TRABAJO FIN DE GRADO

TECONVIDO: APLICACIÓN DE CAR-SHARING PARA LA PROVINCIA DE TERUEL

Alejandro Silva Sanahuja

Director:
Francisco J. Martínez Domínguez

11 de septiembre de 2015

Resumen

En la actualidad, en la provincia de Teruel, existe un gran problema a la hora de moverse entre los municipios de la provincia motivada por una carencia de transporte público. Este problema afecta desde a personas de anciana edad que necesitan ir a Teruel para visitar a especialistas médicos, como en la población más joven que necesitan desplazarse para ir al instituto o universidad.

Para solucionar la situación anterior, surge TeConvido, una aplicación en un entorno de red social, para el uso compartido del automóvil con otras personas o como se le conoce en su término inglés, *Carpooling*.

El sistema está formado por dos aplicaciones, una web y otra Android, con el fin de abarcar el máximo rango de dispositivos y equipos con el que sea compatible. El propósito fundamental es poner en contacto a conductores y pasajeros con preferencias de viajes iguales o compatibles.

La plataforma una vez implementada permitirá a tanto conductores como pasajeros ahorrar tanto dinero como tiempo en los desplazamientos dentro de la provincia de Teruel, así como la práctica de una conducción más ecológica motivada por el descenso del número de vehículos en circulación, y por tanto, una reducción de la emisión de gases a la atmósfera.

Palabras clave: Carpooling, carsharing, red social, web, android.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introdución

1.1. Motivación

Poco a poco los vehículos se han ido convirtiendo en algo casi imprescindible a la hora de desplazarse por la libertad y comodidad que nos proporcionan, hasta el punto que se utilizan para los desplazamientos tan comunes como ir a trabajar o a estudiar. Esta necesidad está provocando que haya cada vez más vehículos en circulación, y por tanto, hayan aumentado los problemas asociados como las emisiones de carbono a la atmósfera, las congestiones de tráfico o consumo de combustible.

Para lidiar con estos inconvenientes han surgido soluciones como compartir coche, también conocido por su término en inglés *Carpool*. Esta idea surgió en Estados Unidos durante la Segunda Guerra Mundial como una táctica de racionamiento y volvió en la década de los 70 debido a la crisis del petróleo. Durante 30 años se redujo esta práctica, pero con Internet y posteriormente con la popularidad de los smartphones, se ha facilitado el crecimiento del *carpooling* y el incremento de sitios web para compartir coche.

TeConvido, la aplicación desarrollada en el presente Trabajo Fin de Grado (TFG), nace bajo esta idea para el caso específico de la provincia de Teruel. Este Trabajo parte de la necesidad de moverse entre poblaciones de esta provincia motivada por la carencia generalizada de transporte público y las necesidades de sus habitantes, por ejemplo, las personas de anciana edad que necesitan ir a Teruel para ir al médico.

El proyecto se dirigirá al sector del transporte en la provincia de Teruel, más en concreto a la modalidad *Carpool* citada anteriormente, en la cual un conductor con plazas libres encuentre ocupantes que vayan al mismo destino que él. Se espera facilitar a través de esta plataforma la comunicación de personas con una compatibilidad en el origen (población de origen o poblaciones que se desvíen poco del itinerario) que quieran desplazarse a un destino concreto y en una fecha determina-

da. El servicio de esta plataforma será gratuito y se planteará como una red social.

La plataforma no busca que los conductores que se ofrezcan a realizar un viaje obtengan beneficio de éste, sino que solamente reduzcan su gasto.

Las características del servicio son:

- Barato. El coste del viaje se abarata ya que se comparte los gastos entre los ocupantes del vehículo. Más barato incluso que cualquier transporte público.
- Ecológico. El hecho de facilitar que conductores y pasajeros se pongan de acuerdo, consigue aprovechar un viaje para más de una persona, y por tanto, facilita que no haya tantas emisiones de gases a la atmósfera.
- Orientado a una región. El hecho de centrarse en una región y que en esta región no exista nada igual, facilita su repercusión y aumenta la utilidad de este servicio.

El motivo de mi elección se debe a intentar paliar una necesidad en las poblaciones de la provincia de Teruel de la cual, no hay nada concreto hecho al respeto y también por el bien social que puede hacer esta plataforma a la hora de facilitar un transporte a quienes por carencia de él u otros motivos, le sea difícil desplazarse a otras localidades.

A nivel personal, mi trabajo fin de grado me va a posibilitar poder aprender y profundizar en lenguajes de programación web, desde programación en el lado del servidor como es el caso de PHP, hasta lenguajes de marcado y estilo como HTML y CSS, del cual partía prácticamente de cero. También afianzaré mis conocimientos en Android y MySQL al realizar una aplicación mucho más compleja que las que había realizado hasta ahora. Por último, me va a permitir aprender a interoperar entre varios lenguajes de programación como si te tratase del mismo a través de *JavaScript Object Notation*(JSON).

1.2. Objetivos

El objetivo principal de este proyecto es la creación de una plataforma para compartir coche en la provincia de Teruel que abarque el mayor mercado posible. Para ello se va a realizar una aplicación tanto web, como móvil. Esta aplicación pondrá en comunión a conductores y pasajeros que tengan orígenes y destinos comunes, o en su defecto, esté uno de ellos o ambos en la ruta del conductor.

Los objetivos secundarios serían los siguientes:

- Diseñar una estructura conforme a una red social para facilitar la comunicación directa y rápida entre los usuarios de la plataforma.
- La creación de una plataforma fácil de utilizar, con una apariencia simple y cómoda para el usuario.

- Realizar un diseño de la plataforma totalmente modulable ante posibles actualizaciones o implementaciones adicionales.

- Realizar una implementación segura para proteger los datos de los usuarios ante ataques de hacking.

- Desarrollar un protocolo de interoperabilidad entre la parte web y la parte móvil.

1.3. Visión general del documento

El presente documento consta de los apartados indicados a continuación:

- **Conceptos previos**. Apartado en el que se contextualiza la aplicación y se exponen conceptos básicos, necesarios para entender el funcionamiento de la misma.

- **Estado del arte**. Se expondrán diferentes aplicaciones presentes en el mercado que ofrecen el mismo servicio o servicios complementarios al trabajo fin de grado que estamos tratando, presentando al finalizar una tabla comparativa de las principales aplicaciones para que se puedan visualizar de forma clara las características más importantes.

- **Análisis**. En este apartado se desarrollan todas las condiciones que debe satisfacer la aplicación,siguiendo en todo momento las directrices que se marcan en el estándar del *IEEE Recommended Practice for Software Requirementes Specifications IEEE 830-1998* [1]

- **Planificación y estimación de costes**. Se detallará tanto la planificación prevista, como la real del proyecto. A partir de estos datos se realizará una estimación de los costes del desarrollo del proyecto.

- **Diseño**. Sección en la que se expone y explica detalladamente toda la fase de diseño de la aplicación.

- **Conclusiones**. Apartado para la reflexión y el análisis sobre la realización del proyecto y los resultados obtenidos.

- **Bibliografía**. Se incluyen las referencias bibliográficas utilizadas a lo largo de la redacción del presente proyecto.

- **Anexos**. En esta sección se incluye todo aquella información complementaria sobre los diferentes apartados del proyecto que no ha tenido cabida en los apartados mencionados con antelación.

1.4. Agradecimientos

En primer lugar quería agradecer a mi familia con la cual siempre he podido contar a la hora de pedir ayuda o consejo, y me han dado apoyo tanto a lo largo de la carrera, como en el propio proyecto.

También quería agradecer a mi director Francisco J. Martínez por proponerme este proyecto, darme rienda suelta para hacer y deshacer como he querido, y por todo el apoyo y la ayuda recibida. También a todos los componentes de grupo INIT con los he compartido un magnífico ambiente de trabajo y siempre estaban dispuestos a echarme un cable cuando lo he necesitado.

Por último lugar, agradecer a toda la comunidad de desarrolladores que se molestan en hacer manuales o tutoriales, librerías de código de software libre, y un largo etc., con los que una persona con ganas de aprender, puede hacer cualquier cosa que se proponga.

Capítulo 2

Conceptos previos

En este apartado se va a detallar las tecnologías utilizadas para la realización del proyecto. En primer lugar, hablaré de los lenguajes de programación utilizados en la parte web de la aplicación que son PHP, HTML, CSS y Javascript. Debido a su relevancia a la hora de realizar esta parte, también incluiré una descripción de la biblioteca de Javascript, jQuery, y de JSON, un formato ligero de intercambio de datos. En segundo lugar, detallaré el sistema operativo Android que es el elegido para la parte móvil de la aplicación. En tercer lugar, se expondrá el sistema gestor de base de datos MySQL. Y en último lugar, se detallarán algunos servicios de Google que han sido esenciales para la realización del proyecto.

2.1. PHP

Creado originalmente por Rasmus Lerdorf en 1995, es un lenguaje de programación de software libre de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico [2]. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página web resultante. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

PHP se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocidos hasta el día de hoy, lo que ha atraído el interés de múltiples sitios con gran demanda de tráfico, como Facebook, para optar por el mismo como tecnología de servidor.

2.2. HTML

HTML, siglas de *HyperText Markup Language*, hace referencia al lenguaje de marcado para la elaboración de páginas web [3]. Es un estándar que sirve de referencia para la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código para la definición de contenido de una página web, como texto, imágenes y vídeos, entre otros. Es un estándar a cargo de la W3C, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación.

El lenguaje HTML basa su filosofía de desarrollo en la referenciación. Para añadir un elemento externo a la página (imagen, vídeo, script, etc.), éste no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene sólo texto mientras que recae en el navegador web la tarea de unir todos los elementos y visualizar la página final. Al ser un estándar, HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma por cualquier navegador web actualizado.

2.3. CSS

CSS es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas [4].

Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos HTML/XHTML bien definidos y con significado completo. Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro de la página.Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página, etc.

2.4. Javascript

JavaScript, abreviado comúnmente "JS", es un lenguaje de programación interpretado, dialecto del estándar ECMAScript [5]. Se define como orientado a objetos, basado en prototipos, imperativo, débilmente tipado y dinámico.

Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

JavaScript se diseñó con una sintaxis similar al C, aunque adopta nombres y convenciones del lenguaje de programación Java. Sin embargo Java y JavaScript no están relacionados y tienen semánticas y propósitos diferentes.

Todos los navegadores modernos interpretan el código JavaScript integrado en las páginas web. Para interactuar con una página web se provee al lenguaje JavaScript de una implementación del Document Object Model (DOM).

2.5. jQuery

jQuery es una biblioteca de JavaScript, creada inicialmente por John Resig, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica AJAX a páginas web [6]. Fue presentada el 14 de enero de 2006 en el BarCamp NYC. jQuery es la biblioteca de JavaScript más utilizada.

jQuery es software libre y de código abierto, posee un doble licenciamiento bajo la Licencia MIT y la Licencia Pública General de GNU v2, permitiendo su uso en proyectos libres y privados. jQuery, al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra manera requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

2.6. JSON

JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos [7]. Está basado en un subconjunto del Lenguaje de Programación JavaScript. JSON es un formato de texto que es completamente independiente del lenguaje, pero utiliza convenciones que son ampliamente conocidas por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituído por dos estructuras: una colección de pares de nombre/valor y una lista ordenada de valores. Estas son estructuras universales, todos los lenguajes de programación las soportan de una forma u otra.

2.7. Android

Android es un sistema operativo basado en el núcleo Linux, diseñado principalmente para dispositivos móviles con pantalla táctil, como teléfonos inteligentes

o tablets [8]. También está siendo implementado en otros sistemas como relojes inteligentes, televisores y automóviles. Inicialmente fue desarrollado por Android Inc., empresa que Google respaldó económicamente y más tarde, en 2005, compró. Android fue presentado en 2007 junto la fundación del Open Handset Alliance (un consorcio de compañías de hardware, software y telecomunicaciones) para avanzar en los estándares abiertos de los dispositivos móviles.

Android es multitarea y permite a los desarrolladores acceder a las funcionalidades principales del dispositivo mediante aplicaciones, que pueden ser reemplazadas por otras oficiales o de terceros desarrolladas a través de las herramientas proporcionadas por Google. El sistema consta de 12 millones de líneas de código escritas en XML, C, C++ y Java. La mayoría de dicho código se libera bajo la licencia Apache, lo que permite a la comunidad de desarrolladores realizar cambios y mejoras en el sistema operativo.

Las aplicaciones para Android se desarrollan habitualmente en el lenguaje Java con Android Software Development Kit (Android SDK), pero están disponibles otras herramientas de desarrollo.Todas las aplicaciones están comprimidas en formato APK, que se pueden instalar sin dificultad desde cualquier explorador de archivos en la mayoría de dispositivos.

2.8. MySQL

MySQL [9] es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB, desde enero de 2008, una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009, desarrolla MySQL como software libre en un esquema de licenciamiento dual. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL está patrocinado por una empresa privada, que posee el copyright de la mayor parte del código. MySQL AB fue fundada por David Axmark, Allan Larsson y Michael Widenius.

MySQL es usado por muchos sitios web grandes y populares, como Wikipedia, Google, Facebook, Twitter, Flickr y YouTube.

2.9. Servicios de Google

2.9.1. Google Cloud Messaging para Android (GCM)

Google Cloud Messaging (GCM) es un servicio que permite a los desarrolladores enviar datos desde los servidores a las aplicaciones Android o al navegador Chrome [10].

El servicio ofrece un mecanismo sencillo y ligero que los servidores pueden utilizar para avisar a las aplicaciones móviles que deben contactar con el servidor directamente en busca de datos actualizados. El servicio se encarga de todos los aspectos de gestión de colas de mensajes y la entrega a la aplicación de destino.

GCM es un servicio gratuito que tiene la capacidad de enviar un mensaje ligero informando a la aplicación Android de nuevos datos en el servidor. También puede enviar mensajes más grandes con hasta 4KB de datos útiles. Además de un tamaño máximo en los mensajes, Google limita también el número de mensajes que un remitente envía en conjunto, y el número de mensajes que un remitente envía a un dispositivo específico.

La aplicación no tiene por qué estar en ejecución para la recepción de los mensajes. El sistema despertará la aplicación cuando llegue el mensaje a través de un mecanismo llamado Broadcast, siempre y cuando la aplicación tenga definido el sistema Broadcast apropiado y los permisos adecuados.

2.9.2. Google Maps Android

Es una API que puede agregar mapas basados en mapas de Google en aplicaciones Android [11]. La API se encarga de automatizar el acceso a los servidores de Google Maps, la descarga de datos y la visualización del mapa. También puede utilizar llamadas a la API para agregar marcadores, polígonos y superposiciones en un mapa básico, y para cambiar la vista del usuario de un mapa a una zona en particular. Estos objetos proporcionan información adicional para ubicaciones del mapa y permiten la interacción del usuario con el mapa.

2.9.3. Google Maps Directions

Google Maps Directions es un servicio que calcula direcciones entre ubicaciones mediante una petición HTTP [12]. Puede buscar direcciones para varios modos de transporte: en transporte público, en coche o andando. Directions pueden especificar orígenes, destinos y puntos de interés, ya sea como cadenas de texto o como coordenadas de latitud/longitud.

Este servicio está diseñado en general para el cálculo de direcciones conocido con antelación y por tanto, no concebido para responder en tiempo real.

Capítulo 3

Estado del Arte

En esta sección se presentarán aplicaciones similares a la que se pretende realizar en este proyecto, realizando primero una introducción de las distintas aplicaciones, y presentándose para finalizar, un cuadro comparativo de las herramientas descritas.

3.1. BlaBlaCar

BlaBlaCar [13] es una plataforma web organizada como una red social nacida en 2004, con la finalidad de compartir coche entre sus usuarios. Este servicio se centra en el marco europeo, y actualmente, es uno de los referentes en plataformas de compartir coche.

Características

Las principales características a tener en cuenta de esta aplicación son las siguientes:

- Está organizada como una red social, lo que facilita que los usuarios se comuniquen entre ellos sin necesidad de intermediarios a la hora de compartir viaje.
- Los perfiles de los usuarios son muy completos, ya que proveen información sobre las preferencias en el viaje y las valoraciones de otros usuarios sobre el conductor.
- El buscador es sencillo con funciones de autocompletado, por lo que facilita una búsqueda rápida y eficaz que es esencial en una plataforma para compartir coche.
- Dispone de una aplicación móvil con las funciones principales de la web.

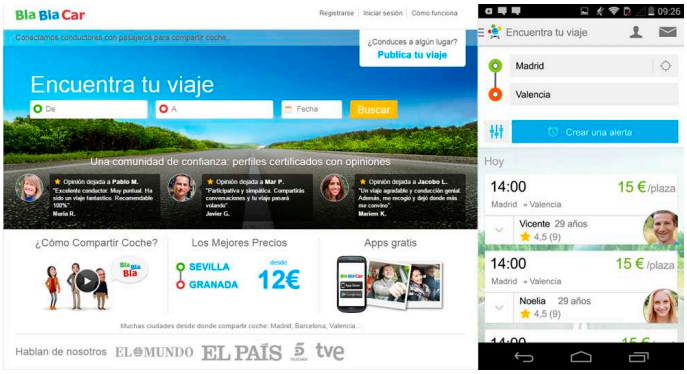

Figura 3.1: Página web y aplicación móvil de BlaBlaCar

- Tiene un buen sistema de alertas de viaje, lo que permite a los conductores encontrar pasajeros.

3.2. Carpooling

Es una red europea de viajes en coche compartido con gran experiencia en este ámbito y que transporta más de 1 millón de personas cada mes [14].

Características

Las principales características a tener en cuenta de esta aplicación son las siguientes:

- Tiene una sección de dudas muy elaborada.
- El sistema de reservas se toma muy en serio las experiencias de los usuarios, permitiendo valoraciones y vigilando todo el proceso.
- El origen y el destino no tienen porqué coincidir con el del conductor, y se pueden introducir hasta cinco ciudades de paso, lo que amplía la flexibilidad y la rentabilidad del viaje.
- Dispone de una aplicación móvil y la estructura web también tiene forma de red social.
- El servicio es gratuito en la modalidad de pago en efectivo. Opcionalmente, el pasajero puede hacer una reserva de plaza pagándola por adelantado, quedándose la empresa un 11 % del importe, que deja de recibir el conductor.

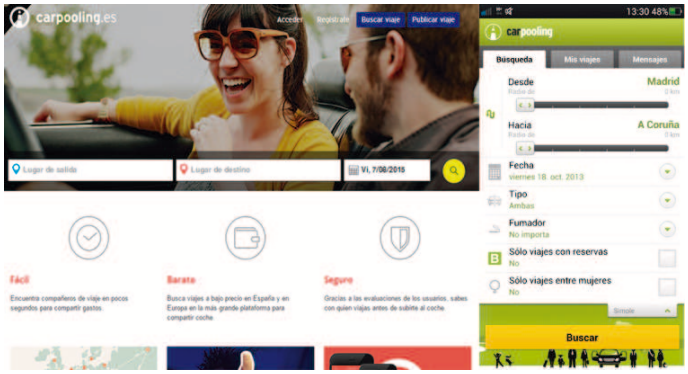

Figura 3.2: Página web y aplicación móvil de Carpooling

3.3. Amovens

Es una plataforma tanto de alquiler como para compartir coche realizada por una empresa española [15]. Focaliza buena parte de sus esfuerzos en extender el uso del coche compartido en las empresas, universidades, administraciones, eventos y festivales.

Características

Las principales características a tener en cuenta de esta aplicación son las siguientes:

- Una interfaz sencilla en la que es fácil publicar y encontrar un viaje.
- Tiene una sección adicional para encontrar conductores o pasajeros para eventos, por ejemplo para festivales de música.
- No permite publicar un anuncio como pasajero, pero sí que permite crear alertas para los viajes.
- La estructura de la plataforma como en los casos anteriores, está organizada en forma de red social y posee una aplicación móvil.

3.4. Otras plataformas

Aparte de las mencionadas anteriormente, hay varias web como **Compartocoche.com** [16] y **Viajamosjuntos.com** [17]. Estas web parecen menos funcionales e incluso abandonadas, por ello, no se han tenido en cuenta a la hora de estudiarlas para análisis del estado del arte.

Figura 3.3: Página web y aplicación móvil de Amovens

3.5. Tabla comparativa

Después de haber descrito las plataformas más importantes para compartir coche, se va a exponer una tabla comparativa con las características más importantes (Ver Tabla 3.1).

Analizando la tabla se puede observar que las tres aplicaciones tienen prácticamente los mismos servicios y características, lo cual nos indica las características a tener en cuenta en nuestra aplicación, por ejemplo la importancia de un buscador sencillo y que sea fácil publicar un viaje.

Se ha observado que todas las aplicaciones no tienen un servicio que dado un origen, un destino y una desviación muestren automáticamente los municipios de paso; solo te permiten seleccionar algunas manualmente. Tampoco existe la posibilidad de un seguimiento del viaje en tiempo real, lo que facilitaría a los usuarios saber en todo momento dónde está el conductor y cuándo les va a recoger.

Otro punto a tener en cuenta es que el conductor sea el que pueda encontrar a los pasajeros, solo se contempla en la plataforma Carpooling, mientras que las otras dos tienen un sistema de alertas muy útil para saber cuándo un conductor ha creado un viaje que se ajusta a las preferencias del pasajero.

TeConvido parte de las fortalezas y debilidades de las aplicaciones que se han detallado anteriormente. En una primera versión de la aplicación contemplaría:

- Sencillez a la hora de publicar como de buscar un viaje.
- Posibilidad de publicar como pasajero y a su vez añadir una alerta de viaje que avise a ese usuario cuando hay un conductor con esas preferencias de viajes.
- Selección automática de los municipios colindantes a una ruta dada, aumentando la posibilidades de que un pasajero encuentre un conductor con un

	BlaBlaCar	Caropooling	Amovens	TeConvido v1	TeConvido v2
Aplicación móvil	X	X	X		X
Buscador sencillo	X	X	X	X	X
Fácil de publicar un viaje	X	X	X	X	X
Municipios de paso automático				X	X
Publicar como pasajeros		X		X	X
Alertas de viajes	X		X	X	X
Información de coche	X	X	X		X
Valoración de los viajes	X	X	X		X
Reserva de plaza	X	X	X	X	X
Precio regulado	X			X	X
Seguimiento del viaje					X

Tabla 3.1: Comparativa de aplicaciones

origen y un destino que no tienen por qué ser los del conductor.

- Regulación del precio del viaje conforme a la media del coste del combustible en la provincia de Teruel, para evitar que el conductor saque beneficio.

Por último, debido al gran volumen de trabajo no se ha podido incluir todas las características en la primera versión de la aplicación y se dejará para líneas futuras. Estas mejoras serán la creación de una aplicación móvil, la posibilidad de seguimiento del viaje, la valoración de los viajes por parte de los usuarios, así como información ampliada del coche con el que se realizará el viaje.

Capítulo 4

Análisis

Este capítulo trata sobre la fase de análisis de la aplicación. Se presentará el documento de especificación de requisitos en el que se detallarán cada uno de los requisitos de la aplicación.

4.1. Introducción

El presente capítulo contiene una Especificación de Requisitos de Software (ERS) de la aplicación informática *TeConvido*, siguiendo el estándar *IEEE Recommended Practice for Software Requirementes Specifications IEEE 830-1998* [1].

4.1.1. Propósito

El presente capítulo tiene como propósito ser una guía para el desarrollo de la aplicación, así como facilitar la posterior implementación. Irá dirigido al equipo de desarrollo de la aplicación, así como a los usuarios de la misma.

4.1.2. Alcance

El objetivo es obtener una plataforma software tanto web, como para dispositivos Android que permita la comunicación entre conductores y pasajeros dentro de la provincia de Teruel.

Además del software, se realizará también la especificación del hardware necesario para su ejecución con unos tiempos de respuesta que se consideren aceptables.

Se requerirá que toda la documentación referente al proceso de diseño sea conforme al estándar UML.

4.1.3. Definiciones, acrónimos y abreviaturas

En este apartado se proporcionan las definiciones, acrónimos y abreviaturas necesarios para interpretar adecuadamente el presente Documento de Especificación de Requisitos. Los términos son los siguientes:

- BD. Es un acrónimo utilizado en este documento para referirse a la base de datos.
- MCBD. Es un acrónimo utilizado en este documento para referirse al módulo que hace de nexo de unión entre la base de datos y los servidores.
- SSL. Es un acrónomio para referirse al *Secure Sockets Layer*, es una capa adicional añadida a la comunicación mediante Sockets que utiliza técnicas criptograficas de clave asimétrica para intercambiar una clave simétrica, y empezar una comunicación encriptando y desencriptando con esta clave [18].
- HTTPS. Es un acrónomio para referirse al *Hypertext Transfer Protocol Secure*, protocolo seguro de trasferencia de datos de hipertexto, es decir, es la versión segura de HTTP [19].
- Prepared Statement. En los sistemas gestores de bases de datos, es una declaración preparada o parametrizada que forma una plantilla de un SQL en la que se sustituyen determinados valores en tiempo de ejecución. Su principal característica es la optimización en la declaración, ya que con una declaración se puede ejecutar más de una vez, y son declaraciones resistentes contra la inyección de SQL [20].
- GCM. Es un acrónimo de *Google Cloud Messaging*, es un servicio que permite a los desarrolladores enviar datos desde los servidores de aplicaciones a dispositivos Android.

4.1.4. Referencias

Este capítulo se ha elaborado siguiendo las directrices indicadas en el documento *IEEE Recommended Practice for Software Requirementes Specifications* [1]

4.1.5. Visión general del capítulo

El resto del capítulo está estructurado de la siguiente manera:

- Descripción general: Apartado en el que se proporciona una visión general acerca de las funcionalidades de la aplicación web y móvil.
- Requisitos específicos: En este apartado se describen las funcionalidades de la aplicación de un modo más técnico que en el primer apartado, ya que va dirigido principalmente a los desarrolladores.

4.2. Descripción Global

4.2.1. Perspectiva del producto

Para el funcionamiento correcto de la plataforma *TeConvido* son necesarios los siguientes componentes software que se pueden observar en la Figura 6.1. El sistema está compuesto por cinco elementos clave, que son los siguientes:

- Servidor PHP. Interactúa con el cliente o navegador web y el MCBD. Su objetivo es mostrar la información de la BD de manera visual en el navegador web y realizar acciones con ésta.
- Servidor Java. Gestiona las peticiones de los clientes Android. La función de este servidor es comunicar los clientes Android con el MCBD y así, evitar una comunicación directa desde el cliente, por motivos de seguridad, ya que el cliente no tiene porqué saber dónde están alojados el MCBD y la BD. También podría tener futuros usos con clientes de escritorio si se considerase oportuno.
- Cliente Android. Representa y transforma la información de la BD desde las interfaces Android.
- Módulo de Conexión con la BD. Es el encargado de comunicar los servidores PHP y Java con la BD. Es un módulo diseñado para la optimización de un código común y facilitar la interoperabilidad de diversos lenguajes. Por ello, se realizarán peticiones HTTPS mediante el método POST a páginas en PHP que gestionarán acciones determinadas con la BD. Los parámetros de estas peticiones POST, así como la respuesta, vendrán dados mediante JSON.
- Base de Datos. Es la encargada de almacenar todos los datos tanto de usuarios como de los propios viajes de la plataforma *TeConvido* y gestionarlos. Es la pieza fundamental en la que gira toda la plataforma.

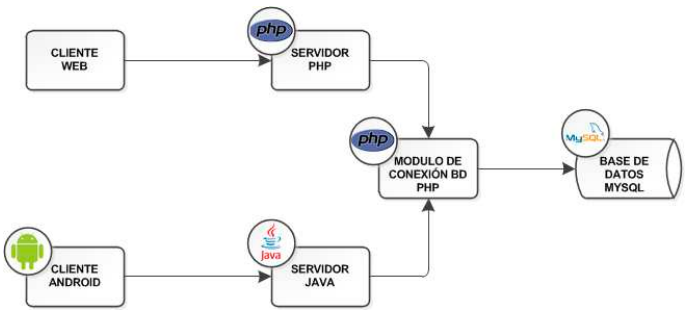

Figura 4.1: Arquitectura de la aplicación

Interfaces del sistema

El sistema a desarrollar constará de una aplicación web y otra para dispositivos móviles Android para acceder a todas las funcionalidades del sistema.

Interfaces de usuario

Las interfaces de usuario deberán ser adecuadas y accesibles para cualquier usuario que utilice la plataforma. Se utilizarán los diferentes criterios de usabilidad aplicables al diseño de la aplicación, tales como navegación por la aplicación, ergonomía, interacción y manejo de errores.

Las aplicaciones tanto web como Android estarán diseñadas para equipos y dispositivos con un tamaño de pantalla medio o grande, y una densidad de píxeles media o alta, para los cuales se garantiza una correcta visualización.

Interfaces software

Las interfaces software de nuestra aplicación serán las siguientes:

- Cliente web. Cualquier navegador que soporte HTML5 y Javascript.
- Cliente Android. Sistema operativo Android 4.0.3 o superior.

Interfaces de comunicación

En lo referente a las comunicaciones, la aplicación tendrá de cinco tipos:

- Comunicación entre el navegador web y el servidor web a través del protocolo HTTPS.
- Comunicación entre el cliente Android y el servidor Java mediante el protocolo de comunicación TCP/IP y se establecerá a través de SSL.
- Comunicación entre el servidor web o Android con el MCBD a través del protocolo HTTPS.
- Comunicación entre el MCBD y la BD, mediante el protocolo TCP/IP.
- Comunicación con GCM realizada mediante la API de Google.

Restricciones de memoria

En el caso de los servidores, los requisitos mínimos de memoria para asegurar una correcta respuesta deberán ser de 4GB de memoria RAM y 50MB de espacio libre. En el caso específico del servidor que contendrá la base de datos se requerirá 6GB de memoria RAM y un espacio libre conforme al tamaño de la base de datos.

En el caso de ejecutar la aplicación web desde cualquier terminal, no hay restricción de memoria RAM, ya que, con cualquier dispositivo de hoy en día, incluso

con los de gama baja, se podrá ejecutar la aplicación de forma fluida. No habrá ninguna restricción de almacenamiento.

En el caso de ejecutar la aplicación cliente Android, no habrá ninguna restricción de memoria RAM. En cuanto al almacenamiento serán necesarios 15MB libres, como mínimo.

Casos de uso

En la Figura 4.2 se muestra el diagrama casos de uso correspondientes, tanto para la aplicación Web como Android.

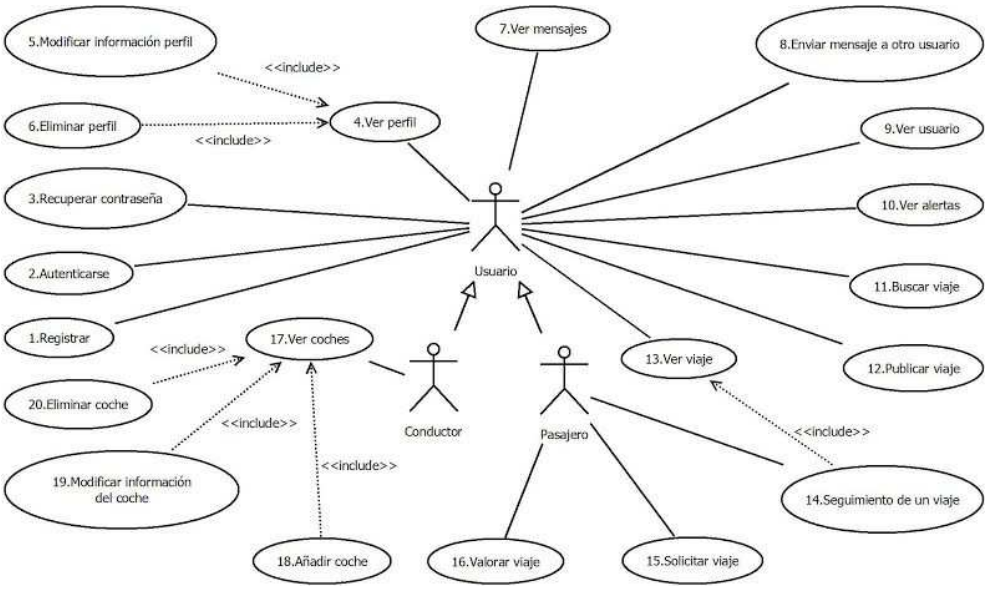

Figura 4.2: Diagrama de casos de uso

CU-01. Registrar. La interfaz proporcionará al usuario la posibilidad de registrarse en el sistema.

CU-02. Autenticarse. El usuario podrá acceder al sistema mediante su correo y la contraseña con la que se registró, tanto para poder acceder a todas las funcionalidades como publicar un viaje.

CU-3. Recuperar contraseña. El usuario podrá recuperar la contraseña a través del correo electrónico con el que se registró.

CU-04. Ver perfil. Se podrá consultar el perfil del usuario con la información de todos sus datos.

CU-05. Modificar información perfil. Se podrá modificar la información personal del usuario.

CU-06. Eliminar perfil. Se permitirá al usuario eliminar su cuenta en cualquier momento.

CU-07. Ver mensajes. Se podrá consultar la lista de los mensajes que tiene el usuario de otros usuarios.

CU-08. Enviar mensaje a otros usuarios. Se podrán enviar mensajes a otros usuarios con el fin de poder comunicarse entre ellos para coordinarse en referencia a un viaje.

CU-09. Ver usuario. Se podrá consultar el perfil de otros usuarios con la información pública de éstos.

CU-10. Ver alertas. El usuario podrá consultar la lista de alertas de viaje y ver las solicitudes a sus viaje de otros usuarios.

CU-11. Buscar viaje. Se podrán consultar todos los viajes que hay con unas preferencias determinadas de origen, destino y fecha.

CU-12. Publicar viaje. El usuario podrá publicar un viaje con el rol de conductor o como pasajero. El usuario, independientemente de si es conductor o pasajero, deberá rellenar un formulario con la información básica del viaje.

CU-13. Ver viaje. Se podrá consultar la información sobre un determinado viaje, independientemente si el viaje se ha realizado o todavía no.

CU-14. Seguimiento de un viaje. El pasajero de un viaje podrá consultar en "tiempo real"la posición del conductor durante el trascurso de ese viaje.

CU-15. Solicitar viaje. El pasajero podrá solicitar una plaza en un viaje publicado por un conductor.

CU-16. Valorar viaje. Una vez realizado un viaje, los pasajeros de ese viaje podrán valorar la experiencia con el conductor con el que han realizado el viaje y las condiciones del vehículo.

CU-17. Ver coches. Los conductores podrán consultar la lista de coches que han añadido al sistema.

CU-18. Añadir coche. Un conductor podrá añadir coches al sistema.

CU-19. Modificar información del coche. Se podrá modificar la información de los coches añadidos por un conductor.

CU-20. Eliminar coche. El usuario podrá eliminar su coche del sistema.

4.2.2. Funcionalidad del producto

Funcionalidades generales del sistema

FP-01. El sistema deberá poder atender a un número indeterminado de usuarios, desde los servidores de atención hasta el sistema gestor de base de datos.

FP-02. La aplicación deberá estar disponible las 24 horas del día, los 365 días del año

FP-03. El usuario podrá acceder al sistemas identificándose con el correo electrónico y la contraseña con la que se registró.

FP-04. Las comunicaciones del sistema estarán cifradas.

Funcionalidades de la aplicación

FP-05. Ver información de cada viaje.

FP-06. Ver información pública de cada usuario.

FP-07. Añadir y eliminar viajes.

FP-08. Solicitar ser el pasajero de un viaje.

FP-09. Añadir, modificar y eliminar valoración de un viaje.

FP-10. Modificar datos del perfil.

4.2.3. Características de los usuarios

Solo habrá un tipo de usuario, pero con dos roles diferenciados: conductor y pasajero. Los usuarios del sistema no tendrán porque tener amplios conocimientos de informática, por lo que la web tendrá que ser simple y fácil de usar.

El rol de conductor tendrá la función principal de crear los viajes y aceptar las solicitudes de los pasajeros, mientras que el rol de los pasajeros sería encontrar un viaje acorde a sus preferencias y enviar la solicitud al conductor de este.

4.2.4. Restricciones

Restricciones de diseño

- Uso de la notación de diagramas de diseño UML.
- Seguimiento del documento *IEEE Recommended Practice for Software Requirements Specifications* [1] para la documentación.

Restricciones de sistema

- El servidor web y el módulo de conexión a la base de datos deberán estar escritos en PHP.
- Se contará con un servidor Apache, un módulo de PHP y estará habilitado las peticiones HTTPS.
- El servidor Java deberá utilizar la versión 1.7 del JDK.
- El cliente Android deberá estar escrita en Java utilizando el SDK de Android, concretamente la API nivel 15.
- El sistema gestor de base de datos que se deberá usar es MySQL.

Restricciones de implementación

- Los equipos y dispositivos deberán disponer conexión a Internet.
- Los servidores deberán estar alojados en máquinas accesibles a través de Internet.
- La base de datos deberá estar alojada en un equipo al que tenga acceso el módulo de conexión con la base de datos.
- Se requerirá Android 4.0.3 o superior para el terminal que utilice el cliente Android.
- Se deberán cumplir las limitaciones tanto de la API de Google Maps como la API de GCM.

Restricciones de usabilidad

- La interfaz de ambas aplicaciones será sencilla, intuitiva y fácil de usar.
- La interfaz deberá adaptarse al tamaño de los distintos equipos y dispositivos y será capaz de visualizarse todo correctamente.

4.3. Especificación de requisitos

La especificación de requisitos del sistema impone restricciones en el diseño o la implementación para la creación del sistema. Para realizar la generación de éstos se han tenido en cuenta los aspectos descritos en los apartados anteriores donde se describe la estructura del sistema.

4.3.1. Requisitos funcionales

4.3.1.1. Requisitos de Interfaz

RQ-001. Todas las interfaces de cada aplicaciones tendrán el mismo *Look and Feel*, aunque pueden variar entre aplicaciones.

RQ-002. En el caso que el usuario no recuerde la contraseña, tendrá la opción de recuperarla, pero antes deberá facilitar el correo con el que se registró.

RQ-003. Todas las interfaces de las aplicaciones seguirán el estándar de 7+-2; con la intención de no sobrecargar al usuario de información.

RQ-004. Se utilizarán pocos colores (5+-2), y poco saturados, ya que existe un gran número de usuarios que tiene problemas visuales.

RQ-005. Todos los formularios tendrán un control de errores y todo error será debidamente notificado al usuario.

RQ-006. Todo campo de entrada de datos llevará, en la medida de lo posible, una imagen o un texto asociado que sea representativo de la información que se está introduciendo.

RQ-007. La navegación por la aplicación será sencilla y eficiente.

RQ-008. El usuario deberá saber en todo momento, en qué sección de la aplicación se encuentra.

RQ-009. El usuario no autenticado, deberá ver de forma fácil la forma de registrarse y autenticarse en las aplicaciones.

RQ-010. Todos los campos de entrada donde información a introducir sea uno de los municipios, será un campo autocompletable con los diferentes municipios de la provincia de Teruel.

RQ-011. La interfaz de *Buscar Viaje* contendrá un buscador con el origen, destino y la fecha de salida. Tanto el origen como el destino serán campos obligatorios. La fecha será un desplegable con un calendario y no será un campo obligatorio.

RQ-012. La interfaz de *Publicar Viaje* contendrá un formulario en el que se podrá elegir si publicar como pasajero o conductor. Los campos comunes a ambas opciones serán el origen, destino, fecha y hora, todos obligatorios. En caso de ser conductor se añadirán también campos para seleccionar el coche, las plazas y el importe.

RQ-013. El importe del viaje, con el fin de que los usuarios no saquen beneficio de la plataforma, no será superior a cinco veces el importe estimado a la media actualizada del precio de combustible en la provincia de Teruel, junto a la media de consumo.

RQ-014. La navbar contendrá iconos correspondientes a las alertas y los mensajes del usuario. Estos iconos cambiarán de estado en caso de que haya alertas y mensajes sin leer.

RQ-015. En la interfaz de *Mis Viajes* se podrán ver los viajes próximos, los anuncios publicados como pasajero y los viajes pasados. Se mostrará a modo de lista en los tres casos con la siguiente información del viaje: fecha y hora del viaje, origen, destino y fecha de creación del viaje. En el caso de los viajes próximos o pasados, se podrá consultar en todo momento la información adicional sobre ese viaje.

RQ-016. En la interfaz de *Alertas* se mostrará a modo de lista todas las alertas pendientes que tiene el usuario de forma paginada sin incluir más de ocho elementos por página. Se distinguirán con un color diferente las alertas leídas y las que no.

RQ-017. En la interfaz de *Mensajes* se mostrará a modo de lista con todos los mensajes recibidos del usuario de forma paginada sin incluir más de ocho elementos por página y se distinguirán los mensajes leídos de los que no, como en el requisito previo.

RQ-018. En la interfaz de *Mis Coches* se mostrará a modo de lista todos los coches del usuario de forma paginada sin incluir más de ocho elementos por página. Esta interfaz contendrá un botón que te redirigirá a la interfaz *Añadir Coche* que contendrá un formulario con la información a detallar sobre el coche.

RQ-019. En la interfaz de *Ver viaje* se facilitará la información del viaje, el vehículo y el conductor de éste. En caso de no ser el conductor y que haya plazas, podrá solicitar ser pasajero de ese viaje. Pasados un máximo de 24h del viaje, los pasajeros podrán rellenar un formulario con su valoración del viaje, que incluirá una puntuación y una descripción.

RQ-020. En caso de que el conductor habilite el seguimiento del viaje, los pasajeros de un viaje podrán ver en un mapa dónde se encuentra el conductor.

Aplicación Web

RQ-021. La aplicación se desarrollará para equipos y dispositivos con una resolución mínima de 480 x 800 píxeles. En caso de tener una resolución menor, no se garantiza su correcta visualización.

RQ-022. Todas las páginas se compondrán de una navbar, un cuerpo y un pie de página.

RQ-023. La navbar tendrá dos estados dependiendo de si estamos autenticados o no. Si no estamos identificados te permitirá: autenticarte, registrarte o recuperar la contraseña; por el contrario, si lo estamos, nos permitirá consultar todo lo relativo a nuestro perfil (mis viajes, alertas, mensajes, etc.) y publicar viajes. En ambos casos la navbar permitirá acceder a la página de inicio a través del nombre de la aplicación y acceder a la página de buscar viaje.

RQ-024. El píe de página estará compuesto por el logo del grupo de investigación INIT, el logo de la Escuela Universitaria Politécnica de Teruel y el copyright, con el nombre del creador, el nombre de la aplicación y el año de creación.

RQ-025. Todas las páginas tendrán un "camino de hormigas".

Aplicación Android

RQ-026. La aplicación android se desarrollará para dispositivos con pantalla de 4,3 pulgadas, con una resolución de 720 x 1280 píxeles. En el caso de dispositivos con pantalla más grande o más pequeña y una resolución menor, no se garantiza su correcta visualización.

RQ-027. Todas las pantallas una vez autenticado, tendrán una navbar que te permitirá ir al menú inicial, a las alertas y a los mensajes de forma rápida.

RQ-028. El icono que hace referencia al menú principal en la navbar, será el logo de la aplicación que se muestra en la Figura 4.3.

RQ-029. Se podrá acceder a la mayor parte de las pantallas directamente desde la navbar y el menu principal en su conjunto.

RQ-030. En la navbar pondrá en todo momento en qué pantalla nos encontramos.

4.3.1.2. Requisitos de autenticación

RQ-031. Para el acceso a la aplicación se deberá autenticar el usuario mediante el correo y la contraseña.

RQ-032. Dentro de *Mi Perfil* se podrá modificar la información del usuario.

Figura 4.3: Logo de la aplicación

RQ-033. Se permitirá cerrar sesión en cualquier momento.

RQ-034. Dentro de la interfaz donde autenticarse habrá un enlace que permita recuperar la contraseña. Se deberá introducir el correo con el que se registró y se le facilitará a ese correo un enlace donde cambiar la contraseña.

4.3.1.3. Requisitos de registro

RQ-035. Los campos que deberá rellenar el usuario son para registrarse serán: nombre de usuario, contraseña, correo electrónico, nombre y apellidos. Todos los campos serán obligatorios.

RQ-036. No se debe permitir crear usuario con un nombre o un correo electrónico que ya exista.

4.3.1.4. Requisitos de noticificación Push

RQ-037. En la aplicación Android, una vez autenticado se registrará el código GCM del usuario en la base de datos, para los notificaciones Push.

RQ-038. Se notificará al usuario mediante notificación Push si hay un nuevo mensaje, alerta o si se actualiza la posición en el seguimiento de un viaje.

4.3.1.5. Requisitos de los servidores

RQ-039. El servidor deberá estar disponible las 24 horas del día 365 días al año.

RQ-040. El sistema deberá poder manejar un número indeterminado de usuarios.

4.3.1.6. Requisitos de la base de datos

RQ-041. La base de datos y el módulo de conexión con la base de datos se deben encontrar en el mismo equipo, pero no tendrá porqué estarlo con los otros servidores.

RQ-042. La base de datos estará construida siguiendo el esquema entidad-relación indicado en la documentación.

4.3.2. Requisitos no funcionales

4.3.2.1. Requisitos de rendimiento

RQ-043. Los servidores podrán gestionar un número indeterminado de peticiones.

4.3.2.2. Requisitos de seguridad

RQ-044. Se encriptará la contraseña en la base de datos mediante cifrado SHA1.

RQ-045. Se utilizará SSL en la comunicación TCP/IP mediante Socket.

RQ-046. Se utilizará el protocolo seguro HTTPS.

RQ-047. Se utilizará *Prepared statement* a la hora de realizar operaciones en la base de datos.

RQ-048. El acceso a la base de datos estará protegido mediante un usuario y contraseña, que no será conocida más que por los programadores y solo se podrá acceder de manera local.

Capítulo 5

Planificación y cálculo de costes

En el presente apartado se va a desarrollar la planificación realizada antes de acometer el desarrollo del proyecto y el cálculo de los costes que ocasionaría dicho desarrollo.

5.1. Planificación

Para realizar la planificación del presente proyecto, se ha tenido en cuenta cada una de las fases necesarias para finalizar exitosamente cualquier desarrollo software de cierta envergadura. Dichas fases son la de análisis, diseño, implementación y pruebas.

5.1.1. Planificación inicial

La planificación inicial está pensada para que ocupe todo el segundo cuatrimestre del curso 2013/2014, pensando en presentar el proyecto en septiembre de 2014. De esta forma la duración estimada de proyecto será de 134 días. En la Figura 5.1 se puede observar el diagrama de Gantt con la planificación inicial.

5.1.2. Planificación final

Finalmente la duración total del proyecto ha costado 499 días. La causas de esta demora son debidas a los siguientes puntos.

- Desconocimiento total de los lenguajes para programación web: PHP, CSS y Javascript, ya que partia prácticamente de cero.

- Desconocimiento de cómo interoperar dos lenguajes de programación distintos.
- Dedicación de las dos últimas semanas de agosto al estudio del examen B1 de inglés.
- Abordar un proyecto de grandes dimensiones, ya que está compuesto de dos aplicaciones con interfaz de usuario, que es la parte de la creación de software más costosa en cuanto a tiempo.
- El planteamiento desde un primer momento de crear una base en el que se apoya la aplicación totalmente genérica y reutilizable, para futuras aplicaciones con un planteamiento totalmente diferente.
- Mi incorporación al mundo laboral a través de la beca anual en el departamento de informática de la Diputación Provincial de Teruel.

En la Figura 5.2 se puede observar el diagrama de Gantt con la planificación final.

5.2. Estimación de costes

En el desarrollo de aplicaciones informáticas, al igual que en el resto de actividades comerciales, es importante conocer los costes para obtener la rentabilidad del producto. Por ello es necesario realizar una estimación de costes, para evitar tener pérdidas y sobre todo, para obtener beneficios económicos. Para ello, se va a emplear la técnica de los puntos función [21] para medir el tamaño de la aplicación.

Para comenzar la estimación mediante puntos función, debemos obtener una serie de datos que nos facilitarán su cálculo. En primer lugar, obtendremos los datos funcionales. Estos datos están divididos en dos subconjuntos:

- *Internal Logic File* (ILF). Es un grupo de datos lógicamente relacionados, identificables por el usuario y mantenidos a través de procesos dentro de los límites de la aplicación.
- *External Interface File* (EIF). Es un grupo de datos lógicamente relacionados, identificables por el usuario y usados por la aplicación pero mantenidos dentro de los límites de otra aplicación.

Los dos tipos anteriores estan formados por:

- *Data Element Type* (DET). Corresponden a atributos o campos de los ficheros, incluyendo las claves ajenas.
- *Record Element Type* (EIF). Corresponden a campos del fichero que referencia a otros ficheros.

RETs	1-19 DETs	20-50 DETs	51+DETs
1	Baja	Baja	Media
2-5	Baja	Media	Alta
6+	Media	Alta	Alta

Tabla 5.1: Matriz de complejidad para ILFs y EIFS

La complejidad de los datos que maneja la aplicación, es calculada en base a la Tabla 5.1, que asigna valores en función de los DET's y RET's que cada fichero maneja.

De este modo, se pueden observar los siguiente datos mostrados en la Tabla 5.2, para nuestra aplicación. Solo se considera que la aplicación tiene ILFs.

ILFs	DETs	RETs	Calificación	PF
Usuario	7	8	Media	10
Coche	6	2	Baja	7
Mensaje	7	2	Baja	7
Precio Combustible	4	0	Baja	7
Municipio	3	3	Baja	7
Viaje	7	7	Media	10
Anuncio pasajero	2	2	Baja	7
Viaje de conductor	5	5	Baja	7
Seguimiento viaje	4	1	Baja	7
Alerta viaje	6	2	Baja	7
Valoración viaje	5	2	Baja	7
Parada	3	2	Baja	7
Pasajero	2	2	Baja	7
Coche-Propietario	2	2	Baja	7
			TOTAL	104

Tabla 5.2: Estimación de la complejidad para ILFs

El siguiente lugar hay que identificar las distintas transacciones funcionales, divididas en:

- *External Input* (EI): Es un proceso elemental que procesa datos o información de control que viene desde fuera de los límites de la aplicación. La intención principal de un EI es mantener uno o varios ILF's y/o cambiar la conducta del sistema.

- *External Output* (EO): Es un proceso elemental que envía datos o información de control fuera de los límites de la aplicación. La intención principal de un EO es presentar información al usuario mediante lógica de proceso diferente a, y en adición a, la recuperación de datos o información de control.

La lógica del proceso ha de contener al menos una fórmula matemática o cálculo o datos derivados. Un EO puede también mantener uno o más ILF's y/o cambiar la conducta del sistema.

- *External Inquiry* (EQ): Es un proceso elemental que envía datos o información de control fuera de los límites de la aplicación. La intención principal de un EQ es presentar información al usuario mediante la recuperación de datos o información de control desde un ILF o EIF. La lógica de proceso no contiene fórmulas matemáticas o cálculos y no crea datos derivados. Ningún ILF es mantenido mediante el proceso, ni la conducta del sistema es alterada.

En las Tablas 5.3, 5.4, 5.5 se muestran las matrices partir de las cuales se calcula la complejidad.

FTRs	1-4 DETs	5-15 DETs	16+DETs
0-1	Baja	Baja	Media
2	Baja	Media	Alta
3+	Media	Alta	Alta

Tabla 5.3: Matriz de complejidad para EIs

FTRs	1-5 DETs	5-19 DETs	20+ DETs
0-1	Baja	Baja	Media
2-3	Baja	Media	Alta
4+	Media	Alta	Alta

Tabla 5.4: Matriz de complejidad para EOs

FTRs	1-4 DETs	5-15 DETs	16+DETs
0-1	Baja	Baja	Media
2	Baja	Media	Alta
3+	Media	Alta	Alta

Tabla 5.5: Matriz de complejidad para EQs

La tabla resultante para nuestra aplicación se observa en la Tabla 5.6.

Transacciones Funcionales	Tipo	DETs	FTRs	Calificación	PF
Autenticarse	EQ	2	1	Baja	3
Obtener Viaje	EQ	1	2	Baja	3
Obtener municipios de una sección	EQ	5	1	Baja	3
Obtener municipios	EO	0	1	Baja	4
Obtener viajes con conductor	EQ	3	2	Baja	3
Obtener anuncios de pasajero	EQ	3	2	Baja	3
Obtener viajes actuales de usuario	EQ	1	3	Media	4
Obtener anuncios pasajero de usuario	EQ	1	3	Media	4
Obtener viajes pasados de usuario	EQ	1	3	Media	4
Obtener valoraciones usuario	EQ	1	4	Media	4
Hay alertas de usuario	EQ	1	2	Baja	3
Hay mensajes de usuario	EQ	1	2	Baja	3
Obtener alertas de usuario	EQ	1	4	Media	4
Obtener mensajes de usuario	EQ	1	4	Media	4
Obtener coches de usuario	EQ	1	3	Media	4
Obtener mensaje de usuario	EQ	1	2	Baja	3
Obtener precio del combustible	EQ	1	1	Baja	3
Obtener seguimiento de viaje	EQ	2	4	Media	4
Es viaje de un usuario	EQ	2	3	Media	4
Obtener usuario	EQ	1	1	Baja	3
Añadir usuario	EI	5	1	Baja	3
Añadir viaje de conductor	EI	11	4	Alta	6
Añadir anuncio de viaje	EI	5	3	Alta	6
Solicitar reserva	EI	4	3	Media	4
Reservar plaza	EI	2	5	Media	4
Añadir coche	EI	6	3	Alta	6
Añadir mensaje	EI	5	2	Media	4
Añadir valoración viaje	EI	5	3	Alta	6
Modificar precio combustible	EI	2	1	Baja	3
Modificar usuario	EI	5	1	Baja	3
Modificar alerta como leída	EI	1	1	Baja	3
Modificar mensaje como leído	EI	1	1	Baja	3
Añadir seguimiento de viaje	EI	4	3	Media	4
Modificar código GCM	EI	2	1	Baja	3
Modificar coche	EI	6	3	Alta	6
Modificar valoración del viaje	EI	5	3	Alta	6
Eliminar usuario	EI	1	7	Media	4
Eliminar coche	EI	1	3	Media	4
Eliminar viaje	EI	1	6	Media	4
Eliminar valoración del viaje	EI	1	3	Media	4
				TOTAL	156

Tabla 5.6: Estimación de la complejidad para EIs, EOs y EQs

Los valores obtenidos de los puntos función se han establecido mediante la Tabla 5.7. El resultado obtenido se conoce como Unadjusted Function Points (UFP), en nuestro caso es 260 puntos función, resultado de sumar los totales de la estimación de complejidad para los IFs, EIs, EOs y EQs, correspondiente a las Tablas 5.2 y 5.6.

Para obtener un valor más real necesitamos calcular el *Value Adjustment Factor*(VAF), que se calcula a través de 14 características del sistema (GSC). Esta característica se evalua del 0 al 5, según su grado de influencia. Teniendo en cuenta lo anterior, en nuestra aplicación le corresponderian lo valores que se pueden observar en la Tabla 5.8.

La suma de la puntuación de cada característica da como resultado el grado

	Baja	Media	Alta
EI	3	4	6
EO	4	5	7
EQ	3	4	6
ILF	7	10	15
EIF	5	7	10

Tabla 5.7: Tabla de equivalencias por PF

	Factor de Complejidad	Valor (0..5)
1	Comunicación de Datos.	5
2	Proceso Distribuido.	3
3	Objetivos de Rendimiento	4
4	Configuración de Explotación compartida	1
5	Tasa de Transacciones	4
6	Entrada de Datos on-line	5
7	Eficiencia con el usuario final	3
8	Actualizaciones on-line	3
9	Procesos complejos	5
10	Reusabilidad del Código	5
11	Facilidad de instalación	4
12	Facilidad de Operación	4
13	Instalaciones Múltiples	5
14	Facilidad de Cambios	4
Factor de Complejidad Total (FCT)		55

Tabla 5.8: Cálculo de GSCs

total de influencia (TDI). El VAF se calcula según la siguiente fórmula:

$$VAF = (TDI * 0,01) + 0,65$$

Por lo tanto, en nuestro caso resulta:

$$VAF = (55 * 0,01) + 0,65 = 1,2$$

Por último, solo queda calcular los puntos función, pero esta vez teniendo en cuenta el VAF calculado en el apartado anterior. El resultado obtenido se conoce como *Adjusted Funtion Points* (AFP). Dicho cálculo queda del siguiente modo:

$$AFP = 260 * 1,2 = 312$$

En nuestro caso debido a que utilizamos lenguajes de programación de alto

nivel, la productividad viene dada por 5h cada punto función [22] Por tanto, el número de horas necesarias para realizar nuestra aplicación es de:

$$312PF * 5h/PF = 1560h$$

Asumiendo que el sueldo básico neto diario de un Ingeniero, según el convenio colectivo en vigor [23], es de 61.03€/día, el coste estimado de la mano de obra es el siguiente:

$$61,03€/8h = 7,63€/hora$$

$$1560h * 7,63€/h = 11902,8€$$

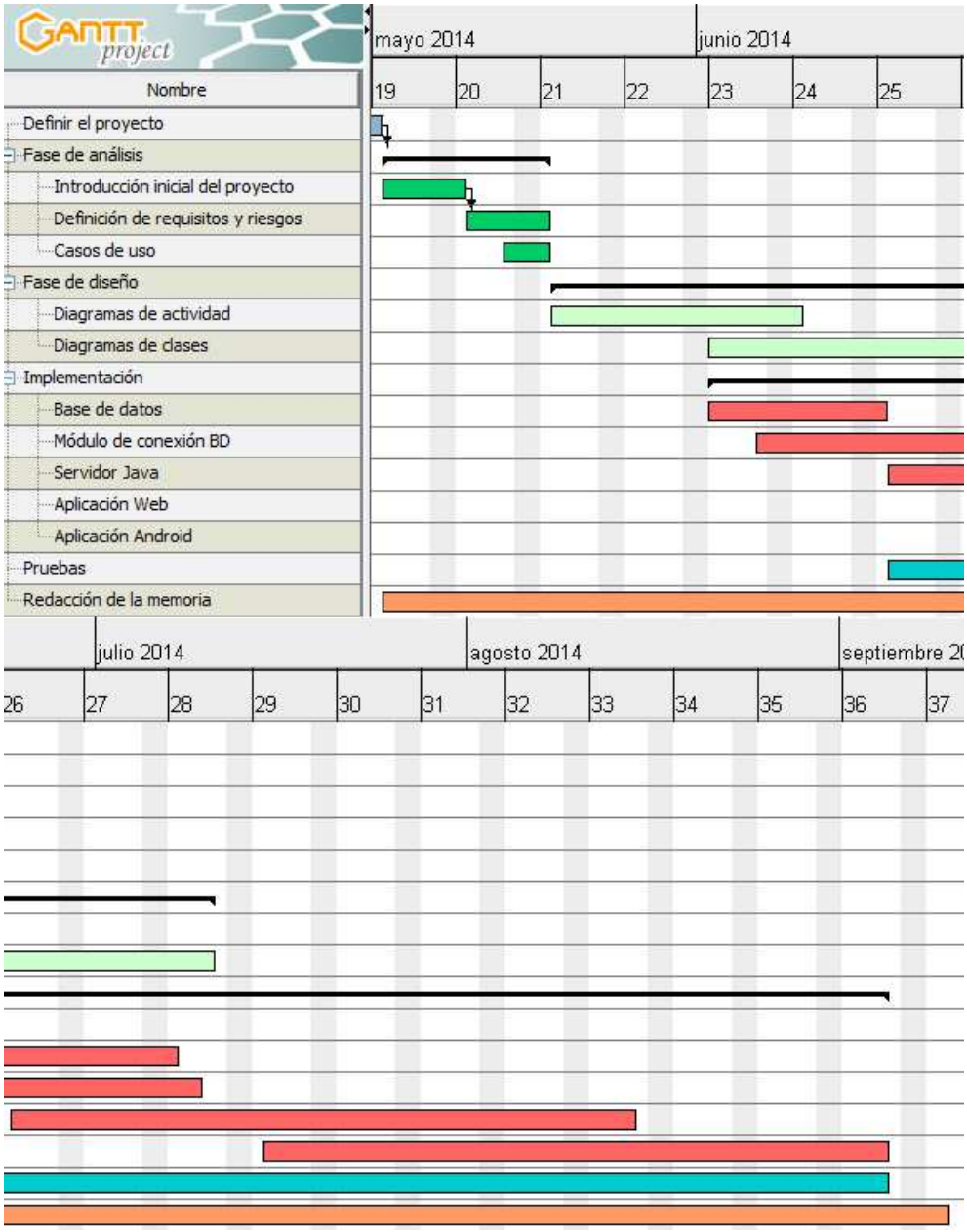

Figura 5.1: Planificación inicial

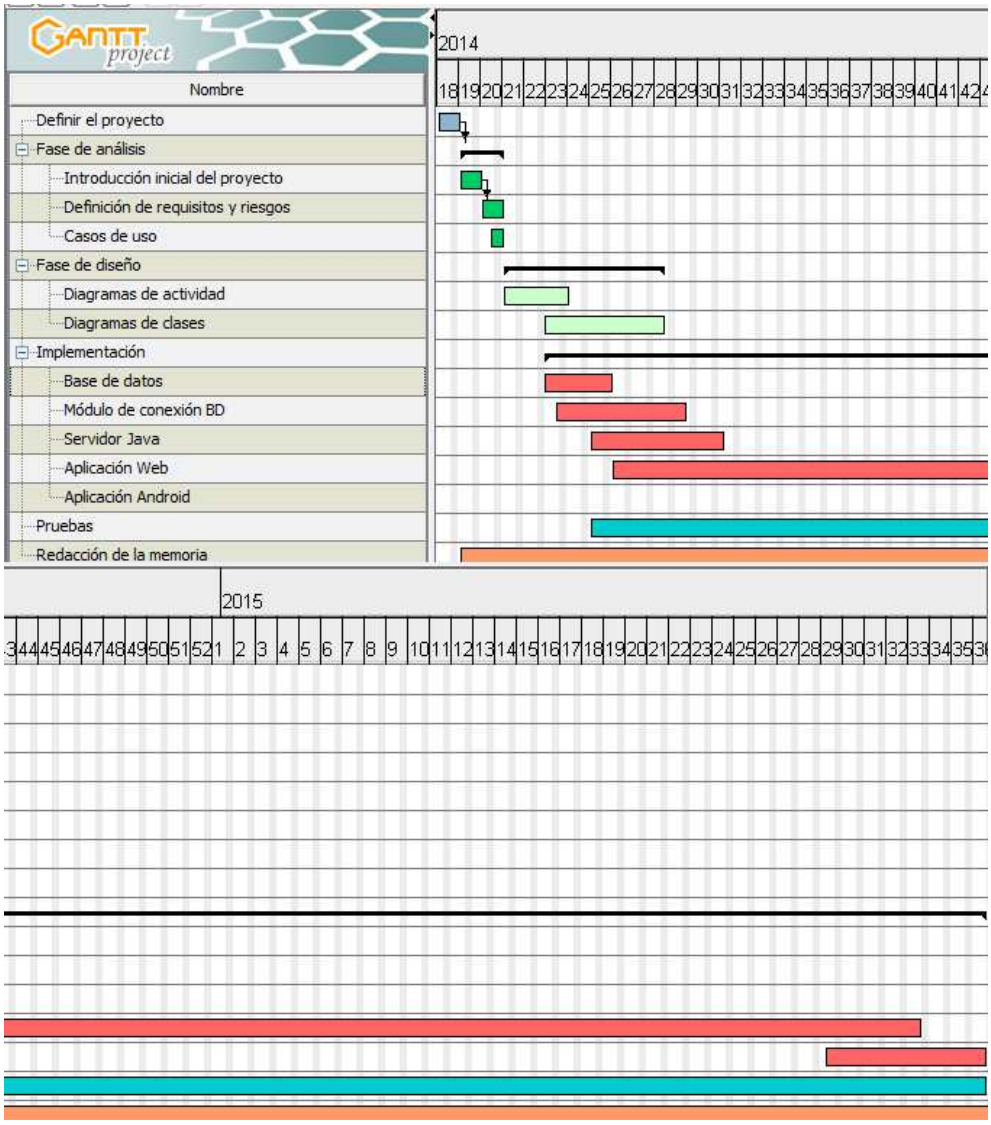

Figura 5.2: Planificación final

Capítulo 6

Diseño

Este apartado se centra en la fase de diseño de la aplicación. Dicho diseño debe ser exhaustivo y claro, de manera que no se generen dudas o confusiones durante el proceso de implementación.

6.1. Base de Datos

La base de datos es el eje central de esta aplicación, debido a que es donde se almacenarán la información sobre todos los viajes y usuario. Durante este apartado se expondrá la estructura de la base de datos del sistema, con una explicación sobre el diagrama entidad-relación de la Figura 6.1, correspondiente al diseño conceptual.

En el diagrama se observa que la base de datos se compone de 11 entidades con numerosas relaciones entre ellas. A continuación, se realiza una explicación más detallada de las entidades y las relaciones:

- *Usuario*. Contiene los datos necesarios para realizar la autenticación y los datos personales del usuario. También contendrá el código GCM en caso de haber accedido con la aplicación Android para poder notificar de nuevas alertas y mensajes.

- *Coche*. Almacena la información de un coche con el que se podrá realizar futuros viajes. Está relacionado con Usuario de manera muchos a muchos, debido a que un mismo vehículo puede ser utilizado por más de un usuario.

- *Mensaje*. Almacenará toda la información acerca de los mensajes enviados entre usuarios. Esta entidad está relacionada con la entidad *Usuario* para almacenar el emisor y el receptor.

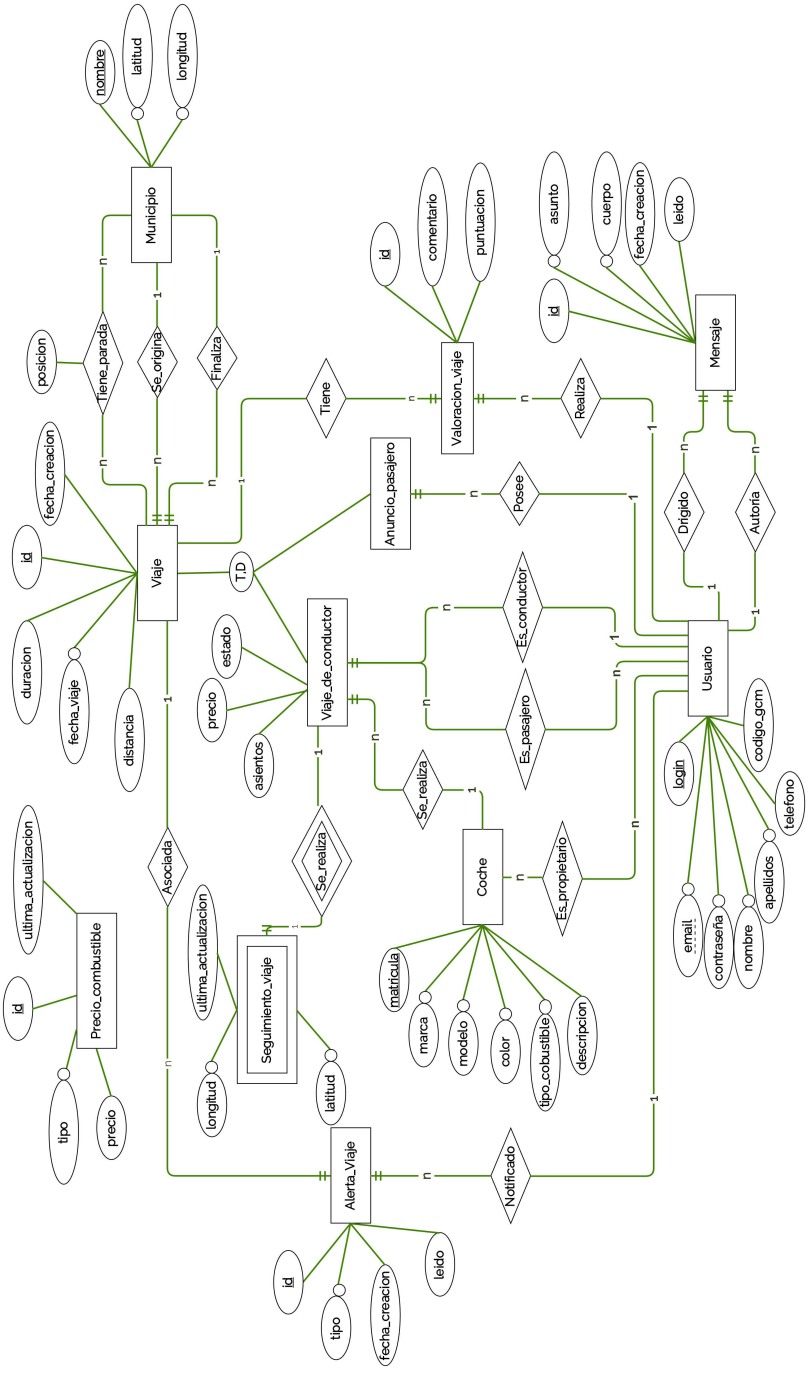

Figura 6.1: Diagrama entidad-relación de la Base de Datos

- *Precio_combustible*. Se encarga de contener los datos actualizados del precio medio del combustible en la provincia de Teruel.

- *Municipio*. Contiene el nombre de todos los municipios de la provincia de Teruel, así como su coordenadas geográficas, ya que son necesarias para obtener los municipios cercanos a una ruta.

- *Viaje*. Es la entidad central de la base de datos junto con Usuario. Almacena toda la información relativa a un viaje. Está relacionado de tres maneras diferentes con *Municipio*, debido a que se debe conocer el origen, el destino y las paradas del viaje. Esta entidad se especializa en dos: *Viaje_de_conductor* y *Anuncio_pasajero*; esta especialización es total y disjunta.

- *Anuncio_pasajero*. Guarda la información sobre los anuncios de viaje de pasajeros que buscan conductor. Está relacionado con *Usuario*, para contener el usuario que creó el anuncio.

- *Viaje_de_conductor*. En esta entidad se almacena toda la información sobre un viaje creado por un conductor, y por tanto, se va a realizar. Se guarda toda la información relativa al viaje, al conductor, a los pasajeros y al coche con que se realiza, de ahí la relación doble con *Usuario* y la relación con coche.

- *Seguimiento_viaje*. Contiene las coordenadas referentes a la posición geográfica de un conductor mientras que esté realizando el viaje.

- *Alerta_viaje*. Almacena la información sobre las alertas relativas a los viajes. Se han considerado tres tipos de alertas en esta primera versión: aviso de viaje, que notificará al usuario al poner un anuncio como pasajero si se crea un viaje con las preferencias que había puesto; de solicitud de viaje, para alertar a un conductor de que un pasajero solicita ser pasajero de un viaje; y de confirmación de viaje, para notificar a un pasajero que el conductor ha aceptado su solicitud. La entidad está relacionada con la entidad *Viaje* del cual notifica y con la entidad *Usuario* para saber de quién es la solicitud de viaje o a quién notifica la alerta.

- *Valoracion_viaje*. Esta entidad contendrá la información relativa a la retroalimentación de los viajes, para que los pasajeros puedan valorar su experiencia en un determinado viaje.

6.2. Interfaz de usuario

La interfaz de usuario es un aspecto importante en la mayoría de aplicaciones informáticas. En el caso de la aplicación que estamos desarrollando, adquiere especial importancia debido a que los usuarios puede ser de un amplio abanico de edades, y con conocimientos de informática muy variables, y por tanto, es necesario una interfaz fácil, sencilla y usable. En este apartado, se desarrollan todos los aspectos relacionados con dicho interfaz.

6.2.1. Interfaz Web

Árbol de pantallas

En la Figura 6.2 se muestra el árbol de pantallas de la interfaz web, con sus diferentes transacciones.

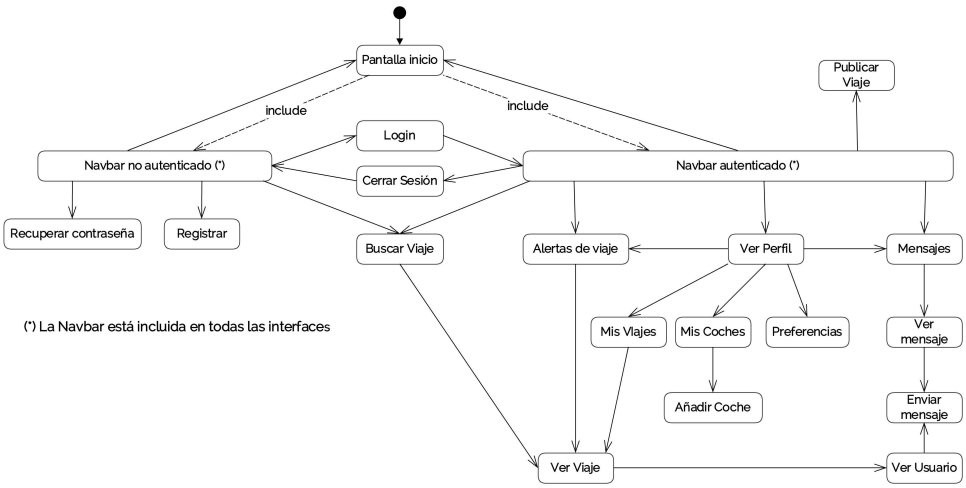

Figura 6.2: Árbol de pantallas web

Como se puede observar, cada elemento rectangular representa una pantalla distinta en el entorno web. La pantalla de inicio correspondería a la primera interfaz que ve el usuario. Esta interfaz, junto con el resto, está compuesta con un elemento navbar que nos permite acceder a todas las interfaces de primer nivel.

Esta navbar tiene dos posibles estados dependiendo si se está autenticado o no. Dependiendo del estado podremos acceder a unas interfaces u otras. Las interfaces comunes a los dos estados son *Buscar viaje* y la pantalla de inicio.

Prototipo de interfaz

Previamente a la implementación de la interfaz gráfica de usuario, se han realizado los prototipos (mockups) de cada una de las pantallas que forman la aplicación. Esto se hace para tener una idea clara de cómo debe ser cada una de las pantallas, así como sus funcionalidades. En el Anexo C se presentan los prototipos de las interfaces para la aplicación web.

6.2.2. Interfaz Android

Árbol de pantallas

En la Figura 6.3 se muestra el árbol de pantallas de la interfaz Android, con sus diferentes transacciones.

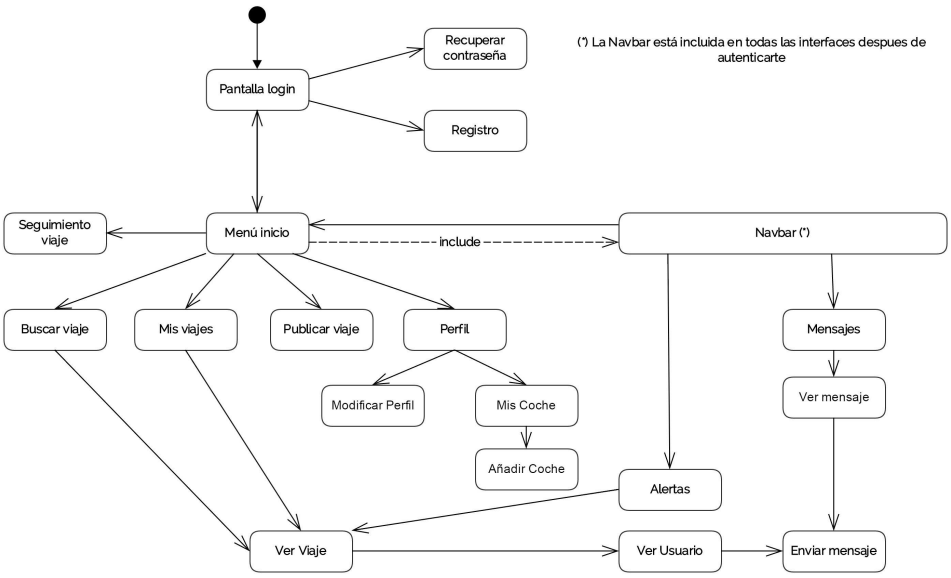

Figura 6.3: Árbol de pantallas Android

Como se puede observar en la figura, para utilizar esta aplicación habrá que estar registrado en el sistema. La primera pantalla una vez autenticado será un menú de inicio donde se podrá acceder directamente a las funciones principales de la aplicación.

Todas las pantallas una vez autenticado, tendrá una navbar con un acceso rápido al menú de inicio, mensajes y alertas. En todo momento se podrá volver a la interfaz que ha invocado a la actual.

Prototipo de interfaz

Como en el caso de la aplicación web, se han desarrollado unos prototipos para orientar la implementación de la interfaz Android en el Anexo D.

6.3. Diagramas de clases

En este apartado se van a exponer los diagramas de clases de nuestro proyecto de todos las partes del sistema. Esta plataforma sigue el patrón modelo-vista-controlador (MVC). En todos los casos se mostrara el diagrama de clases completo con sus relaciones y se indicara donde se pueden visualizar las clases en detalle.

6.3.1. Diagrama de clases del modelo

En la Figura 6.4 se observa la transformación de las entidades y las relaciones necesarias de la base de datos a clases, junto con la clases *Coordenada* y *TicketLogin*. Estas clases son necesarias en todos los sistemas, es decir, el servidor web, el servidor Java, el cliente Android, y el módulo de conexión con la base de datos, ya que corresponden al modelo de la plataforma.

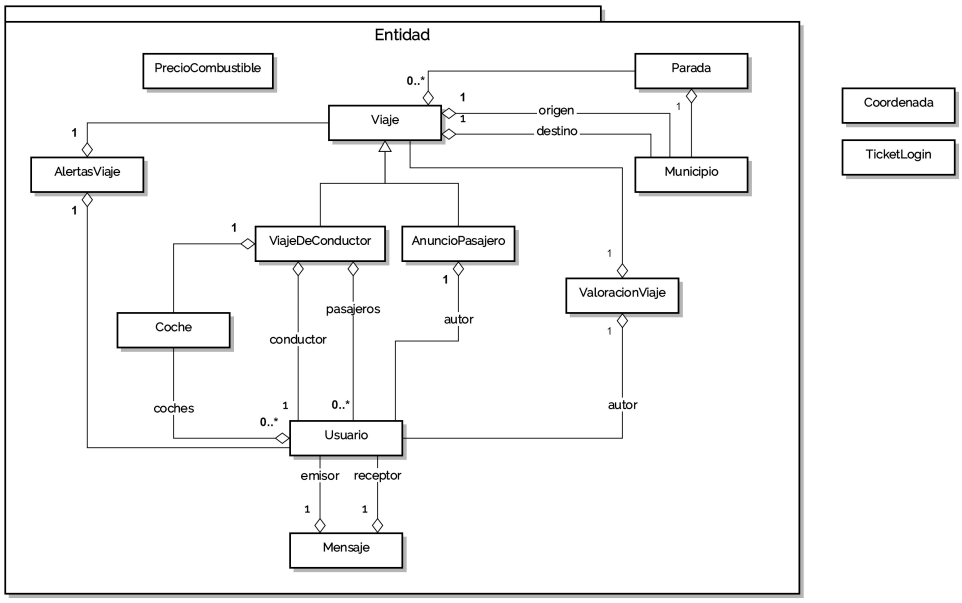

Figura 6.4: Diagrama de clases del modelo

6.3.2. Diagrama de clases servidor Web

La Figura 6.5 corresponde al diagrama de clases del servidor Web.

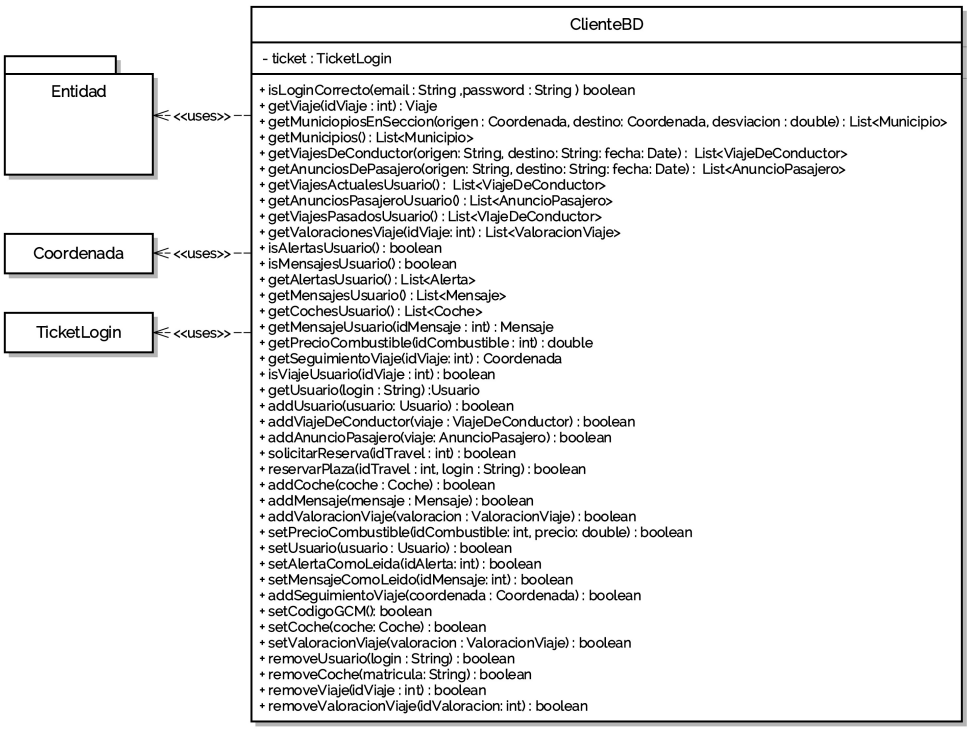

Figura 6.5: Diagrama de clases del servidor Web

Está compuesto por las clases citadas en el punto anterior y la clase *ClienteBD*, encargada de comunicarse con el módulo de conexión de base de datos a través de peticiones HTTPS, para realizar consultas y modificaciones de la información del sistema.

6.3.3. Diagramas de clases cliente Android y servidor Java

En este apartado se va a exponer tanto el diagrama de clases del servidor Java como el del cliente Android, ya que tienen las clases correspondientes a la comunicación en común. En primer lugar, en la Figura 6.6 se observa la estructura del servidor. Es un servidor multihilo para la atención de tareas procedentes de cliente.

Está compuesto de un gestor de tareas que es el encargado de crear *HiliTareaPetición* en cuanto llega una petición de un cliente, que se reconvierte en una tarea más específica dependiendo del tipo de tarea que quiera realizar el cliente.

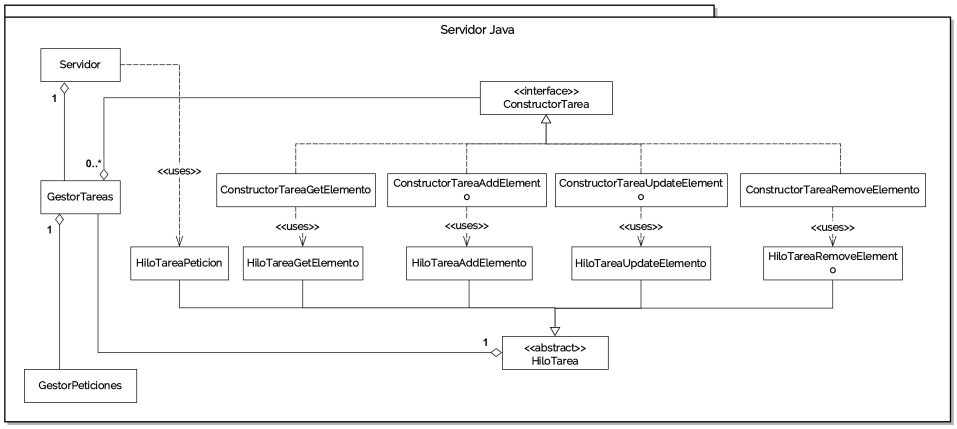

Figura 6.6: Diagrama de clases servidor Java

En la Figura 6.7, se observa tanto el diagrama de clases de cliente Android, como las clases que tienen en común con el servidor Java.

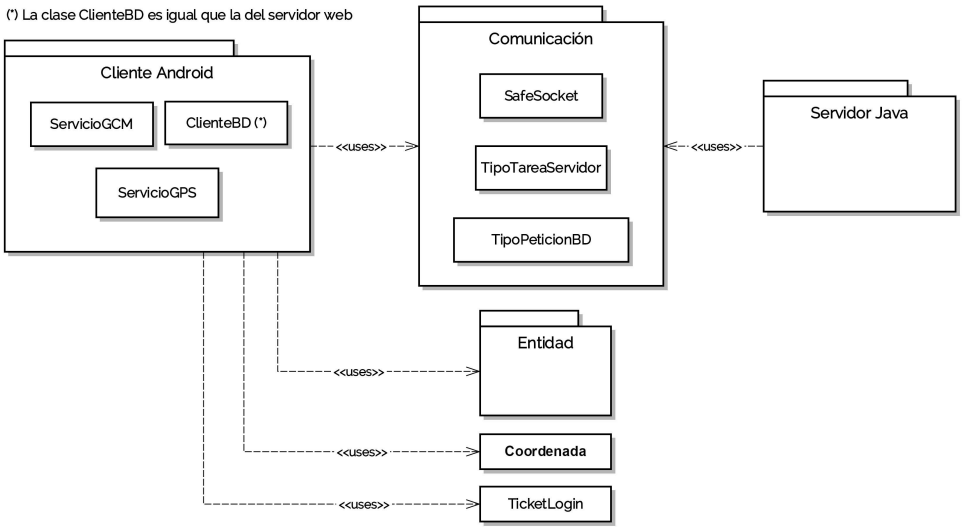

Figura 6.7: Diagrama de clases servidor Java y cliente Android

Capítulo 7

Conclusiones

El principal objetivo que se pretende conseguir con la realización de este proyecto, es diseñar e implementar una aplicación centrada en la provincia de Teruel que ayude a comunicar conductores y pasajeros con rutas iguales o compatibles, para conseguir así paliar una carencia de transporte público y las necesidades de sus habitantes, reduciendo también las emisiones de CO2.

Para alcanzar este objetivo se a realizado una aplicación web en el lenguaje PHP. El sistema cuenta también con un módulo en PHP que interactúa con el sistema gestor de base de datos MySQL y los servidores Web y Java.

Debido a la sobrecarga de horas que ha conllevado adaptar la página web a todos los dispositivos y el tiempo invertido en aprender los lenguajes de programación PHP, CSS y JavaScript; no se ha podido lograr todos los objetivos de la aplicación. Concretamente no se ha implementado en su totalidad la aplicación Android, quedando pendiente la implementación de parte de las interfaces.

Por último, la experiencia ha sido muy buena, tanto a nivel personal como profesional. Me he enriquezado en todo lo referente a lenguajes de programación web, ya que empecé con pocos conocimientos y termino con gran conocimiento sobre la creación de páginas web. También he ido un paso mas allá con respecto a los sistemas gestores de base de datos, realizando numerosos procedimientos almacenados y he observado las ventajas de rendimiento y optimización que conlleva.

Bibliografía

[1] IEEE. IEEE Recommended Practice for Software Requirements Specifications, 1998. URL: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?reload=true&punumber=5841.

[2] Wikipedia. PHP, 2015. URL: https://es.wikipedia.org/wiki/PHP.

[3] Wikipedia. HTML, 2015. URL: https://es.wikipedia.org/wiki/HTML.

[4] LibrosWeb. CSS, 2015. URL: https://librosweb.es/libro/css/capitulo_1.html.

[5] Wikipedia. Javascript, 2015. URL: https://es.wikipedia.org/wiki/JavaScript.

[6] Wikipedia. jQuery, 2015. URL: https://es.wikipedia.org/wiki/JQuery.

[7] JSON, 2015. URL: http://json.org/json-es.html.

[8] Wikipedia. Android, 2015. URL: https://es.wikipedia.org/wiki/Android.

[9] Wikipedia. MySQL, 2015. URL: https://es.wikipedia.org/wiki/MySQL.

[10] Wikipedia. Google GCM, 2015. URL: https://en.wikipedia.org/wiki/Google_Cloud_Messaging.

[11] Google Developers. Google Maps Android, 2015. URL: https://developers.google.com/maps/documentation/android/intro.

[12] Google Developers. Google Directions, 2015. URL: https://developers.google.com/maps/documentation/directions/intro.

[13] BlaBlaCar, 2015. URL: https://www.blablacar.es/.

[14] Carpooling, 2015. URL: http://www.carpooling.es/.

[15] Amovens, 2015. URL: https://amovens.com/.

[16] Compartococho.com, 2015. URL: http://www.compartococho.com/.

[17] ViajamosJuntos.com, 2015. URL: http://www.viajamosjuntos.com/.

[18] Wikipedia. SSL, 2015. URL: https://es.wikipedia.org/wiki/Transport_Layer_Security.

[19] Wikipedia. HTTPS, 2015. URL: https://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol_Secure.

[20] Wikipedia. Prepared Statement, 2015. URL: https://en.wikipedia.org/wiki/Prepared_statement.

[21] Fernando Naranjo Palomino. Apuntes de la asignatura de Ingienería del Software II, 2004.

[22] Universidad Politécnica de Valencia. Estimación del esfuerzo en proyectos informáticos, 2015. URL: www.upv.es/~jmontesa/eog/4-eog00.doc.

[23] Convenio colectivo estatal, 2015. URL: http://www.boe.es/boe/dias/2009/04/04/pdfs/BOE-A-2009-5688.pdf.

Anexo A

Diagramas de clases del modelo

En este anexo se va a exponer los diagramas de clases del modelo en detalle. Todas las clases tiene la estructura de un *Bean*, es decir, esta compuesto por un constructor sin parámetros y otro con todos los atributos, y todos los atributos tiene un método *set* y *get*.

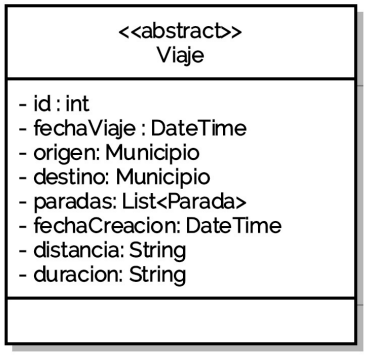

Figura A.1: Clase Viaje

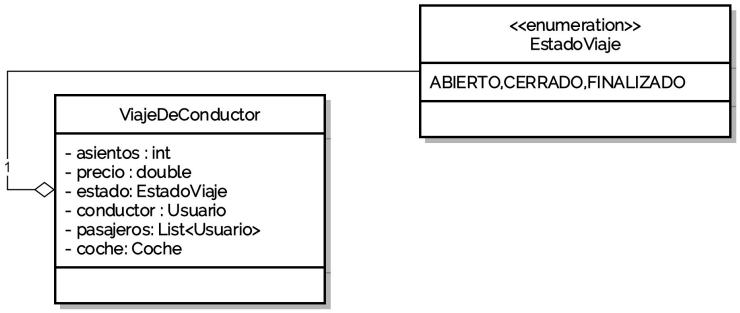

Figura A.2: Clase ViajeDeConductor

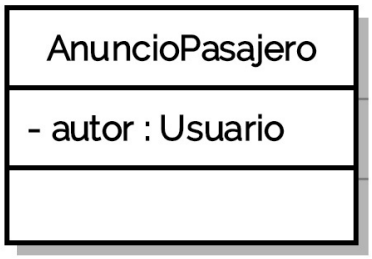

Figura A.3: Clase AnuncioPasajero

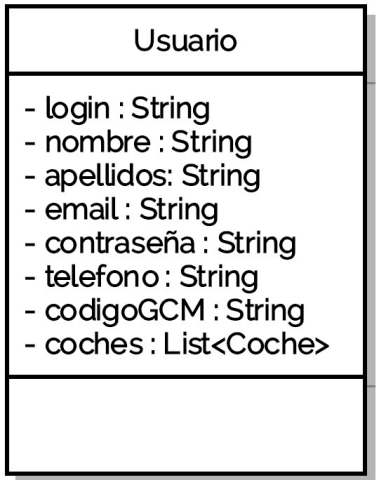

Figura A.4: Clase Usuario

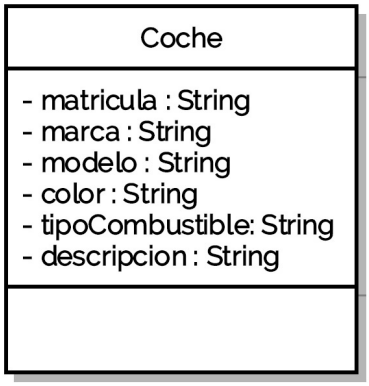

Figura A.5: Clase Coche

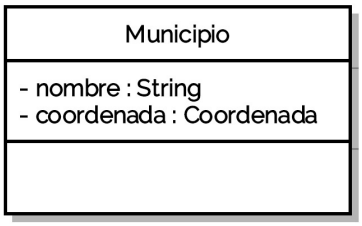

Figura A.6: Clase Municipio

Figura A.7: Clase Parada

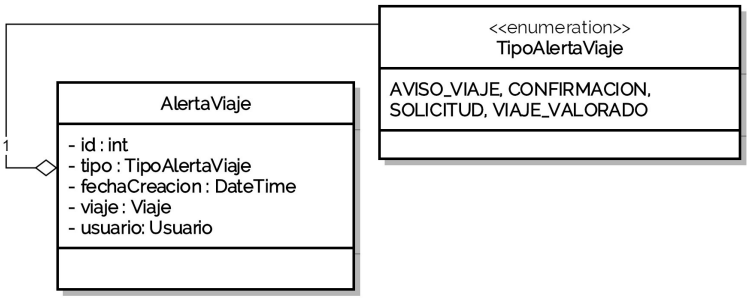

Figura A.8: Clase AlertaViaje

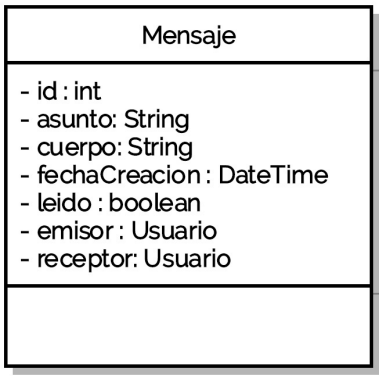

Figura A.9: Clase Mensaje

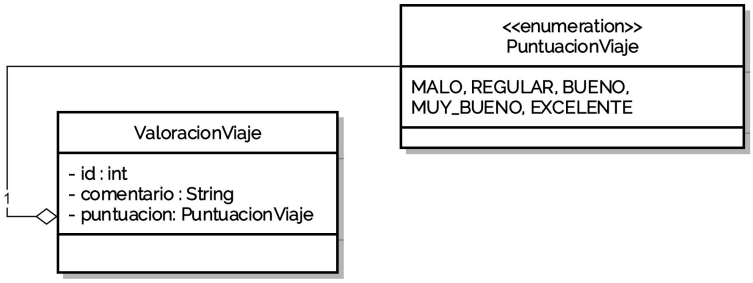

Figura A.10: Clase ValoracionViaje

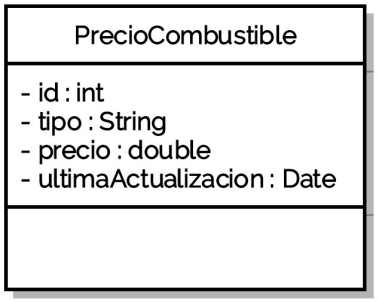

Figura A.11: Clase PrecioDeViaje

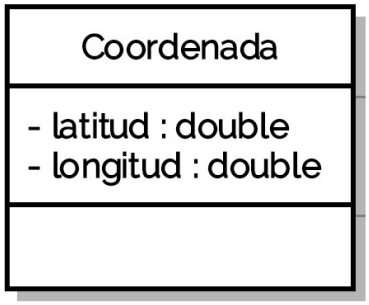

Figura A.12: Clase Coordenada

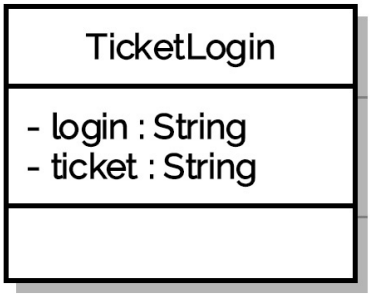

Figura A.13: Clase TicketLogin

Anexo B

Diagramas de clases del servidor Java y cliente Android

En este anexo se va a exponer los diagramas de clases en detalle del servidor Java y cliente Android.

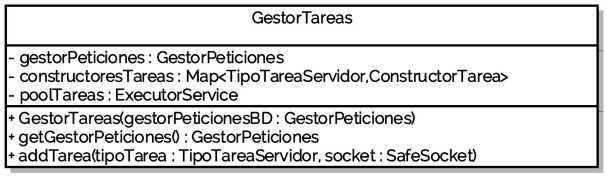

Figura B.1: Clase GestorTareas

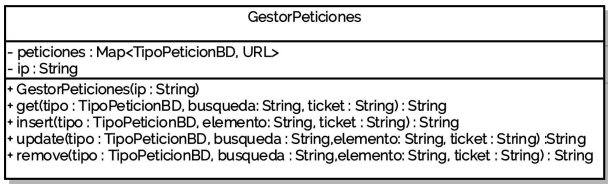

Figura B.2: Clase GestorPeticiones

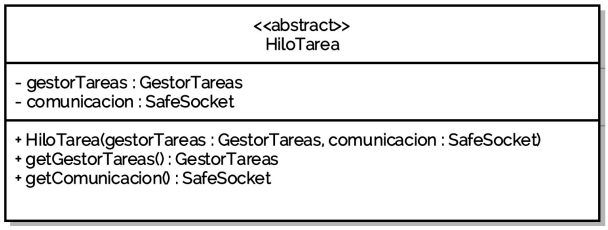

Figura B.3: Clase HiloTarea

Figura B.4: Interfaz ConstructorTarea

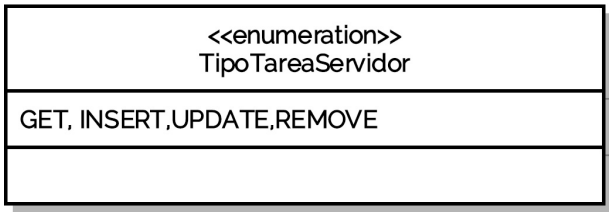

Figura B.5: Clase TipoTarea

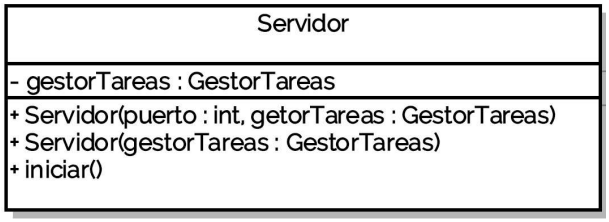

Figura B.6: Clase Servidor

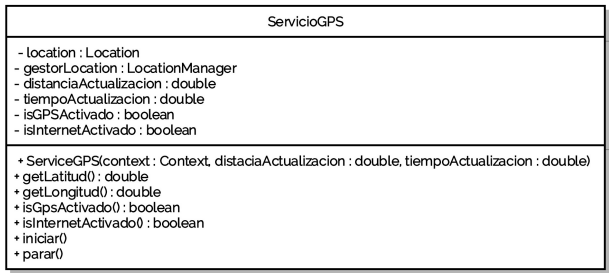

Figura B.7: Clase ServicioGPS

Anexo C

Interfaces de la aplicación web

En este anexo se va a exponer las interfaces de la aplicación web.

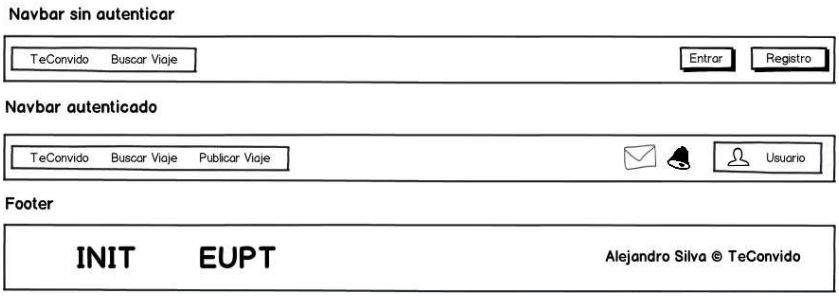

Figura C.1: Navbar y Footer

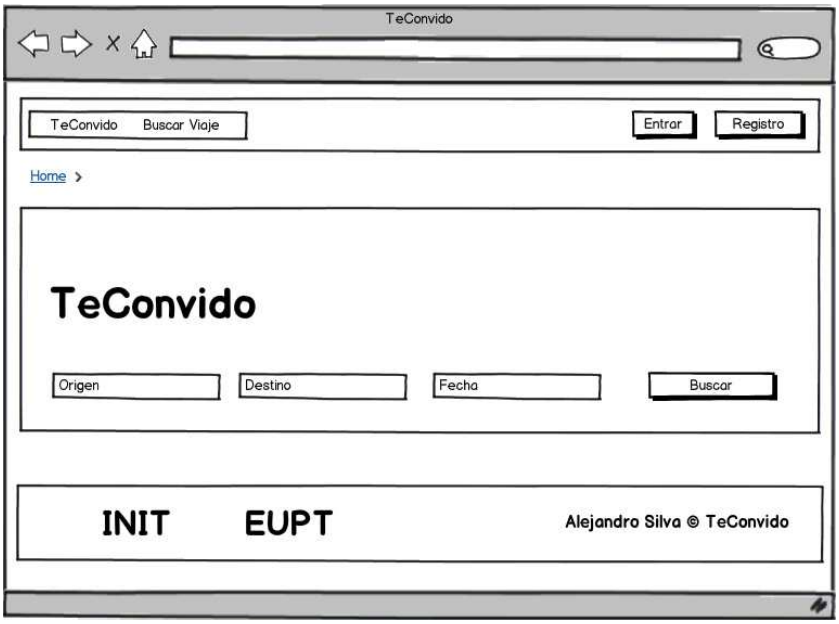

Figura C.2: Interfaz principal

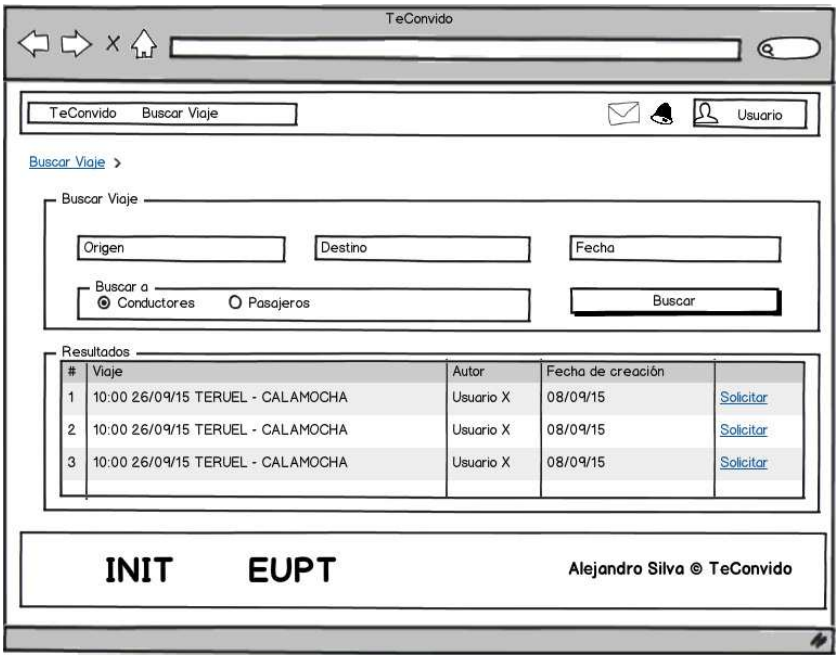

Figura C.3: Interfaz buscar viaje

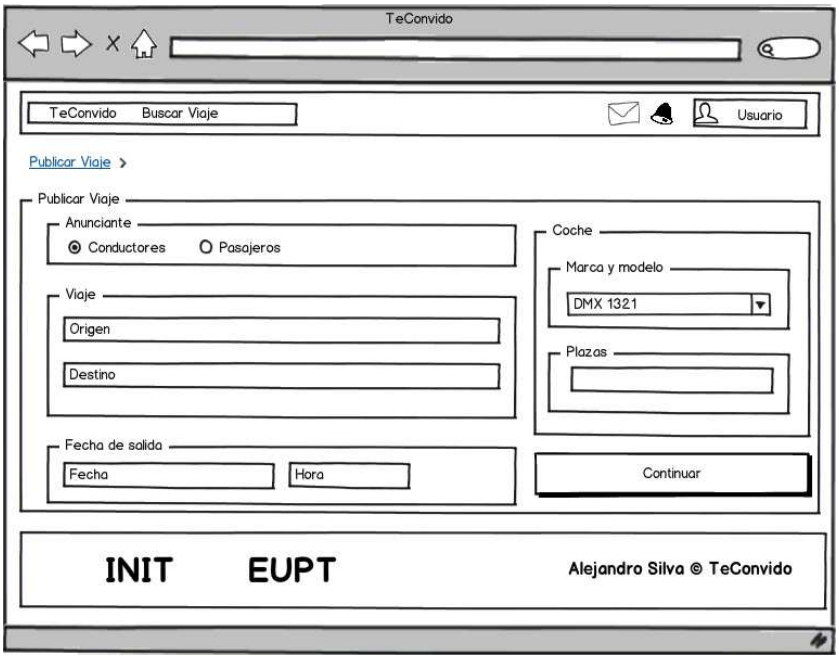

Figura C.4: Interfaz publicar viaje (parte 1)

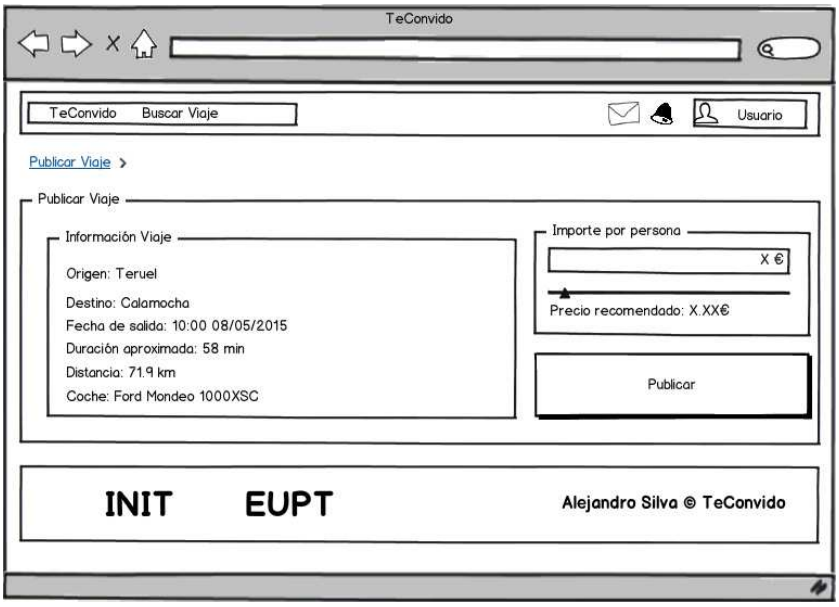

Figura C.5: Interfaz publicar viaje (parte 2)

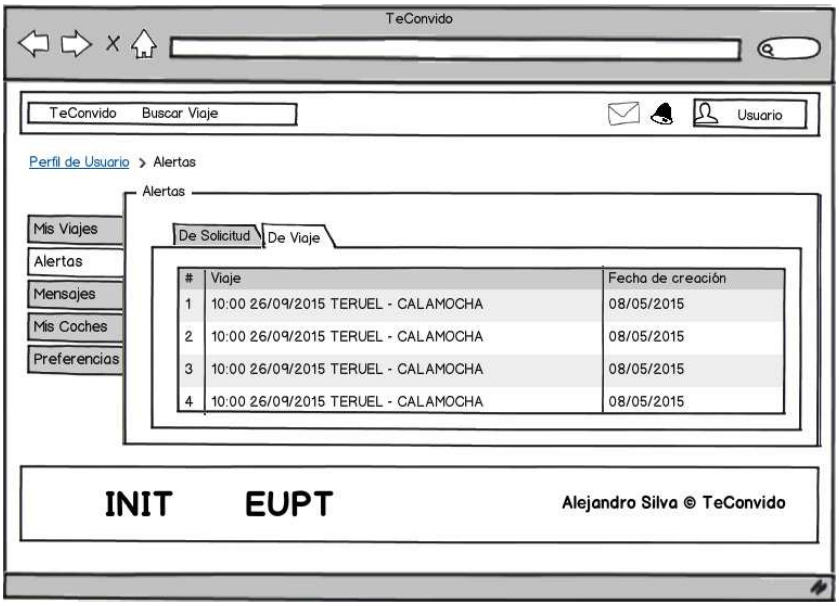

Figura C.6: Interfaz alertas

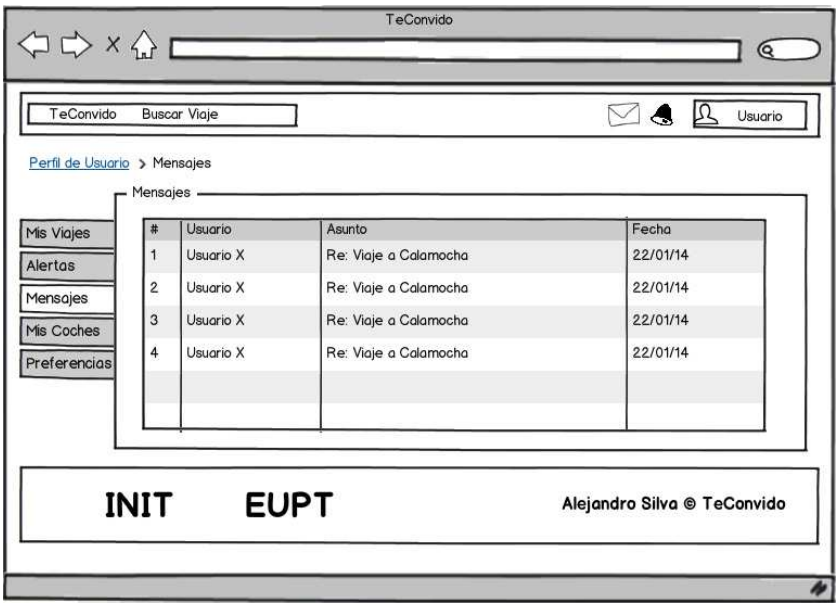

Figura C.7: Interfaz mensajes

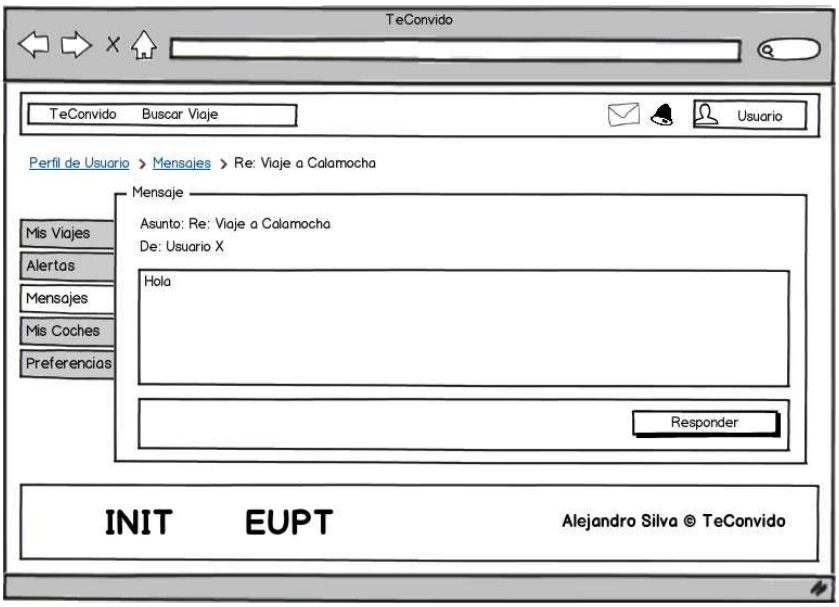

Figura C.8: Interfaz mensaje

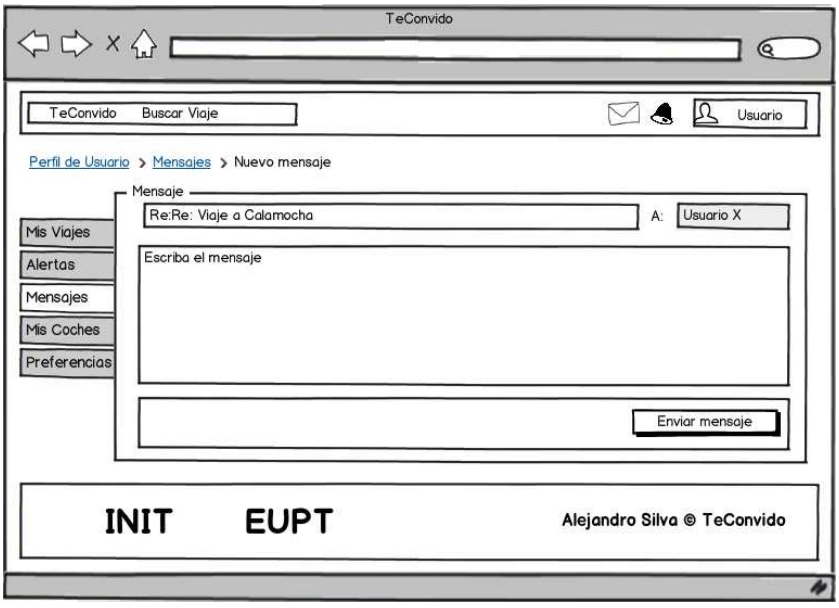

Figura C.9: Interfaz enviar mensaje

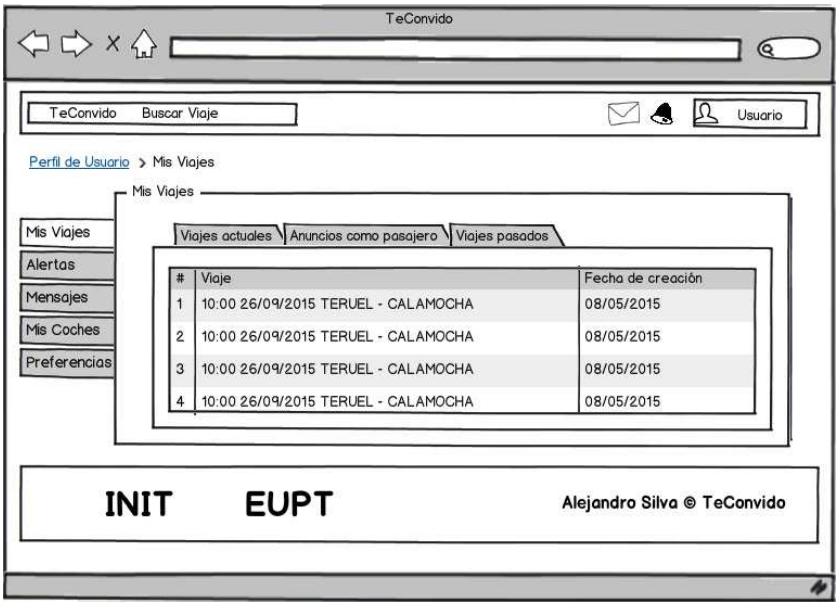

Figura C.10: Interfaz mis viajes

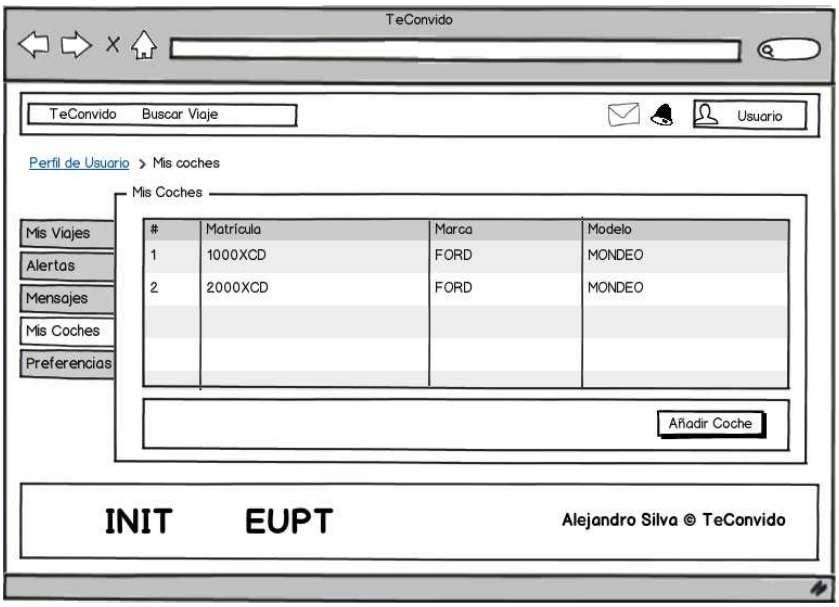

Figura C.11: Interfaz mis coches

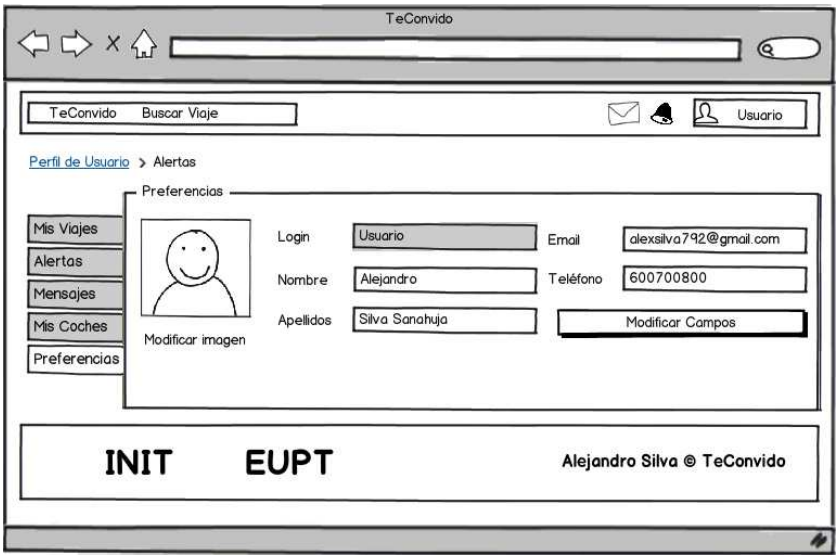

Figura C.12: Interfaz preferencias

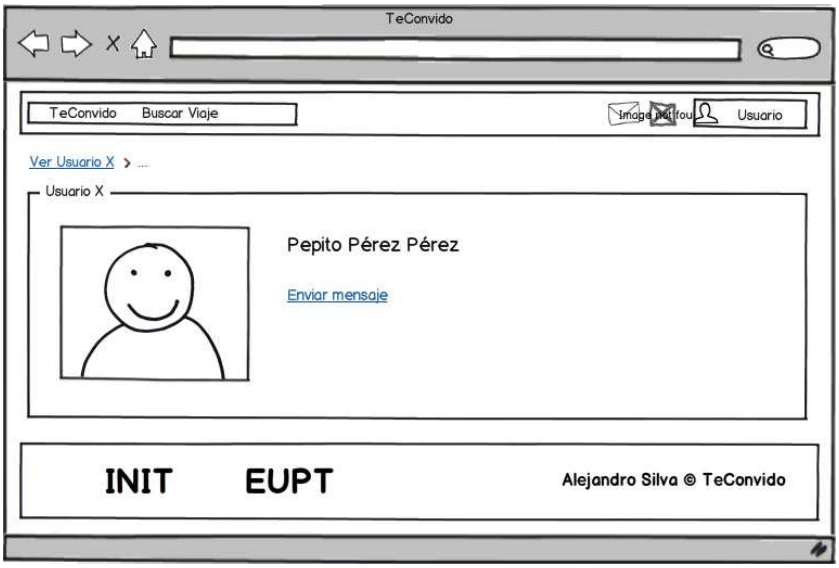

Figura C.13: Interfaz usuario

Figura C.14: Interfaz viaje

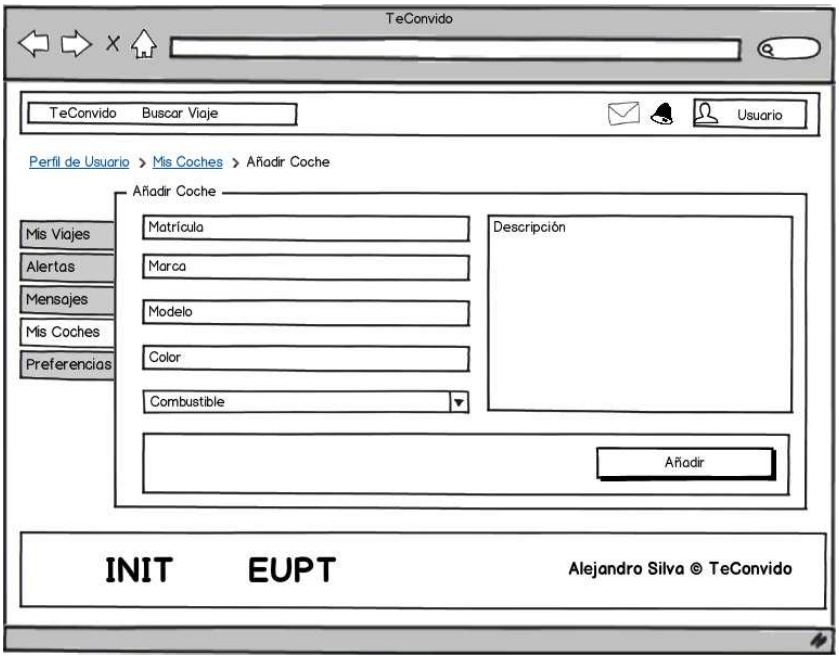

Figura C.15: Interfaz añadir coche

Anexo D

Interfaces de la aplicación Android

En este anexo se va a exponer las interfaces de la aplicación Android.

Figura D.1: Interfaz pantalla principal

Figura D.2: Interfaz menu principal

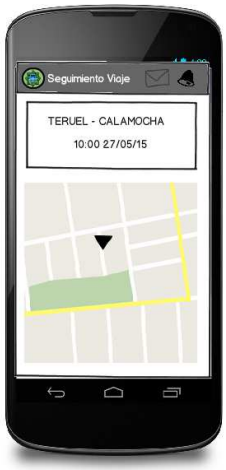

Figura D.3: Interfaz seguimiento viaje

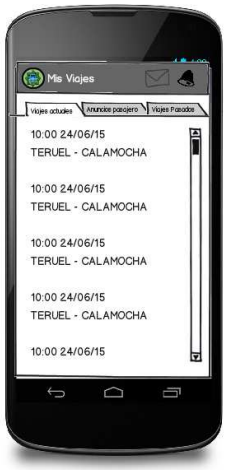

Figura D.4: Interfaz mis viajes

Figura D.5: Interfaz buscar viaje

Figura D.6: Interfaz publicar viajes

Figura D.7: Interfaz perfil

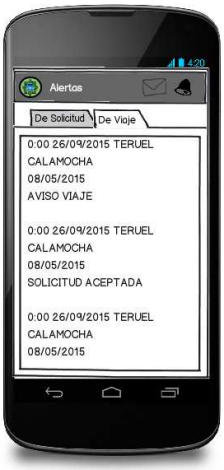

Figura D.8: Interfaz alertas

Figura D.9: Interfaz mensajes

Figura D.10: Interfaz usuario

Figura D.11: Interfaz viaje

Anexo E

Obtención de poblaciones de una ruta

Para obtener las poblaciones de una ruta, se utiliza la información contenida en la base de datos con las coordenadas geográficas de las poblaciones. En primer lugar, se obtiene la ruta a través de *Google Maps Directions API* [12]. Para obtener la rutas tenemos que hacer una petición GET a la url que nos proporciona esta API con una serie de parámetros.

Una vez tenemos todos los puntos de una ruta, utilizando la teoría de planos, se va a obtener una consulta con la cual dado dos puntos de una ruta se obtenga todas las poblaciones que se encuentran en la sección definida por esos dos puntos y la distancia de desviación.

En primer lugar, a partir de la ecuación de la recta:

$$y = y_0 + m(x - x_0)$$

Despejando obtenemos:

$$y - mx + (mx_0 - y_0) = 0$$

Dada la Figura E.1, tenemos dos rectas paralelas a la recta r, s y s'. Sustituyendo los coeficientes de x, y y el término independiente, por lo términos A, B y C , se obtiene lo siguiente:

$$A = -m \quad B = 1 \quad C = mx_1 - y_1$$

$$r : Ax + By + C = 0$$

$$s : Ax + By + C' = 0$$

Despejamos el valor C' con la fórmula de distancia entre dos rectas, dónde d es la desviación:

$$d(r,s) = \frac{|C - C'|}{\sqrt{A^2 + B^2}} \Longrightarrow \begin{cases} C' = mx_0 - y_0 + d\sqrt{m^2 + 1} \\ C' = mx_0 - y_0 - d\sqrt{m^2 + 1} \end{cases}$$

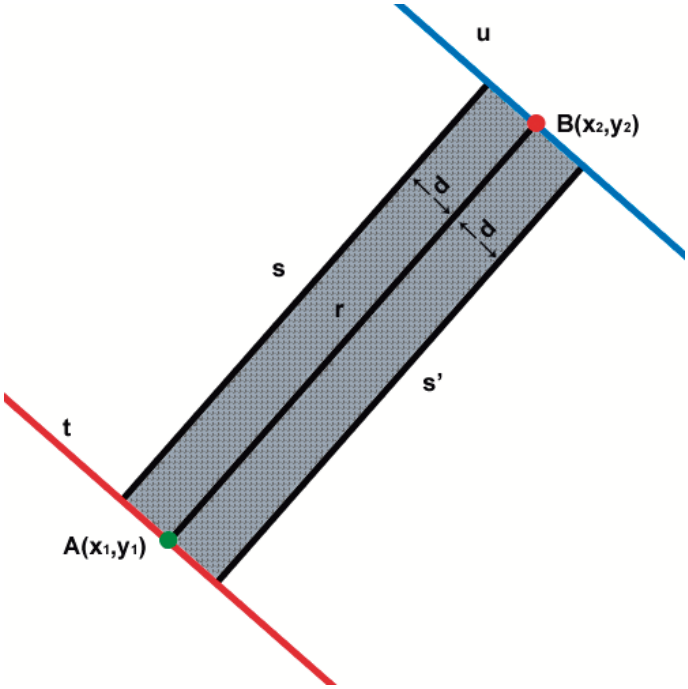

Figura E.1: Diagrama de una sección delimitada por cuatro rectas

Sustituyendo el valor *C'*, que se ha despejado en el paso anterior, se obtiene las siguientes ecuaciones para las rectas *s* y *s'*:

$$s : -mx + y + mx_1 - y_1 + d\sqrt{m^2+1} = 0$$

$$s' : -mx + y + mx_1 - y_1 - d\sqrt{m^2+1} = 0$$

Se calculan las ecuaciones de las rectas *t* y *u* sabiendo que la multiplicación de las pendientes de dos rectas perpendiculares es -1:

$$n = -1/m$$

$$t : y - nx + (nx_1 - y_1) = 0$$

$$u : y - nx + (nx_2 - y_2) = 0$$

Con las cuatro rectas (*s*, *s'*, *t* y *u*), se obtienen la sección delimitada por estás:

$$t : y - nx + (nx_1 - y_1) > 0$$

$$u : y - nx + (nx_2 - y_2) < 0$$

$$s : -mx + y + mx_1 - y_1 + d\sqrt{m^2+1} > 0$$

$$s' : -mx + y + mx_1 - y_1 - d\sqrt{m^2+1} < 0$$

Incorporando estas inecuaciones al *WHERE* de una consulta de todos los municipios, obtenemos los municipios que están definidos entre dos coordenadas y una desviación.