción entre el hardware y el resto de las capas de la arquitectura. Además también se encarga de gestionar los diferentes recursos del teléfono (energía, memoria, etc.) y del sistema operativo.

10.1.1.2 Libraries:

Esta capa, que se sitúa justo sobre el kernel, la componen las bibliotecas nativas de Android, también llamadas librerías. Escritas y compiladas para la arquitectura hardware específica del teléfono. El objetivo de dicha capa, es proporcionar funcionalidad a las aplicaciones para tareas que se repiten con frecuencia, evitando

tener que codificarlas cada vez y garantizando que se llevan a cabo de la forma más eficiente eficiente posible.

10.1.1.3 Android Runtime:

Como se aprecia en el diagrama, el entorno de ejecución de Android no es considerada una capa en sí misma, dado que también está formado por librerías. Aquí se encuentran las librerías con la funcionalidades habituales de Java así como otras específicas de Android.

El componente principal del entorno de ejecución de Android es la máquina virtual Dalvik. Las aplicaciones se codifican en Java y son compiladas en un formato específico para que esta máquina virtual las ejecute.

10.1.1.4 Application Framework:

La siguiente capa está formada por todas las clases y servicios que utilizan directamente las aplicaciones para realizar sus funciones. La mayoría de los componentes de esta capa son librerías Java que acceden a los recursos de las capas anteriores a través de la máquina virtual Dalvik. En dicha capa se encuentran los siguientes módulos principales:

- Activity Manager. Se encarga de administrar la pila de actividades de nuestra aplicación así como su ciclo de vida.
- Windows Manager. Se encarga de organizar lo que se mostrará en pantalla. Básicamente crea las superficies en la pantalla que posteriormente pasarán a ser ocupadas por las actividades.
- Content Provider. Esta librería es muy interesante porque crea una capa que encapsula los datos que se compartirán entre aplicaciones para tener control sobre cómo se accede a la información.
- Views. En Android, las vistas los elementos que nos ayudarán a construir las interfaces de usuario: botones, cuadros de texto, listas y hasta elementos más avanzados como un navegador web o un visor de Google Maps.
- Notification Manager. Engloba los servicios para notificar al usuario cuando algo requiera su atención mostrando alertas en la barra de estado. Un dato importante es que esta biblioteca también permite jugar con sonidos, activar el vibrador o utilizar los LEDs del teléfono en caso de tenerlos.
- Package Manager. Esta biblioteca permite obtener información sobre los paquetes instalados en el dispositivo Android, además de gestionar la instalación de nuevos paquetes. Con paquete nos referimos a la forma en que se distribuyen las aplicaciones Android, estos contienen el archivo .apk, que a

su vez incluyen los archivos .dex con todos los recursos y archivos adicionales que necesite la aplicación, para facilitar su descarga e instalación.

- Telephony Manager. Con esta librería podremos realizar llamadas o enviar y recibir SMS/MMS, aunque no permite reemplazar o eliminar la actividad que se muestra cuando una llamada está en curso.
- Resource Manager. Con esta librería podremos gestionar todos los elementos que forman parte de la aplicación y que están fuera del código, es decir, cadenas de texto traducidas a diferentes idiomas, imágenes, sonidos o layouts. En un post relacionado a la estructura de un proyecto Android veremos esto más a fondo.
- Location Manager. Permite determinar la posición geográfica del dispositivo Android mediante GPS o redes disponibles y trabajar con mapas.
- Sensor Manager. Nos permite manipular los elementos de hardware del teléfono como el acelerómetro, giroscopio, sensor de luminosidad, sensor de campo magnético, brújula, sensor de presión, sensor de proximidad, sensor de temperatura, etc.
- Cámara: Con esta librería podemos hacer uso de la(s) cámara(s) del dispositivo para tomar fotografías o para grabar vídeo.
- Multimedia.Permiten reproducir y visualizar audio, vídeo e imágenes en el dispositivo.

10.1.1.5 Applications

En la última capa se incluyen todas las aplicaciones del dispositivo, tanto las que tienen interfaz de usuario como las que no, las nativas (programadas en C o C++) y las administradas (programadas en Java), las que vienen preinstaladas en el dispositivo y aquellas que el usuario ha instalado.

En esta capa encontramos también la aplicación principal del sistema: Inicio (Home) o lanzador (launcher), porque es la que permite ejecutar otras aplicaciones mediante una lista y mostrando diferentes escritorios donde se pueden colocar accesos directos a aplicaciones o incluso widgets, que son también aplicaciones de esta capa.

10.1.2 Código de la aplicación

10.1.2.1 CarStatus.java

```
package com.demo.carmanagerclient.models;

public class CarStatus {
    private String id_status;
    private long date_status_car;
    private long date_status_server;
    private String id_car;
    private double latitude;
    private double longitude;
    private boolean locked_car;
    private int status_gas;
    private boolean front_left_window;
    private boolean front_right_window;
    private boolean back_left_window;
    private boolean back_right_window;
    private boolean front_left_door;
    private boolean front_right_door;
    private boolean back_left_door;
    private boolean back_right_door;
    private boolean lights_on;
    private int status_battery;
    private int front_left_wheel;
    private int front_right_wheel;
    private int back_left_wheel;
    private int back_right_wheel;
    private int status_oil;
    private double odometer_km;

    public String getId_status() {
        return id_status;
    }
    public void setId_status(String id_status) {
        this.id_status = id_status;
    }
    public long getDate_status_car() {
        return date_status_car;
    }
    public void setDate_status_car(long date_status_car) {
        this.date_status_car = date_status_car;
    }
    public long getDate_status_server() {
        return date_status_server;
    }
    public void setDate_status_server(long date_status_server) {
        this.date_status_server = date_status_server;
    }
    public String getId_car() {
        return id_car;
    }
    public void setId_car(String id_car) {
        this.id_car = id_car;
    }
    public double getLatitude() {
```

Diagrama 54

Como ya se indicó, esta es la clase que representa el estado que el vehículo envía al servidor y por tanto al usuario.

10.1.2.2 Order.java

```
package com.demo.carmanagerclient.models;

public class Order {
    private long date_order;
    private String id_car;
    private boolean lock_car;
    private boolean front_left_window;
    private boolean front_right_window;
    private boolean back_left_window;
    private boolean back_right_window;
    private boolean front_left_door;
    private boolean front_right_door;
    private boolean back_left_door;
    private boolean back_right_door;
    private boolean switch_lights_on;

    public long getDate_order() {
        return date_order;
    }
    public void setDate_order(long date_order) {
        this.date_order = date_order;
    }
    public String getId_car() {
        return id_car;
    }
    public void setId_car(String id_car) {
        this.id_car = id_car;
    }
    public boolean isLock_car() {
        return lock_car;
    }
    public void setLock_car(boolean lock_car) {
        this.lock_car = lock_car;
    }
    public boolean isFront_left_window() {
        return front_left_window;
    }
    public void setFront_left_window(boolean front_left_window) {
        this.front_left_window = front_left_window;
    }
    public boolean isFront_right_window() {
        return front_right_window;
    }
    public void setFront_right_window(boolean front_right_window) {
        this.front_right_window = front_right_window;
    }
    public boolean isBack_left_window() {
        return back_left_window;
    }
    public void setBack_left_window(boolean back_left_window) {
        this.back_left_window = back_left_window;
    }
    public boolean isBack_right_window() {
```

Diagrama 55

Por otra parte, esta es la clase que representa la orden que el usuario le envía al vehículo.

10.1.2.3 status_view.xml

```
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/ScrollView01"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:focusable="false"
    tools:context=".MainActivity"
    android:scrollbars="none">

    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:padding="20dip" >

        <RelativeLayout
            android:id="@+id/layout_id_date"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentLeft="true"
            android:layout_alignParentTop="true" >

            <TextView
                android:id="@+id/id_status"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignParentLeft="true"
                android:layout_alignParentTop="true"
                android:text="ID_STATUS"
                android:textAppearance="?android:attr/textAppearanceLarge" />

            <TextView
                android:id="@+id/id_car"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignBottom="@+id/id_status"
                android:layout_alignParentRight="true"
                android:text="ID_CAR"
                android:textAppearance="?android:attr/textAppearanceLarge" />

            <TextView
                android:id="@+id/date_car"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_alignLeft="@+id/id_status"
                android:layout_below="@+id/id_status"
                android:text="DATE_CAR"
                android:textAppearance="?android:attr/textAppearanceLarge" />

            <TextView
                android:id="@+id/odometer"
                android:layout_width="wrap_content"
```

Diagrama 56

Este es un fragmento del código XML en el que se especifica el diseño que ha de tener la pantalla en la que se muestra el estado del vehículo, en el dispositivo del usuario.

10.1.2.4 MyAsyncTask.java

```
package com.demo.carmanagerclient;

import org.json.JSONArray;

class MyAsyncTask extends AsyncTask<JsonParser, Integer, JSONArray> {

    private static String LogDavid = "LogDavid";

    @Override
    protected JSONArray doInBackground(JsonParser... parser) {
            JSONArray status = parser[0].getServerData();
            return status;
    }

}
```

Diagrama 57

Las clases que heredan de la clase AsyncTask, se utilizan para lanzar procesos asíncronos en segundo plano, sin que estos afecten al funcionamiento de la aplicación.

10.2 Actas de reunión

10.2.1 Acta 1

Lugar: Despacho 113 del edificio departamental

Fecha: 26 de Septiembre 2011

Hora: 10:30

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de la propuesta del PFC

Desarrollo
Propuesta por parte del alumno, del desarrollo de un sistema de geolocalización de vehículos. Se acuerda la realización de un documento inicial en el que quedará detallado el alcance del mismo.

Conclusión
Se aprueba la realización de dicho proyecto.

10.2.2 Acta 2

Lugar: Despacho 113 del edificio departamental

Fecha: 20 de Noviembre 2012

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación del alcance

Desarrollo
Se retoma el proyecto después de una temporada en el aire. Se corrige y aprueba el documento en el que se detalla el alcance del proyecto y se realiza una estimación inicial de tiempo

Conclusión
Se aprueba el documento realizado y se establece fecha para la siguiente reunión.

10.2.3 Acta 3

Lugar: Despacho 113 del edificio departamental

Fecha: 3 de Diciembre 2012

Hora: 12:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación del DOP

Desarrollo
Puesta en común del DOP, y corrección de errores del mismo.

Conclusión
Se establece nueva fecha para entrega del DOP y nueva reunión

10.2.4 Acta 4

Lugar: Despacho 113 del edificio departamental

Fecha: 12 de Diciembre 2012

Hora: 10:30

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación del DOP

Desarrollo
Una vez corregido el DOP se aprueba. Además se establece la metodología a seguir para la realización del proyecto.

Conclusión
Se establece fecha para la siguiente reunión

10.2.5 Acta 5

Lugar: Despacho 113 del edificio departamental

Fecha: 20 de Diciembre 2010

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Comprobación de avance del proyecto

Desarrollo
Reunión para poner en común el avance tanto en la documentación como en el desarrollo, del proyecto.

Conclusión
Se establece fecha para la siguiente reunión

10.2.6 Acta 6

Lugar: Despacho 113 del edificio departamental

Fecha: 23 de Diciembre 2013

Hora: 11:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Replanificación del PFC

Desarrollo
Se modifica ligeramente el alcance del mismo y se establece una nueva previsión

Conclusión
Se establece fecha para la siguiente reunión

10.2.7 Acta 7

Lugar: Despacho 113 del edificio departamental

Fecha: 6 de Enero 2013

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Análisis

Desarrollo
Puesta en común de los diagramas de casos de uso, los distintos roles que actúan en el sistema y una aproximación inicial de los diagramas de actividad.

Conclusión
Se acuerda la modificación de algunos de ellos y se establece fecha para la siguiente reunión

10.2.8 Acta 8

Lugar: Despacho 113 del edificio departamental

Fecha: 10 de Enero 2013

Hora: 13:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación del análisis

Desarrollo
Con los diagramas modificados a instancia del director, se aprueban y se comienza con la fase de diseño

Conclusión
Se aprueban y se establece fecha para la siguiente reunión

10.2.9 Acta 9

Lugar: Despacho 113 del edificio departamental

Fecha: 27 de Enero 2013

Hora: 9:30

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Puesta en común del avance del diseño

Desarrollo
Se acuerda tanto la arquitectura del sistema como el diseño de la base de datos del servidor

Conclusión
Se establece fecha para la siguiente reunión

10.2.10 Acta 10

Lugar: Despacho 113 del edificio departamental

Fecha: 5 de Febrero 2013

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de las pantallas de la aplicación.

Desarrollo
Puesta en común y aprobación tanto de las pantallas de la aplicación, como de la funcionalidad de cada una de ellas.

Conclusión
Se aprueban y se establece fecha para la siguiente reunión

10.2.11 Acta 11

Lugar: Despacho 113 del edificio departamental

Fecha: 13 de Febrero 2013

Hora: 10:30

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Seguimiento

Desarrollo
Puesta en común del avance de la fase de diseño

Conclusión
Se establece fecha para la siguiente reunión

10.2.12 Acta 12

Lugar: Despacho 113 del edificio departamental

Fecha: 16 de Febrero 2013

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación del diseño de las clases

Desarrollo
Puesta en común de las clases identificadas inicialmente para el desarrollo de la aplicación y aprobación de las mismas.

Conclusión
Se aprueba y se establece fecha para la siguiente reunión

10.2.13 Acta 13

Lugar: Despacho 113 del edificio departamental

Fecha: 24 de Febrero 2013

Hora: 12:30

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de la fase de diseño

Desarrollo
Puesta en común del documento generado durante la fase de diseño, y se acuerdan ciertas modificaciones sobre el mismo.

Conclusión
Se establece fecha para la siguiente reunión

10.2.14 Acta 14

Lugar: Despacho 113 del edificio departamental

Fecha: 27 de Febrero 2013

Hora: 12:30

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de la fase de diseño

Desarrollo
Una vez realizadas las modificaciones propuestas, se da por aprobada dicha fase y se da comienzo a la fase de implementación o codificación del sistema

Conclusión
Se aprueba y se establece fecha para la siguiente reunión

10.2.15 Acta 15

Lugar: Despacho 113 del edificio departamental

Fecha: 3 de Marzo 2013

Hora: 9:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Seguimiento

Desarrollo
Puesta en común de las metas alcanzadas hasta el momento

Conclusión
Se establece fecha para la siguiente reunión

10.2.16 Acta 16

Lugar: Despacho 113 del edificio departamental

Fecha: 10 de Marzo 2013

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Seguimiento

Desarrollo
Puesta en común de las metas alcanzadas hasta el momento

Conclusión
Se establece fecha para la siguiente reunión

10.2.17 Acta 17

Lugar: Despacho 113 del edificio departamental

Fecha: 23 de Marzo 2013

Hora: 11:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Seguimiento

Desarrollo
Puesta en común de las metas alcanzadas hasta el momento y propuesta de modificación de algunos comportamientos

Conclusión
Se establece fecha para la siguiente reunión

10.2.18 Acta 18

Lugar: Despacho 113 del edificio departamental

Fecha: 30 de Marzo 2013

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Seguimiento

Desarrollo
Puesta en común de las metas alcanzadas hasta el momento

Conclusión
Se establece fecha para la siguiente reunión

10.2.19 Acta 19

Lugar: Despacho 113 del edificio departamental

Fecha: 10 de Abril 2013

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de la fase de construcción

Desarrollo
Puesta en común de la fase de construcción y detección de errores

Conclusión
Se establece fecha para la siguiente reunión

10.2.20 Acta 20

Lugar: Despacho 113 del edificio departamental

Fecha: 13 de Abril 2013

Hora: 11:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de la fase de construcción

Desarrollo
Una vez corregidos los errores detectados en la anterior reunión, se aprueba dicha fase y se establecen las pruebas a realizar.

Conclusión
Se aprueba y se establece fecha para la siguiente reunión

10.2.21 Acta 21

Lugar: Despacho 113 del edificio departamental

Fecha: 25 de Abril 2013

Hora: 12:30

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de la fase de pruebas

Desarrollo
Puesta en común de las pruebas y los resultados de las mismas.

Conclusión
Se establece fecha para la siguiente reunión

10.2.22 Acta 22

Lugar: Despacho 113 del edificio departamental

Fecha: 20 de Mayo 2013

Hora: 10:00

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de la fase de conclusiones

Desarrollo
Lectura de las conclusiones y aprobación de las mismas

Conclusión
Se establece fecha para la siguiente reunión

10.2.23 Acta 23

Lugar: Despacho 113 del edificio departamental

Fecha: 26 de Junio 2013

Hora: 12:30

Asistentes
David Ruiz Urraca Francisco Javier Martínez de Pisón Ascacíbar

Orden del día
Aprobación de la memoria

Desarrollo
Puesta en común de la memoria definitiva del PFC y aprobación de la misma

Conclusión
Se establece el procedimiento a seguir para su presentación y defensa.