Universidad Rey Juan Carlos

Escuela Superior de Ingeniería Informática

Ingeniería Informática

Proyecto de Fin de Carrera

Diseño, Implementación y Despliegue de una Aplicación de Entretenimiento en Móviles Android: El Rey de la Colina

Autor: Mario Gijón Soria
Tutor: Gregorio Robles

Índice general

Agradecimientos

A mis padres, amigos y familiares. A Gregorio, y a la Universidad Rey Juan Carlos.

Capítulo 1

Resumen

El presente proyecto fin de carrera consiste en la realización de una aplicación para teléfonos móviles y dispositivos que cuentan con el sistema operativo Android, con el objetivo de presentar finalmente una aplicación que permite a los usuarios participar en un juego denominado *Rey de la colina*, mediante dicho terminal móvil.
Para ello, y en un alto nivel, se dispone de un servidor de juegos, que es el encargado de albergar las estructuras de datos y servicios web necesarios para ofrecer las funcionalidades del juego, mientras que, por otra parte, se dispone de la aplicación móvil que ejecuta en los teléfonos.
Mediante un sistema de comunicación entre los clientes, en este caso los dispositivos con la aplicación móvil, y el servidor, una máquina que ofrece los servicios web y contiene las estructuras de datos, se establecen comunicaciones para el desarrollo de las partidas del juego.
El servidor utiliza una base de datos de geolocalización con tecnología PostGIS, y ha sido desarrollado con Django. El servicio web es servido con un servidor apache, y sus clientes son, por un lado, navegadores web, y por otro, terminales Android. La comunicación entre todos ellos se realiza mediante HTTP y el intercambio de datos se hace con el formato JSON mediante conexión a Internet.
Parece razonable desarrollar aplicaciones de este tipo, ya que actualmente las compañías de telefonía móvil están apostando y compitiendo en el mercado por ofrecer teléfonos 'Smartphones' a sus clientes. Por otra parte, las posibilidades que existen actualmente a la hora de utilizar estos terminales son cada vez más amplias.
Por ejemplo, es bastante común utilizar el dispositivo para localización GPS, orientarse en un mapa, acceder a servicios web en un navegador, utilizar aplicaciones de ocio, como juegos, así como mantener una comunicación con amigos, compañeros y familiares en las diversas redes sociales que existen.

La propuesta de este proyecto es, utilizar una red social como libregeosocial, con la capacidad de geolocalización que proporciona, junto a la posibilidad de geolocalizar a los usuarios de un terminal con Android, para realizar un juego que consiste en una adaptación de lo que en algunos videojuegos RPG se ofrece en partidas multijugador online bajo el nombre de *rey de la colina*, *captura la bandera* o *battlegrounds*.
Además, se incluye un sistema de mensajería interno para facilitar la comunicación entre todos los jugadores de las diferentes partidas, y es, por tanto, éste proyecto una buena forma de aplicar los conocimientos aprendidos durante la carrera para desarrollar un sistema informático que utilice las tecnologías que están a la orden del día.

Capítulo 2

Introducción

En esta parte de la memoria se tratarán diferentes puntos que son necesarios conocer de antemano para poder entender los siguientes. En concreto, se van a explicar las diferentes tecnologías que se utilizan, así como la dinámica del juego y los antecedentes históricos de juegos de este tipo.

2.1. Motivación

El principal motivo de realizar el siguiente PFC es que durante toda la carrera no se ha tenido la oportunidad de construir aplicaciones o sistemas que vayan a ser ejecutados en un terminal móvil. Esto presenta un nuevo reto que hasta ahora no se ha afrontado en las diferentes asignaturas de la carrera.
Durante la carrera, se han hecho asignaturas de redes, en las que se ha tenido que programar algún servidor web y se ha estudiado el funcionamiento de Internet, así como asignaturas relacionadas con bases de datos, sistemas operativos, y programación.
En cambio, éste PFC implica construir un sistema en el que gran parte de los aspectos vistos durante éstos años de estudio deben estar reunidos, lo cual resulta bastante interesante. Además, habrá clientes que no sean computadoras, sino que son terminales móviles, los cuales presentan una serie de limitaciones y nuevos problemas a resolver que no se han tratado con anterioridad.
Por tanto, con este PFC se pretendía aprender nuevos lenguajes de programación, como por ejemplo Python, y ampliar conocimientos de Java, al tener que utilizar librerías de Android. También se pretendía aprender cómo funciona el sistema operativo Android, y las posibilidades que ofrece a la hora de construir una aplicación cuya finalidad principal es entretener a los usuarios.

En conclusión, este PFC reúne diversas disciplinas estudiadas durante la carrera en un único sistema, a la vez que sirve para aprender muchas tecnologías nuevas y para aprender a resolver nuevos problemas que no se han visto hasta ahora.
La utilidad final del proyecto es mas didáctica que lúdica, ya que se adquirirán conocimientos nuevos, y se harán pruebas reales con los resultados obtenidos, aunque también se desarrolla un sistema que permitirá a los usuarios finales disfrutar de un juego enmarcado en lo que se conoce como 'Outdoor Gaming', que se explica mas adelante.

2.2. Outdoor Gaming

El proyecto comienza partiendo de la base de que es necesario adaptar un juego en el que los aspectos fundamentales son:

- Disponer de un espacio en el que desarrollar la actividad.
- Ajustar el juego a un tiempo razonable de juego.
- Establecer una serie de objetivos comunes para que varios equipos compitan por ellos.
- Proporcionar medios para que los equipos se comuniquen y puedan desarrollar una estrategia de forma dinámica, y a tiempo real en función de cómo transcurre la partida.

Los juegos al aire libre, conocidos como *Outdoor Gaming*, son los que mejor se ajustan a esta propuesta.A continuación se detallan algunos de los juegos más conocidos que ya existen en la actualidad.

2.2.1. Alleycat races

Según el artículo de Wikipedia[1], consiste en una carrera en bicicleta, en la que cada uno de los participantes debe alcanzar diversos *checkpoints*, también conocidos como puntos de control, o CP, en los cuales se les dan las directrices para alcanzar el siguiente, y así sucesivamente, hasta alcanzar la meta final. Es destacable que en alguno de los tramos de la carrera, se pueda pedir al concursante que realice alguna tarea concreta, de tal manera que para poder recibir las directrices del siguiente tramo, necesite haber validado la tarea en el CP.
Esto es parecido a una *gymkhana*, pero con el añadido de que los participantes deben realizar ejercicio físico. Es bastante común en muchos lugares de

todo el mundo, pero suele ser necesario un permiso de las autoridades locales para poder organizar este tipo de eventos, y que las carreras sean seguras evitando posibles accidentes o disturbios.
El hecho de querer llevar este tipo de juego a cabo no es viable para nuestro propósito, ya que lo que se pretende es la utilización de una red social móvil, y este tipo de competición es individual, además del peligro que puede suponer utilizar un teléfono mientras se conduce un vehículo, en este caso, una bicicleta.

2.2.2. Geocaching

Como se explica en el sitio web de Geocaching en castellano [2], en este caso los participantes, de forma individual, o en equipo, utilizan un sistema de GPS para encontrar tesoros que han sido escondidos por todo el Planeta. La palabra *geocaching* viene de *Geo*, que quiere decir Geografía, y *Caching*, que desde el punto de vista informático se trata de la utilización de una memoria para recuperar información más rápido, pero que en inglés se utiliza en temas de excursionismo para referirse a un lugar donde ocultar provisiones.
El funcionamiento del juego es sencillo. Consiste en esconder algún objeto o *tesoro* en algún lugar, de forma que sea difícil de encontrar, pero no imposible. Acto seguido, publicar en una página dedicada a este juego las coordenadas GPS de la ubicación del tesoro, y esperar a que alguien lo encuentre y publique en el foro sus impresiones. La recompensa, además del tesoro, es poder disfrutar del lugar donde esté escondido, por ejemplo, un buen lugar para acampar, una bonita vista de algún paisaje, o un lugar extraño o con alguna historia relacionada. Como norma general, siempre que se encuentre un tesoro, se debe dejar un nuevo tesoro en su lugar, para la próxima persona que llegue a dar con él. De este modo se lleva a cabo la interacción entre jugadores. Se han llegado a encontrar tesoros en forma de CD, dinero en metálico, joyas, e incluso lingotes de oro. Como norma general, se recomienda que los tesoros tengan un valor de unos 10 euros de media, aunque se puede dejar lo que el jugador pueda, o desee.

2.2.3. Geodashing

Esta versión parecida al 'geocaching' consiste en visitar diferentes puntos geográficos, y comentar lo que el jugador ha observado y sentido al llegar a ese punto. En este caso, los lugares son mostrados al jugador de manera aleatoria. El objetivo del juego es visitar el mayor número de puntos posibles. Esta información se ha obtenido de la pagina web de geogashing[3].

2.2.4. Highpointing

El objetivo de este juego es sencillo. Dada un área concreta en la cual se desarrollará el juego, se debe encontrar el punto más alto posible, tal y como explican en la página web del juego[4].

2.2.5. Letterboxing

Del mismo modo que en el 'geocaching', se revelan diversas coordenadas GPS en las cuales se indica la ubicación de una caja llamada 'letterbox'. Estas cajas deben ser colocadas en lugares públicos y accesibles, de tal forma que pasen desapercibidas totalmente para cualquier persona que pase por allí, pero no para un jugador de 'letterbox'. Estas cajas deben estar hechas de tal manera que no se rompan fácilmente, y sean resistentes al agua. Su contenido será un matasellos, de tal forma que quien encuentre una caja, podrá estampar el sello en alguna tarjeta o papel, como prueba de que la encontró. Adicionalmente, se pueden encontrar dentro alguna serie de pistas para encontrar una caja extra. Una vez finalizado el hallazgo, el jugador se encargara de dejar todo exactamente como estaba, para garantizar la continuidad del juego. La normativa completa y mas información se puede encontrar en su página web[5].

2.3. El rey de la colina

2.3.1. Antecedentes históricos

El rey de la colina (en inglés king of the hill), también conocido como rey de la montaña.º rey del castillo", es un juego cuyo objetivo es mantenerse en la cima de una gran colina o montón (o cualquier otra zona designada), como el rey de la colina". Otros jugadores tratan de echar al actual rey fuera del montón y tomar su lugar, convirtiéndose así en el nuevo rey de la colina. La forma en que el rey"puede ser eliminado de la colina depende, en gran medida, de las normas establecidas por los jugadores antes de que empiece el juego. El nombre del juego se ha convertido en una metáfora para cualquier tipo de juego competitivo de suma cero o actividad social en la que un único ganador es elegido entre varios competidores, y una jerarquía es elaborada por las cimas que los competidores alcanzan en la colina, y donde ganar sólo puede lograrse a costa del desplazamiento del anterior ganador. El juego también prestó su nombre a 'King of the Hill', una serie de animación americana emitida por la cadena Fox[6].
Suma cero describe una situación en la que la ganancia o pérdida de un

participante se equilibra con exactitud con las pérdidas o ganancias de los otros participantes. Se llama así; porque si se suma el total de las ganancias de los participantes y se resta las pérdidas totales el resultado es cero. El ajedrez, el póker y el juego del oso son ejemplos de juegos de suma cero. La suma cero es un caso especial del caso más general de suma constante donde los beneficios y las pérdidas de todos los jugadores suman el mismo valor, porque se gana exactamente la cantidad que pierde el oponente.
El rey de la colina se ha presentado como una variante de juego en muchos videojuegos, sobre todo en juegos de disparos en primera persona como 'Halo: Combat Envolved'[7] y el más tradicional 'Perfect Dark'[8]. En estas versiones del juego, un jugador o equipo de jugadores deben mantener el control de un área específica u objeto durante una cantidad predeterminada de tiempo. Cuando esa cantidad de tiempo es alcanzada, la ronda finaliza o una nueva área es designada en el mapa. En la variante virtual, los jugadores son, por lo general, eliminados de la colina matándolos.

2.3.2. Dinámica del juego y adaptación

En cuanto al tablero o mapa, se establece una zona de juego suficientemente amplia en función de los jugadores y equipos que participen en la partida, de tal manera que hay una serie de ubicaciones identificadas mediante una bandera en el mapa, que se han de capturar.
En cuanto a los equipos, el número de equipos es ilimitado, siendo como mínimo siempre dos. En cada equipo debe haber como mínimo un jugador, siendo también ilimitado el número de jugadores en un equipo. Todo equipo se identificará mediante un color, que dará sentido a la hora de ver las banderas que son conquistadas por cada equipo.
En cuanto a las puntuaciones, al comienzo de la partida, todas las banderas estarán en un estado neutral, es decir, no puntúan a favor de ningún equipo. Del mismo modo, las puntuaciones de cada equipo empezarán con valor cero. Una vez que un equipo haya capturado una bandera, dicha bandera adoptará el color del equipo que la haya conquistado. A partir de ese momento, por cada intervalo de tiempo previamente fijado que dicha bandera conserve ese color, sumara 1 punto al equipo correspondiente. Para capturar una bandera, basta con que un jugador de un equipo permanezca en el lugar donde se ubica. Para robar una bandera, y cambiar su color al del equipo al que pertenece el jugador que roba, debe permanecer junto a ella. Será, por tanto, ganador, el equipo que consiga alcanzar una puntuación previamente fijada al inicio de la partida.
En cuanto a la comunicación y la estrategia, cada equipo conoce la posición de sus compañeros en el mapa en todo momento, así como la ubicación de

todas las banderas en el mapa, y su color actual. Desconoce la posición de los jugadores rivales. El envío de mensajes entre jugadores se podrá realizar en cualquier momento, para establecer estrategias a tiempo real relacionadas con el juego, como por ejemplo informar de que durante un tiempo un jugador se va a quedar en algún lugar sin moverse, o para informar del próximo punto al que va a dirigirse para intentar capturar bandera, o bien para informar de la presencia de jugadores enemigos en algún punto.

2.3.3. ¿Qué se fomenta con éste juego?

Con este juego se fomentan una serie de valores, que son los siguientes

- La participación en un juego competitivo, y los valores que la competición aporta como si se tratara de cualquier deporte.
- La integración de los jugadores en un equipo, en el que todos deben colaborar entre sí, para poder lograr la victoria en conjunto, es decir, enseña a trabajar en equipo.
- Tomar decisiones en función de cómo se desarrolle la partida, esto es, enfrentarse a situaciones que en cada partida serán diferentes y que sólo dependen del propio desarrollo del juego.
- Familiarización con el uso de dispositivos de una forma que no es habitual, como es el uso de un teléfono para participar en un juego colectivo, en lugar de utilizar las funcionalidades más comunes.
- Divertirse.

2.4. Android

Según la definición que se da en la documentación oficial[9]: 'Android es un conjunto de software para dispositivos móviles que incluye un sistema operativo, un middleware, y aplicaciones clave. La SDK de Android proporciona las herramientas y las API's necesarias para empezar a desarrollar aplicaciones para la plataforma Android, mediante el uso del lenguaje de programación Java.'
Esto significa que Android incluye un sistema operativo que permite gestionar los recursos de un dispositivo móvil, además de una máquina virtual llamada Dalvik, sobre la que ejecutan las diferentes aplicaciones del terminal, así como un conjunto de aplicaciones, entre las cuales se encuentran,

Figura 2.1: Logotipo de Android

por ejemplo, un navegador web, un gestor de e-mails, de contactos, de llamadas, etcétera. Todas estas aplicaciones han sido desarrolladas con el lenguaje Java y dan las funcionalidades principales y características de los teléfonos que incluyen Android. La idea de ofrecer servicios de Internet en dispositivos móviles se remonta hasta los mediados de los años 90. Sin embargo, no ha sido hasta recientemente cuando se han tenido móviles con acceso a Internet en el mercado. Gracias a los productos más novedosos, como la salida al mercado del Iphone, se ha producido el aumento de popularidad de estos teléfonos. En este sentido, es donde cobra importancia Android, ya que es uno de los principales sistemas con los que cuentan los teléfonos con acceso a Internet, también conocidos como 'Smartphones'. El logotipo de Android se puede ver en la figura 2.1.

A la hora de realizar aplicaciones para teléfonos, se deben tener en cuenta algunos problemas característicos de la programación para este tipo de dispositivos, como por ejemplo:

- La pantalla que va a ver el usuario es muy pequeña, de tan solo unas pulgadas, en lugar de tener una pantalla del tamaño de un monitor.
- Los teclados, si es que existen, van a ser muy pequeños. Este teclado puede ser un conjunto de botones hardware que proporcione el aparato, o bien será un teclado limitado.
- Algunos dispositivos permiten la entrada de datos desde una pantalla táctil, y estas tienen un determinado margen de error.
- La velocidad de procesamiento es limitada, al igual que el tamaño de memoria, en comparación con la que se dispone en computadoras corrientes.

Los principales componentes de una aplicación Android son los siguientes:

- Actividades: Son bloques en los que se construyen las interfaces de usuario. Se puede ver una actividad de forma análoga a una ventana o cuadro de diálogo en una aplicación para computador. Aunque es posible construir actividades que no tengan interfaz de usuario, lo habitual es construir servicios o proveedores de contenido para esto.

- Proveedores de contenido: Estos bloques proporcionan un nivel de abstracción para los datos que se almacenan en el dispositivo, y que pueden ser accedidos por múltiples aplicaciones.

- Servicios: Los anteriores bloques son considerados para tener un corto tiempo de vida, sin embargo, los servicios están pensados para ejecutar independientemente de las actividades. Sería necesario, por ejemplo, comprobar actualizaciones de una fuente de noticias RSS o mantener un reproductor de música activo, aunque la actividad que lo controla no esté operando.

- Intents: Estos bloques son mensajes del sistema, ejecutando en el teléfono, y que se encargan de notificar eventos a las a aplicaciones. Estos eventos pueden ser cambios en el estado del hardware, como la inserción de una tarjeta SD, datos que llegan, como un SMS, etcétera. Lo habitual es utilizar Intents para lanzar actividades nuevas cuando se produzca algún evento que interese.

Del mismo modo, Android proporciona una serie de características que ayudan al desarrollo de aplicaciones:

- Almacenamiento: Junto a la aplicación, se pueden empaquetar elementos que no deben cambiar, como por ejemplo iconos o ficheros de ayuda. Del mismo modo, se permite acceder al almacenamiento que ofrecen las tarjetas de memoria SD, para la lectura y escritura de los ficheros que puedan ser necesarios.

- Network: Dado que los dispositivos Android generalmente están preparados para acceder a Internet, se puede aprovechar esto para realizar aplicaciones que requieran comunicación con otros dispositivos, a la vez que se permite la programación a nivel de red que proporciona Java, como son las librerías de java.net.

- Multimedia: Los móviles con Android tienen la capacidad de reproducir y grabar video y audio, por lo que las aplicaciones Android pueden tener acceso a la reproducción de archivos de sonido, a la cámara para realizar fotografías, o al micrófono que incorporan los teléfonos.

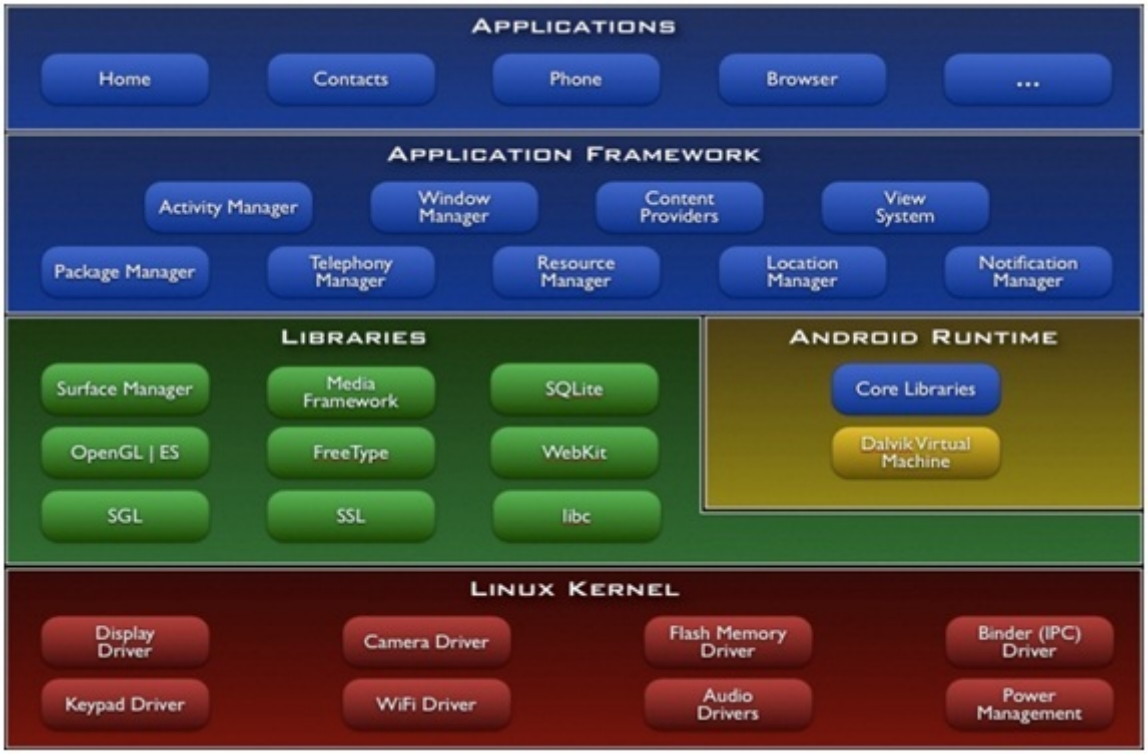

Figura 2.2: Arquitectura software de Android

- Servicios de telefonía: Dado que las aplicaciones son ejecutadas en un teléfono, se puede tener acceso al servicio de mensajes cortos SMS, se pueden realizar llamadas telefónicas, gestionar los contactos de la tarjeta SIM, y cualquier necesidad propia de un teléfono.

El núcleo del sistema operativo Android es Linux 2.6, y se encarga de la gestión de los recursos del terminal, como son: procesador, memoria, cámara, teclado, pantalla... Además gestiona el consumo de la batería, mantiene la integridad del sistema de ficheros, la seguridad y los permisos del mismo, gestiona los procesos, gestionar la red, etcétera. Por encima de este núcleo se encuentran las librerías de Android, así como su núcleo, y la máquina virtual Dalvik. Esta es la parte a la que se llama middleware. Estas librerías son utilizadas por las aplicaciones propias del sistema operativo Android, que son accesibles mediante la API, y por tanto, diseñadas para ser reutilizables. En el mayor nivel, se encuentran las aplicaciones que son descargadas, instaladas, ejecutadas y eliminadas a gusto del usuario, que permiten realizar diferentes tareas con el dispositivo. Algunas de estas aplicaciones vienen de serie con Android, como por ejemplo el gestor de contactos, el navegador web, gestor de e-mails, etcétera. La figura2.2 explica de forma gráfica la arquitectura anteriormente explicada. Para la realización de este PFC, se han cosultado varios libros[10, 11, 12] que ayudan al aprendizaje de Android.

2.5. Python

Python[13] es un lenguaje de programación de alto nivel cuya idea principal es mantener una sintaxis muy limpia, y por tanto producir finalmente un código muy legible. Este lenguaje de programación es multiparadigma, ya que permite la programación orientada a objetos, así como la programación imperativa y funcional. Su logotipo se puede observar en la figura 2.3
Se trata de un lenguaje de programación interpretado, por lo que todo el código no necesita ser compilado previamente, sino que se va compilando y ejecutando línea por línea. Utiliza un tipado dinámico, y es fuertemente tipado. Además, es un lenguaje multiplataforma.
Este lenguaje de programación tiene una licencia de código abierto, y fue creado a finales de los años 80 por Guido Van Rossum en el National Research Institute for Mathematics and Computer Science, ubicado en Los Países Bajos. El nombre viene de la afición del creador por los humoristas Monty Python.

Figura 2.3: Logotipo de Python

Figura 2.4: Logotipo de Django

2.6. Django

Django[14] es un framework de desarrollo de contenido web que está escrito en el lenguaje Python. Se trata de un software de código abierto que cumple con el paradigma de Modelo – Vista – Controlador. En su origen, fue desarrollado con el objetivo de gestionar páginas web orientadas a noticias de la World Company de Lawrence, en Kansas, y en el año 2005 se sacó al público bajo una licencia BSD. Su logotipo es el que se aprecia en la figura 2.4.

El objetivo principal de Django es facilitar la creación de sitios web de cierta complejidad. Propone la reutilización y conectividad de componentes, para conseguir un desarrollo rápido de los sitios web. El lenguaje de programación Python está presente en todo el framework, desde la instalación, hasta la configuración, los modelos de datos, y el lanzamiento del servidor.

Una de las buenas características que tiene este framework es la enorme API que tiene para bases de datos. Esto viene de su origen, ya que en un principio se pretendía desarrollar páginas orientadas a contenido.

Las equivalencias en el paradigma Modelo – Vista – Controlador, son las siguientes:

- Para los modelos, se han de especificar las estructuras de datos y modelos que se van a utilizar dentro del sitio web. Todo esto se guarda en el fichero 'models.py'.
- Para las vistas, se crean lo que se conoce como plantillas, que lo habitual es que sean páginas escritas en HTML.
- Los controladores en este caso se escriben en un archivo 'views.py', y

Figura 2.5: Logotipo de Apache

que incluye el código que se ha de ejecutar en el servidor cada vez que se accede a una URL del sitio. La correspondencia entre URLS y vistas se especifican en un fichero 'urls.py'.

2.7. Apache

El servidor HTTP apache[15] es un software diseñado para servir contenido HTTP de código libre. Es multiplataforma, es decir, que se puede disponer de él en sistemas Unix, Windows y Mac, entre otros. Su logotipo se puede observar en la figura 2.5. En este PFC se ha utilizado para albergar el sitio web desarrollado con Django en código Python, que sirve para gestionar las partidas del juego, así como sus equipos y jugadores, los cuales se verán en profundidad más adelante.

Apache es un servidor muy bueno a la hora de configurar los múltiples parámetros que se pueden establecer, pero no dispone de una interfaz gráfica, por lo que toda la configuración debe realizarse mediante la modificación de archivos de configuración y comandos de Shell. Desde el año 1996 es el más utilizado en Internet, y en el año 2005 era utilizado por el 70 % de los sitios web del mundo.

Este servidor tiene vulnerabilidades de seguridad detectadas, pero sólo pueden ser utilizadas por usuarios malintencionados que se encuentren en red local. Sin embargo, si se utiliza el módulo de PHP para Apache, algunas de estas vulnerabilidades sí pueden ser accedidas remotamente. En cualquier caso, este software presenta muchas ventajas, como son, entre otras, que se trata de software de código abierto, modular, multiplataforma, y muy popular, lo que hace que sea extremadamente sencillo encontrar ayuda o soporte. La arquitectura de este servidor es modular, esto es, consta de un módulo central o core con la funcionalidad principal del servidor, y diversos módulos que aportan la funcionalidad necesaria al servidor, dependiendo de las necesidades de cada sitio. En este PFC se ha necesitado incluir un módulo 'mod_wsgi' [24], que permite utilizar bases de datos que almacenen posiciones de geolocalización, como es éste caso.

Figura 2.6: Logotipo de Libregeosocial

2.8. Libregeosocial

Libregeosocial[16] es una red social móvil con una interfaz de realidad aumentada. Su logotipo se puede observar en la figura 2.6. Todos los nodos de esta red social tienen la peculiaridad de estar geolocalizados, esto es, a cada nodo se le asocia una posición de coordenadas GPS que le ubica en el mundo. Para esta localización, se dispone de tres parámetros: latitud, longitud, y altitud.

Los nodos de esta red social se pueden almacenar en una base de datos, y ofrece una API para crearlos, modificarlos, almacenarlos en una base de datos, y eliminarlos. De esta manera, se tiene la posibilidad de manejar estructuras de datos de usuarios geolocalizados, así como sus relaciones, grupos a los que pertenecen, mensajes, audio, imágenes, etcétera. Libregeosocial ha sido desarrollado por el departamento Libresoft, del grupo de sistemas y comunicaciones de la Universidad rey Juan Carlos, y cuenta con una licencia de código libre. El modelo de datos que interesa conocer para este PFC es el que se muestra en la figura 2.7. Cuenta con dos componentes básicos:

- Libregeosocial, que se trata de un marco de trabajo implementado con Python y Django, que permite crear redes sociales con geolocalización utilizando las estructuras de datos propias de la red social, y su API para manejarlas.

- La aplicación móvil de libregeosocial, que está desarrollada en Android, y que se sirve de los elementos de un teléfono con este sistema operativo para interactuar con la red social. Por ejemplo, utiliza el GPS del terminal para localizar a un usuario, o marca una instantánea tomada con la cámara y la posiciona dentro de la red social, entre otras aplicaciones. La aplicación del Rey de la colina, se basará en este modelo

de datos y la API que libregeosocial proporciona, para extenderla y generar un modelo de datos adecuado para la aplicación.

2.9. PostgreSQL

PostgreSQL[17] es un potente sistema de bases de datos objeto-relacionales de código abierto. Su logotipo es el que se muestra en la figura 2.8. Lleva en desarrollo más de 15 años, y su arquitectura ha alcanzado una gran reputación, debido a su fiabilidad, su corrección, y a la buena capacidad de mantener la integridad de los datos. Se trata de un sistema multiplataforma, ya que funciona en cantidad de sistemas operativos como, por ejemplo y entre otros, Windows, Linux. Mac, Solaris, etcétera.
Esta base de datos soporta la mayoría de los tipos de datos que se especifican en el estándar de SQL 2008, así como también soporta el almacenamiento de ficheros binarios de gran tamaño, como imágenes, audio o video, lo cual hace de este sistema de almacenamiento de bases de datos un candidato muy bueno para utilizar en este PFC.
Permite el acceso concurrente a múltiples usuarios, y la cantidad de datos que es capaz de manejar hace que sea altamente escalable. Concretamente, los límites que maneja actualmente son:

- Tamaño máximo de base de datos ilimitado.
- Tamaño máximo de de tabla, 32 TB.
- Tamaño máximo de fila, 1.6 TB.
- Tamaño máximo de campo, 1GB.
- Máximo de filas por tabla ilimitado.
- Máximo de columnas por tabla, de 250 a 1600, en función del tipo de datos.
- Máximo de índices por tabla, ilimitado.

Además, PostgreSQL es capaz de ejecutar procedimientos almacenados en más de una docena de lenguajes de programación, entre los que se incluyen Java y Python, que son dos de los lenguajes que se utilizan en este PFC, por lo que la compatibilidad está asegurada. Esto habla a favor de la capacidad de configuración que tiene PostgreSQL. Por último, dispone de cantidad de extensiones y características avanzadas, y se debe mencionar en este caso el

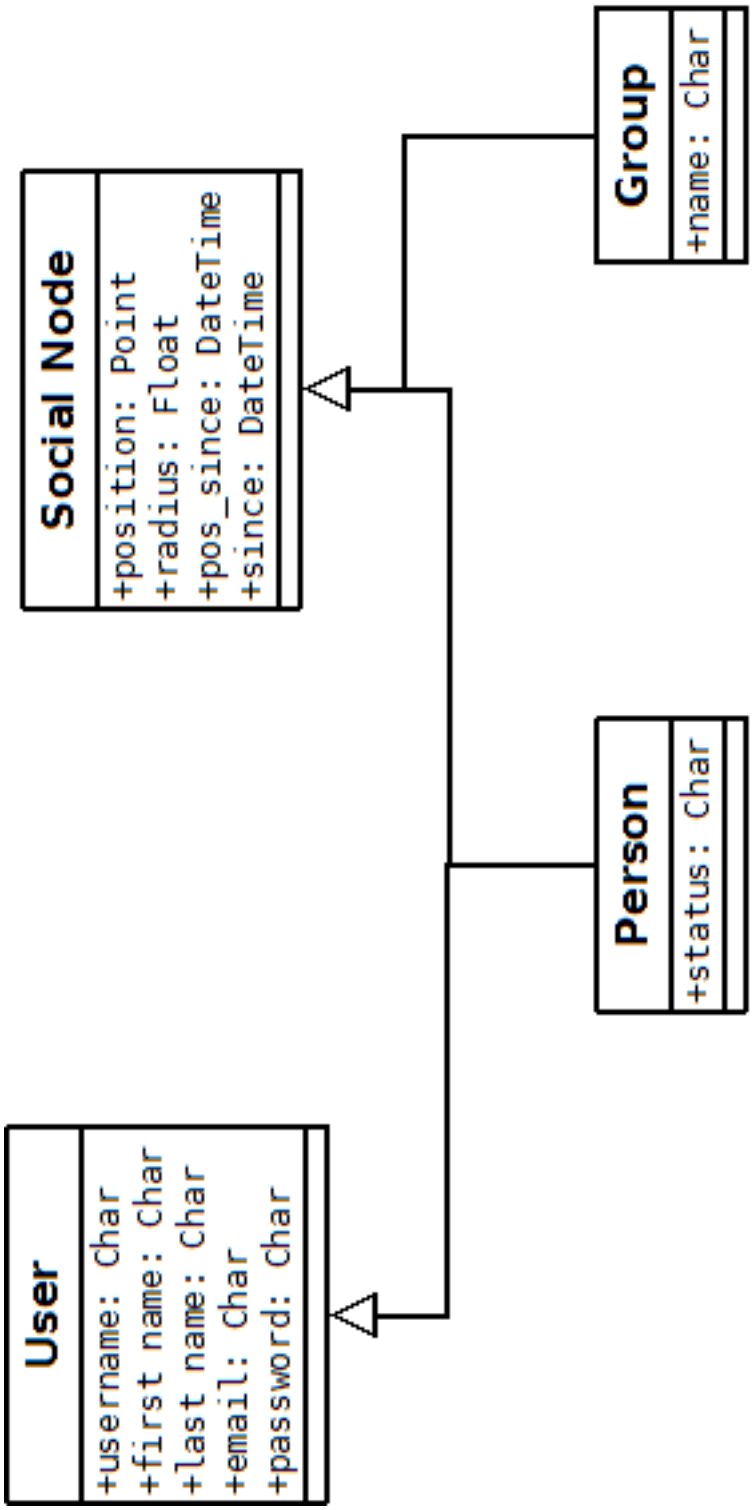

Figura 2.7: Diagrama UML de datos de Libregeosocial

Figura 2.8: Logotipo de PostgreSQL

Figura 2.9: Logotipo de PostGIS

método de almacenamiento GiST (Generalized Search Tree), que implementa eficazmente y eficientemente el almacenamiento y búsqueda en árboles.
Concretamente, se ha tenido que utilizar la expansión PostGIS [18], que añade soporte a objetos geográficos en la base de datos, por tanto, permite el almacenamiento de información geográfica en el servidor. Este módulo, PostGIS, ha sido desarrollado bajo una licencia GNU General Public Licens, y su logotipo se puede ver en la figura 2.9.

2.10. Json

JSON (Java Script Object Notation)[19], es un formato de intercambio de datos ligero. Su logotipo se puede ver en la figura 2.10. Está desarrollado con el objetivo de que sea sencillo de leer e interpretar para un ser humano, y que también sea sencillo de procesar y generar para las máquinas. Basado en un subconjunto del lenguaje de programación Java Script, se trata de un formato de texto que es completamente independiente del lenguaje, pero que utiliza algunos convenios que son similares a los lenguajes de la familia del lenguaje C, como Java, Python, etcétera.
JSON se basa en dos estructuras básicas:

- Una colección de pares nombre-valor, que se pueden entender como objetos de Java o structs de C.

Figura 2.10: Logotipo de JSON

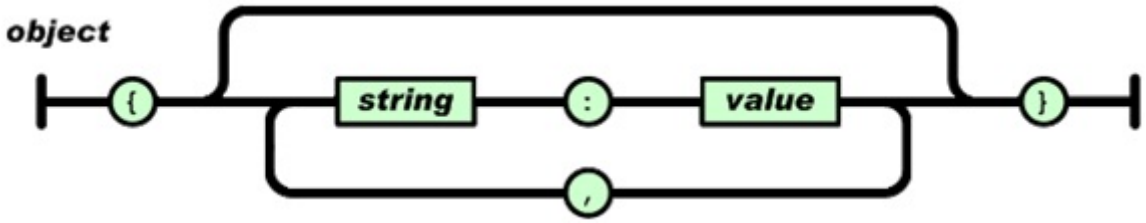

Figura 2.11: Sintaxis de un objeto JSON

- Una lista ordenada de valores, que se pueden entender como un array o una lista.

Dado que estas estructuras son universales y aparecen en algún modo u otro en todos los lenguajes de programación o sistemas, es fácilmente intercambiar datos entre sistemas, independientemente del lenguaje o el entorno en el que estén implementados. La sintaxis de JSON se especifica en las figuras 2.11 y 2.12.

En la aplicación de este PFC, se utiliza JSON para el intercambio de información entre el servidor de partidas, y los dispositivos móviles.

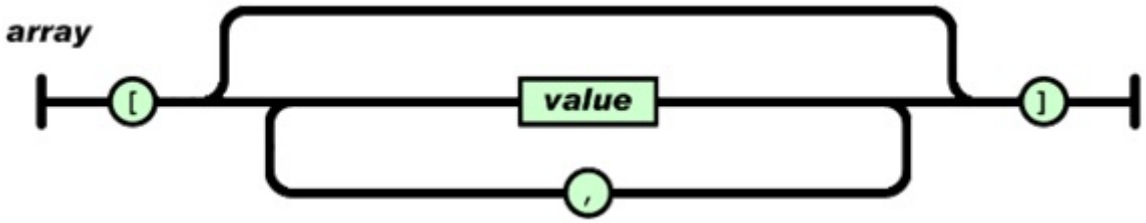

Figura 2.12: Sintaxis de un array JSON

2.11. Rest

REST (Representational State Transfer)[20], es una técnica de arquitectura software para sistemas como la World Wide Web. Este término comenzó a utilizarse en el año 2000, en una tesis doctoral escrita por Roy Fielding, que fue uno de los autores del protocolo HTTP, que es ampliamente utilizado a día de hoy.
Está concebido como un modelo arquitectónico software que explica cómo ha de funcionar la web. Para ello, el modelo dispone de un marco de trabajo guía para el desarrollo de los protocolos estándar de la web, identifica los problemas existentes, evalúa las diferentes alternativas que ofrecen una solución, y pretende asegurar que las extensiones de dichos protocolos no violen las restricciones de la web. Además, recoge todos los aspectos que especifican los requisitos de comportamiento y rendimiento.
Según la técnica, se dispone de una red de recursos que se pueden entender como estados de una máquina por los que el usuario va progresando mediante la selección de enlaces. El resultado de estas transiciones se entiende como el paso a un nuevo estado dentro de la máquina, que se corresponde con un recurso determinado.
Dentro de la arquitectura se tienen recursos, que son representados mediante un identificador y son accedidos mediante una interfaz estandarizada. Cada recurso puede tener asociada una o varias representaciones. Los conectores son todos los nodos de la red que pueden acceder y manipular los recursos, siendo éstos principalmente los clientes y los servidores, aunque hay más. Los componentes son los elementos intermedios que se comunican enviando las representaciones de los recursos, como pueden ser Gateways, Proxys, etcétera.
Según los principios de REST, un recurso es cualquier elemento que tenga una identidad, y dicho identificador no debe exponer detalles acerca de su implementación. Las interfaces por las que se accede a los recursos deben ser uniformes, y se han de utilizar en concordancia a la semántica que se les dio al crearlos. La comunicación entre conectores debe ser una comunicación que no guarde el estado de la misma. Utilizar esta arquitectura favorece la sencillez, la escalabilidad, ya que el estado se guarda en el cliente si es necesario, tiene un mejor rendimiento, ya que favorece la creación de sistemas mediante capas, mejora la capacidad de evolución del software, ya que la interfaz no varía, y se tiene un bajo acoplamiento entre el cliente y el servidor, debido al uso de esta interfaz.
De este modo, todos los recursos están identificados por una URI que sea única para ellos, aunque su representación pueda cambiar. Por ejemplo, para un recurso concreto, podríamos tener estas mismas dos representaciones:

- http://maps.google.com/maps?f=q&hl=es&geocode=&q=fuenlabrada&l=40,-3
- http://maps.google.com/europe/spain/madrid/fuenlabrada

Siendo la segunda de éstas más apropiada para la arquitectura REST, ya que la primera ofrece detalles de implementación. Los métodos HTTP definen operaciones sobre los recursos. Estas operaciones deberían utilizarse para lo que fueron diseñadas:

- POST: Crear un nuevo recurso en la URI que se indica
- PUT: Actualizar el recurso de la URI indicada, y en caso de que no exista, crear uno nuevo en dicha URI.
- DELETE: Eliminar el recurso especificado por la URI.
- GET: Recuperar la representación del recurso especificado en la URI.

En la aplicación del servidor del PFC, se ha tratado de seguir la arquitectura REST para el diseño de las URLS y la representación de los recursos del sistema.

Capítulo 3

Objetivos

Una vez explicados los antecedentes de los juegos que se enmarcan en el ámbito de 'Outdoor Gaming', y las tecnologías que se utilizan para el desarrollo de la aplicación del PFC, vamos a pasar a plantear el problema, a estudiar las alternativas que se barajan para resolverlo, y a presentar la metodología a seguir y la planificación del proyecto.

3.1. Descripción del problema

El presente PFC plantea resolver el siguiente problema:
Se pretende realizar un juego que ejecute en un dispositivo móvil que tiene como sistema operativo el sistema Android. Dicha aplicación, instalada en el dispositivo, utilizará los servicios que ofrece este sistema para la geolocalización del terminal y el acceso a Internet, bien sea por una conexión inalámbrica Wi-Fi, o bien por una conexión a Internet proporcionada por el servidor de telefonía mediante la red 3G. Dadas las limitaciones de alcance de la primera, se recomienda utilizarlo bajo una red 3G, cuya amplitud de alcance es bastante superior.
Dicho juego consistirá en mostrar al usuario una serie de objetivos en un mapa en forma de ubicaciones, a las cuales tendrá que dirigirse un jugador en posesión del teléfono, con lo que conseguirá obtener puntuaciones que jueguen a favor de su equipo.
Por otra parte, se necesita un servidor que tenga capacidad de alojar la base de datos del juego, y sirva peticiones a los diferentes usuarios de los teléfonos mediante el uso del protocolo HTTP vía Internet. Este servidor estará compuesto por dicha base de datos, y una serie de direcciones URL que gestionarán las peticiones de los jugadores, así como servirá un sitio web desde el cual un usuario con permisos de administración podrá gestionar la

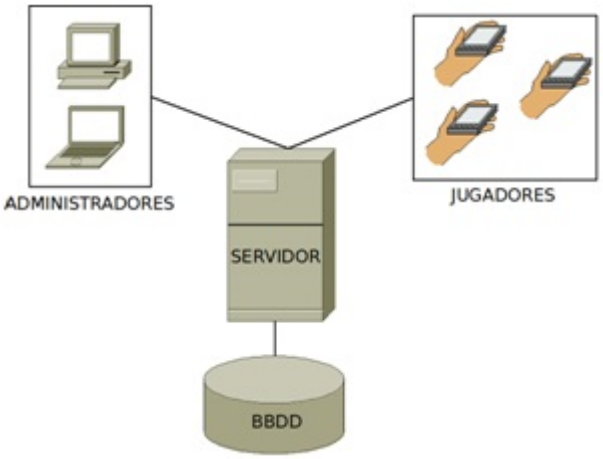

Figura 3.1: Diagrama de componentes del sistema

creación, modificación y eliminación de partidas, equipos, banderas(que son los lugares en los que los jugadores podrán puntuar), y jugadores, así como podrá enviar mensajes a los jugadores, y podrá hacer un seguimiento de las partidas en curso, accediendo al estado de todos los elementos de la base de datos.
La figura 3.1 muestra el diagrama de componentes necesarios para el sistema.

3.2. Estudio de viabilidad

En primer lugar, se ha estado buscando algún tipo de software que previamente se haya desarrollado para resolver este problema, pero en este caso no se conocen antecedentes de un juego como éste.
Si se han encontrado aplicaciones que utilizan el terminal de una manera parecida, como por ejemplo una aplicación que utiliza libregeosocial para hacer gymkhanas[23], desarrollada en la URJC con anterioridad a este proyecto. Por tanto, podemos deducir que la realización de aplicaciones de este tipo es factible, ya que se ha realizado con anterioridad.
A la hora de desarrollar el producto, se consideran varias posibilidades. En primer lugar, el servidor de la aplicación puede desarrollarse con tecnologías diferentes, pero teniendo la posibilidad de utilizar la red social Libregeosocial y su API como base de la que extender este proyecto, se decide construir la presente aplicación de servidor como un módulo dentro de Libregeosocial.
No es así con la aplicación móvil, que también sería posible integrarla como

un módulo dentro de la aplicación móvil de Libregeosocial, pero es preferible desarrollarla independientemente ya que la funcionalidad que se quiere desarrollar no tiene mucho en común con ésta, además de ser más sencillo desarrollarla independientemente.
En cuanto al coste económico de desarrollo del producto, podemos decir que es prácticamente nulo, ya que todo el software y las aplicaciones que se van a utilizar son de código libre, ahorrando por tanto el coste de obtener alguna licencia. El único coste de producción sería el que ha sido necesario para mantener un computador, su conexión a Internet y su conexión a la red eléctrica. Dado que se trata de un PFC, no se tiene en cuenta la compensación económica del programador. Por tanto, desde el punto de vista económico, se trata de un proyecto viable.
Desde el punto de vista técnico, será necesario aprender las tecnologías necesarias para poder desarrollar el software. En este caso, se deberán estudiar con detenimiento las tecnologías necesarias para la aplicación, su funcionamiento, sus API's, y cómo utilizarlas. Esto implicara una fase inicial en el proyecto de estudio y aprendizaje, el cuál consumirá una parte del mismo, pero en cualquier caso, la viabilidad no se ve afectada.
Desde el punto de vista legal, dado que todo el código disponible y el software a utilizar tienen licencias públicas, no se presenta ningún conflicto. Si en el futuro se deseara comercializar la aplicación, habría que estudiar las posibilidades que hay disponibles. En cambio, dado que este proyecto se desarrolla sin ánimo de lucro en un principio, no hay que preocuparse por éste aspecto, y es viable realizarlo.
Desde el punto de vista operativo, teniendo en cuenta que existen aplicaciones anteriores que ya implementan este problema, aunque las funcionalidades se den por separado en aplicaciones completamente diferentes, es decir, que ya existen juegos online que ya implementan este juego, y aplicaciones que ya implementan juegos basándose el geolocalización para móviles Android, lo que se pretende es unificar estos conceptos, junto al de tener una red social basada en libregeosocial, para conseguir desarrollar un sistema que sirva para lo que se propone en la descripción del problema, por tanto, a nivel operativo, la aplicación es viable.
No se especifican requisitos especiales de hardware, salvo disponer de dispositivos con Android con los que realizar las pruebas. Afortunadamente, el entorno de desarrollo Eclipse y la SDK de Android proporcionan un dispositivo Android virtual con el que se podrán realizar las pruebas necesarias, por lo que no es necesario a priori disponer de un terminal físico para comprobar el funcionamiento de la aplicación mientras se desarrolla, aunque sería recomendable disponer de algún terminal de este tipo para hacer pruebas al final del desarrollo.

3.3. Metodología y planificación

El plan del proyecto se recoge en la figura 3.2.

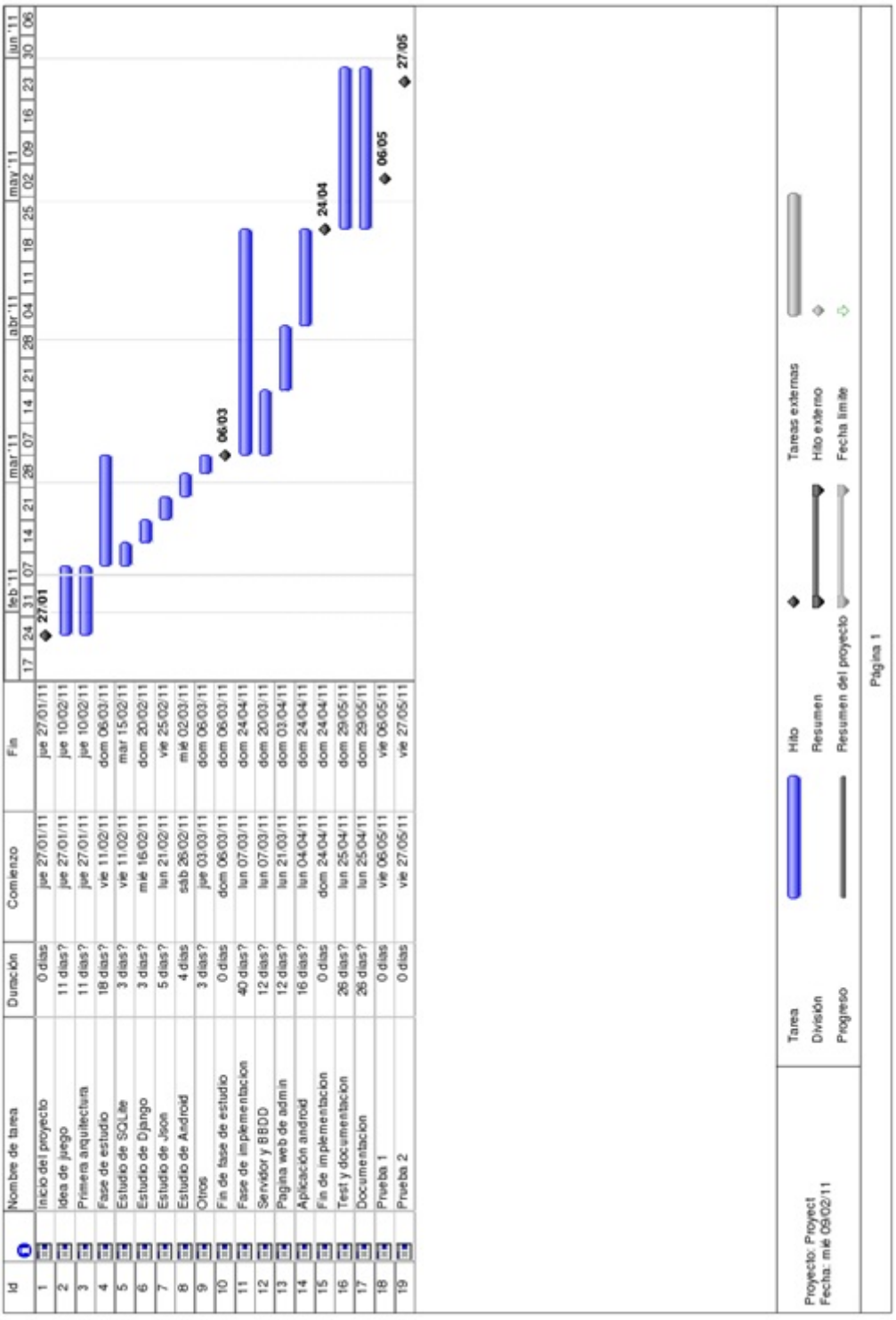

Figura 3.2: Diagrama de Gantt para la planificación del proyecto

Capítulo 4

Descripción informática

En esta parte de la memoria, se explicarán con máximo detalle todas las partes que componen el diseño del sistema que se trata en este PFC, así como se especificarán los detalles implementación y pruebas realizadas.

4.1. Diseño

El diseño del sistema se divide en la realización de varios componentes que colaboran entre sí, con el objetivo de obtener la funcionalidad que hace posible que el producto final sea el juego descrito anteriormente.

4.1.1. Arquitectura del sistema

Se ha decidido utilizar una arquitectura de tres capas, ya que se ajusta perfectamente a las necesidades de este sistema informático. Una forma gráfica de esta arquitectura se puede observar en la figura 4.1.
En la capa de datos, tenemos la base de datos que almacena toda la información relacionada con las estructuras de datos del juego y del sistema. Esta capa la implementa el servidor propio que tiene PostgreSQL.
En la capa de lógica de negocio, tenemos la lógica de la aplicación, es decir, tenemos todo el código que hace que la funcionalidad sea la deseada, y es la que implementa el servidor apache, que utilizando el módulo mod_wsgi le permite comunicarse con el proyecto Django asociado.
En la capa de aplicación tenemos a los clientes, a los cuales hay que diferenciar en dos grupos diferentes. El primer grupo, el de los clientes de la aplicación, serán los dispositivos móviles que estén utilizando la aplicación Android. El segundo grupo, serán los administradores, que utilizan un navegador web para gestionar las partidas del juego.

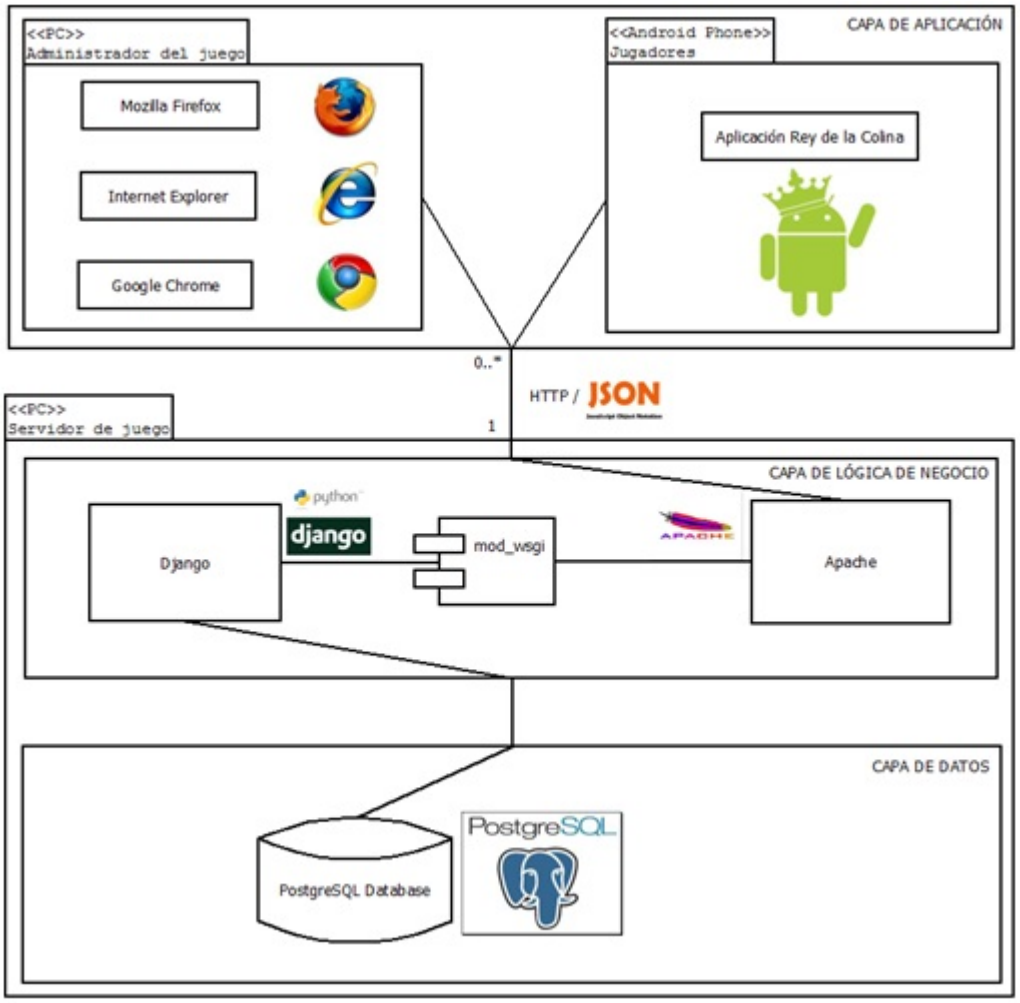

Figura 4.1: Arquitectura software del sistema

En este caso, las capas intermedia e inferior se encuentran ubicadas en una misma máquina, por lo que la comunicación entre estas capas se hace sin tener que realizar ningún protocolo de comunicación. Sin embargo, la comunicación entre el servidor y los clientes se hará mediante HTTP para los administradores, y mediante HTTP y JSON para los jugadores que utilicen la aplicación Android. Por tanto, también puede verse el sistema como una aplicación cliente-servidor.

4.1.2. Diseño del servidor

El diseño del servidor implica el diseño de diferentes módulos que deben colaborar entre sí para garantizar la funcionalidad del sistema. Estos módulos, referidos al servidor, son: el modelo de datos de la aplicación, que será una extensión del modelo de datos de la red social libregeosocial, el diseño del servicio web, de tal manera que esté integrado dentro de libregeosocial como una aplicación más, la forma de diseñar las URLs del sistema, de tal manera que respete las normas que especifica REST, y la aplicación de la arquitectura modelo – vista – controlador, dado que se implementa con Django y la naturaleza de la aplicación es cliente - servidor.

Modelo de datos

El modelo de datos de la aplicación es una extensión de las clases del modelo de datos que tiene libregeosocial. Por tanto, habrá relaciones de herencia entre las clases de la red social móvil y las clases que son necesarias para diseñar el modelo de datos de la aplicación de este PFC. El modelo de datos se puede ver en la figura 4.2 incluye las siguientes clases:

- Partida: contendrá la información relativa a las partidas del sistema. Cada partida incluye el nombre de una partida, la fecha en la que se celebrará, el límite de puntos que tiene que alcanzar un equipo para ganar la partida, y un atributo que indica si la partida está en juego, es decir, si se puede acceder a ella para jugarla o está cerrada. Una partida puede estar cerrada por dos motivos, o bien se están preparando sus banderas y sus equipos para abrirla próximamente, o bien ya ha finalizado. Esta clase no hereda de ninguna clase de libregeosocial.

- Equipo: contendrá la información relativa a los equipos de las partidas. Esta clase tendrá como atributos una referencia a un grupo de libregeosocial, del cual obtendrá el nombre del equipo, una referencia a una partida, que será en la que el equipo participe, un color que será el que

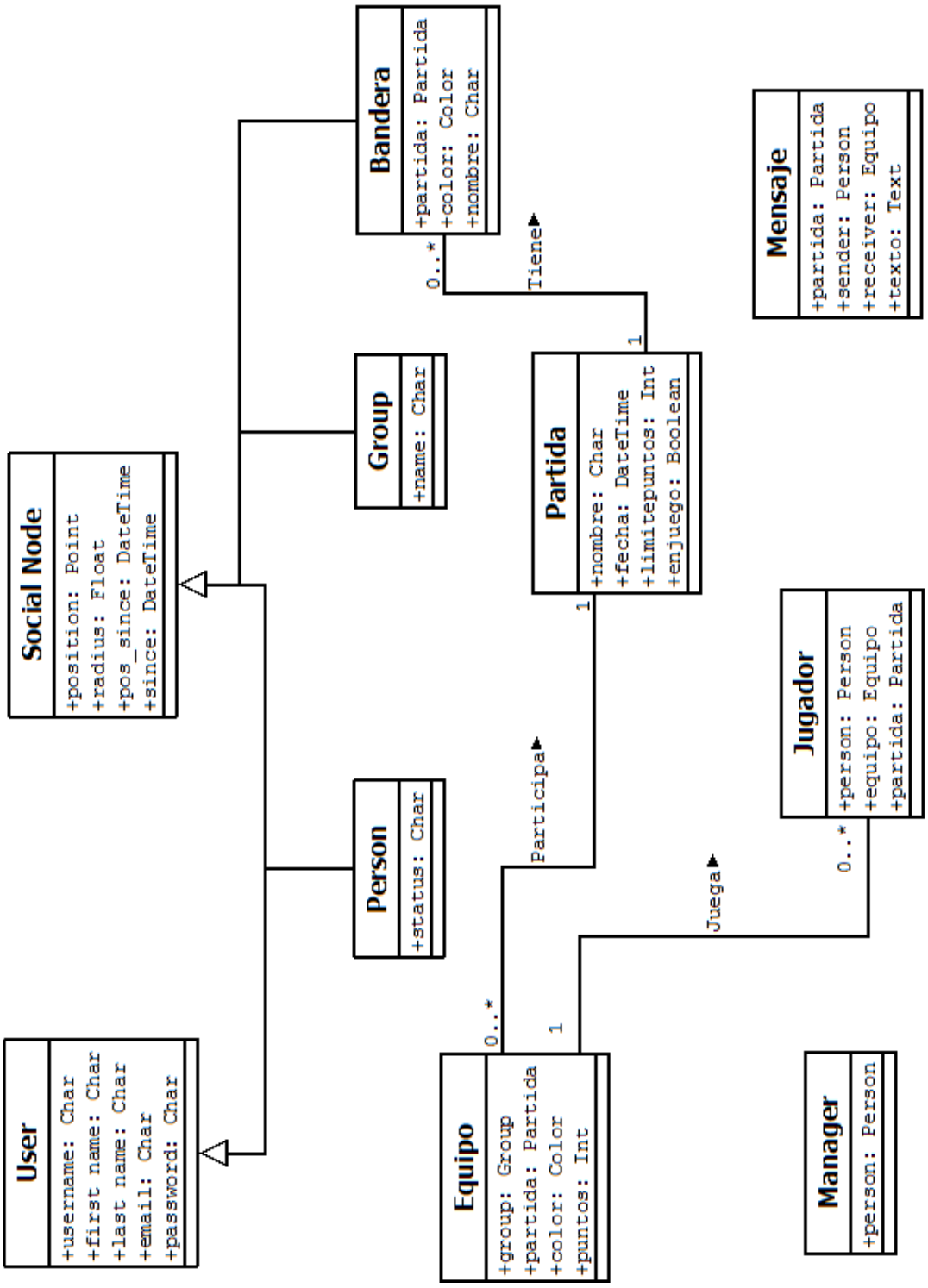

Figura 4.2: Diagrama UML del modelo de datos del sistema

represente al equipo en la partida, y un campo puntos, que servirá para almacenar los puntos que lleva el equipo.

- Jugador: contendrá la información relativa a cada uno de los jugadores que juegan en los equipos del sistema. Tendrá como campos una referencia a una persona de libregeosocial, de donde se sacarán el nombre de usuario, la geolocalización, y el resto de atributos de la clase Persona, tendrá también una referencia a un equipo, y una referencia a una partida.

- Bandera: contendrá la información relativa a los puntos en los que se podrá puntuar dentro de una partida. Como atributos, tendrá una referencia a la partida a la que pertenece, un color que servirá para guardar el estado de color de la bandera, esto es, el color del equipo que puntúa en ese lugar, y un nombre que identifique a la bandera. Esta clase hereda de social_node, por lo que también tendrá atributos para almacenar la geolocalización.

- Mensaje: contendrá la información relativa a los mensajes que se envían dentro del sistema. Tendrá una referencia a la partida a la que pertenece el mensaje, una referencia de la persona que lo envía, y una referencia al equipo al que va dirigido.

- Manager: contiene la información relativa a los usuarios que pueden acceder a la administración de las partidas. Consta de un único atributo, que es una referencia a un objeto de la clase persona de libregeosocial, en la que se accede al nombre de usuario, password, y demás atributos de dicha clase.

Servicio web

La funcionalidad de servicio web se encuentra dentro de libregeosocial como una aplicación dentro de ésta red social. Por tanto, dentro del proyecto Django de libregeosocial se desarrolla la aplicación como un proyecto Django en la carpeta apps.
Según el funcionamiento de libregeosocial, al empezar a servir los sitios web en el servidor, las aplicaciones cuya carpeta comiencen con el símbolo del guión bajo no serán servidas. En este caso, las aplicaciones compareImages y semanticSearch no serán servidas en el servidor, mientras que la aplicación reydelacolina sí será servida. La figura 4.3 muestra la estructura de directorios del servicio web.

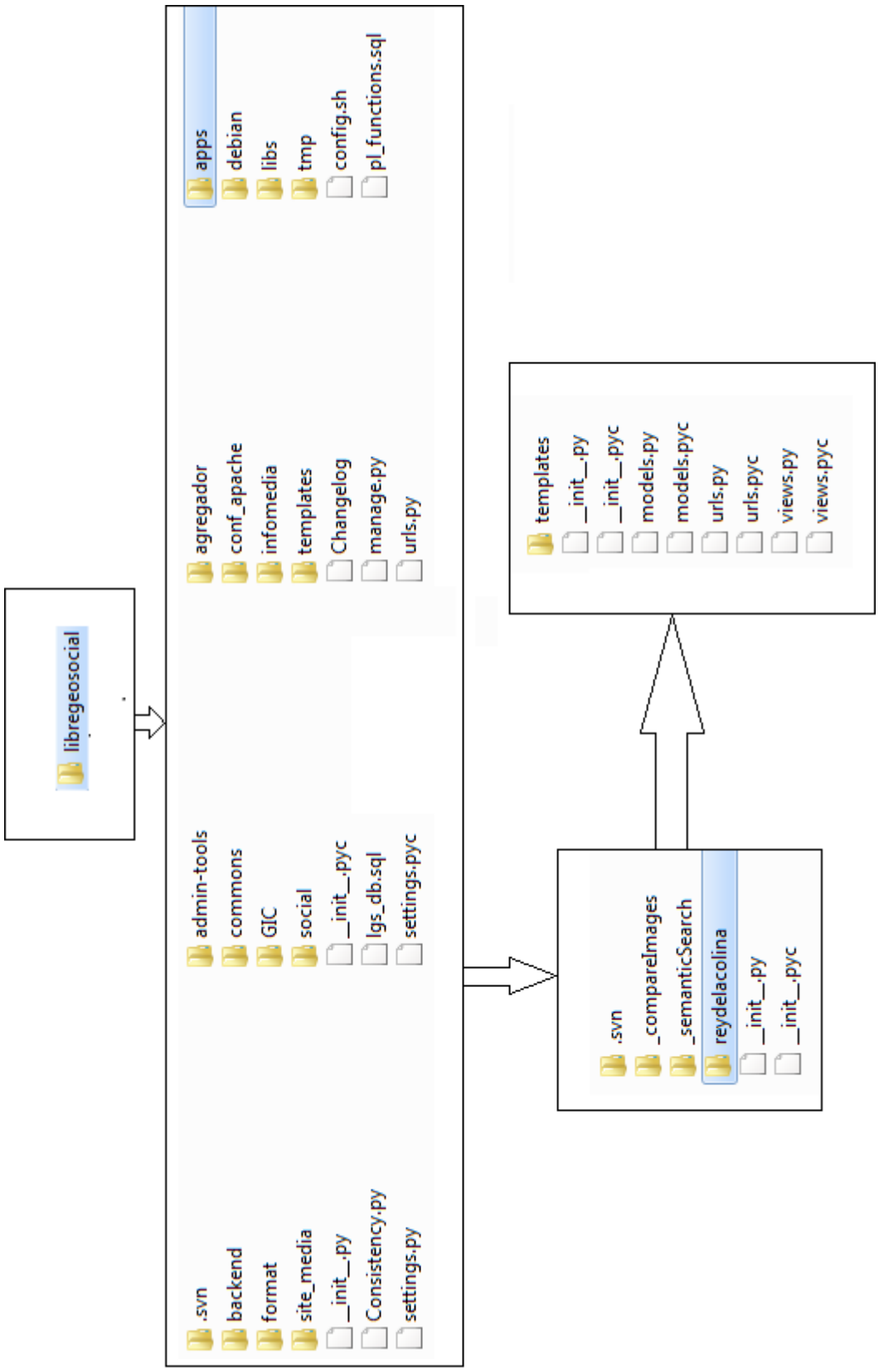

Figura 4.3: Integración dentro de Libregeosocial

Modelo-Vista-Controlador

Aunque los creadores de Django afirman que no utiliza un patrón arquitectónico de modelo – vista – controlador, lo cierto es que sí se sigue este patrón, aunque con un nombre algo diferente. Su equivalente sería un patrón modelo – plantilla – vista.
En este caso, los modelos definen las estructuras de datos y el modelo asociado que el sistema deberá ser capaz de manejar, y que se define en el fichero 'models.py'.
Para el caso de las plantillas, se deberán tener los ficheros HTML o JSON necesarios para el funcionamiento del sistema, así como todo el contenido estático y multimedia que maneje el sistema. Esto incluye imágenes, audio, video, etcétera. Será la manera de representar y mostrar los datos que el sistema maneja, y en concreto en éste PFC se recoge en la carpeta 'templates'.
El controlador es el código que ejecuta el servidor para procesar las peticiones, modificar las estructuras de datos, acceder a la base de datos, crear nuevas estructuras de datos, eliminar estructuras de datos, y responder a peticiones con ficheros HTML o JSON. Este código, escrito en Python, se recoge en el fichero 'views.py'.

URLs y Rest

La manera que tiene Django de asociar el controlador con las diferentes URLs del sistema es mediante un fichero 'urls.py', en el cual se especifican los recursos que pueden ser accedidos del sistema, y el controlador que se encarga de procesar las peticiones de estos recursos.
Dado que el sistema sigue la filosofía REST, las direcciones no muestran detalles de implementación. De esta manera, las URLs del sistema siguen la siguiente estructura:

/reydelacolina/
/reydelacolina/partidas/
/reydelacolina/partidas/crear/
/reydelacolina/partidas/1/
/reydelacolina/partidas/1/borrar/
/reydelacolina/partidas/1/equipos/
/reydelacolina/partidas/1/equipos/crear/
/reydelacolina/partidas/1/equipos/1/borrar/
/reydelacolina/partidas/1/equipos/1/jugadores/
/reydelacolina/partidas/1/equipos/1/jugadores/crear/
/reydelacolina/partidas/1/equipos/1/jugadores/1/

/reydelacolina/partidas/1/equipos/1/jugadores/1/borrar/
/reydelacolina/partidas/1/banderas/
/reydelacolina/partidas/1/banderas/crear/
/reydelacolina/partidas/1/mensajes/

Como se puede observar, no se muestran detalles de implementación en las URLs, lo cual hace que la filosofía REST se cumpla. Del mismo modo, la estructura que tienen las URLs es coherente con el modelo de datos que representa, lo que hace que sea una representación muy adecuada para el servicio web que se pretende servir.

4.1.3. Diseño de los clientes

Dado que se tienen dos tipos de clientes, por un lado los que acceden a través de un navegador para gestionar las partidas, y por otro lado los que juegan a los juegos a través de un terminal Android, debemos especificar las funcionalidades de cada uno de los clientes por separado.

Interfaz web

Los clientes que accedan desde un navegador web, deberán acceder al sistema mediante un usuario y una contraseña, y el sistema les permitirá gestionar toda la información relativa a las partidas, equipos, banderas, jugadores, y mensajes. Del mismo modo, se podrán dar de alta, modificar y eliminar todos estos elementos en cualquier momento, así como la creación, eliminación y modificación de nuevos administradores.
Para simplificar el diagrama de la figura 4.4, existe la posibilidad de salir de la aplicación desde cualquier estado del diagrama, lo cual conduce al estado de 'Pantalla de login'. Estas transiciones no se han dibujado para no complicar más la figura.

Clientes Android

Los clientes que acceden al sistema desde la aplicación para Android, podrán acceder a la lista de partidas del sistema, y en aquellas que estén abiertas para poder jugar, podrán seleccionar un equipo y establecer su nombre de usuario. A partir de este momento, se considera que están jugando la partida y habrá una serie de funcionalidades que se comunican con el servidor de manera automática y transparente al usuario, como por ejemplo el envío periódico de la posición del jugador, la recepción del estado de las banderas del juego, la recepción de mensajes nuevos, etcétera.

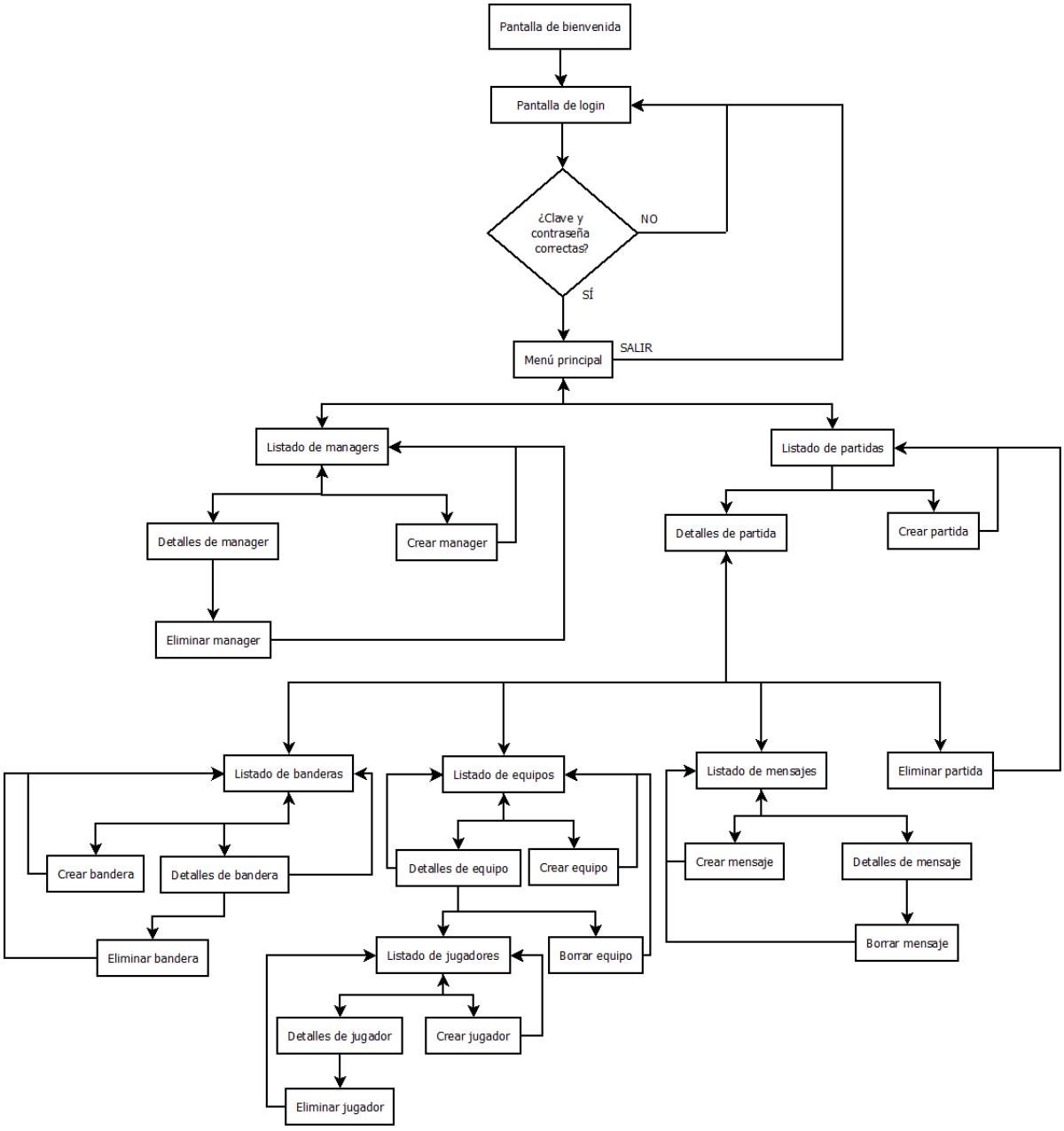

Figura 4.4: Diagrama de flujo del cliente web

En cuanto a la aplicación, internamente hay actividades que no son mostradas al usuario, y sirven para realizar tareas periódicamente. Estas tareas son las siguientes:

- Controlar la localización del usuario: Se encarga periódicamente de obtener las coordenadas de localización que proporciona el GPS del terminal, y enviar esta información al servidor, para que sea actualizada en la base de datos.
- Controlar las banderas: Se encarga periódicamente de obtener el estado de las banderas de la partida, de modo que se pueda mostrar al usuario la ubicación de las banderas de la partida, y su estado de color.
- Controlar las distancias: Se encarga periódicamente de calcular la distancia del jugador a cada una de las banderas de la partida, de tal manera que si se encuentra cerca de una de ellas, manda una solicitud al servidor para cambiar el color de la bandera.
- Controlar los jugadores: Se encarga periódicamente de obtener los jugadores de la partida y sus posiciones, para actualizar el mapa que muestra la ubicación de los jugadores.
- Controlar los mensajes: Se encarga periódicamente de recibir los mensajes del servidor, para mostrarlos en el terminal del usuario y notificar si se ha recibido algún mensaje nuevo.
- Controlador de puntos: Se encarga de calcular los puntos que se han sumado desde la última actualización de puntos recibida, y se envía al servidor una petición para que sume los puntos que faltan.
- La actividad de 'profile', además, se encarga periódicamente de obtener las puntuaciones globales de todos los equipos y mostrarlas en la pantalla del terminal.

La figura 4.5 muestra un diagrama de flujo con la funcionalidad descrita.

4.2. Implementación

En esta parte de la memoria del PFC se tratará cada una de las partes del mismo con el máximo detalle a nivel de implementación. Para ello se explicará cada una de las funcionalidades y partes del sistema con detalle, y se incluirán capturas de pantalla a modo de ejemplo. Hay que mencionar

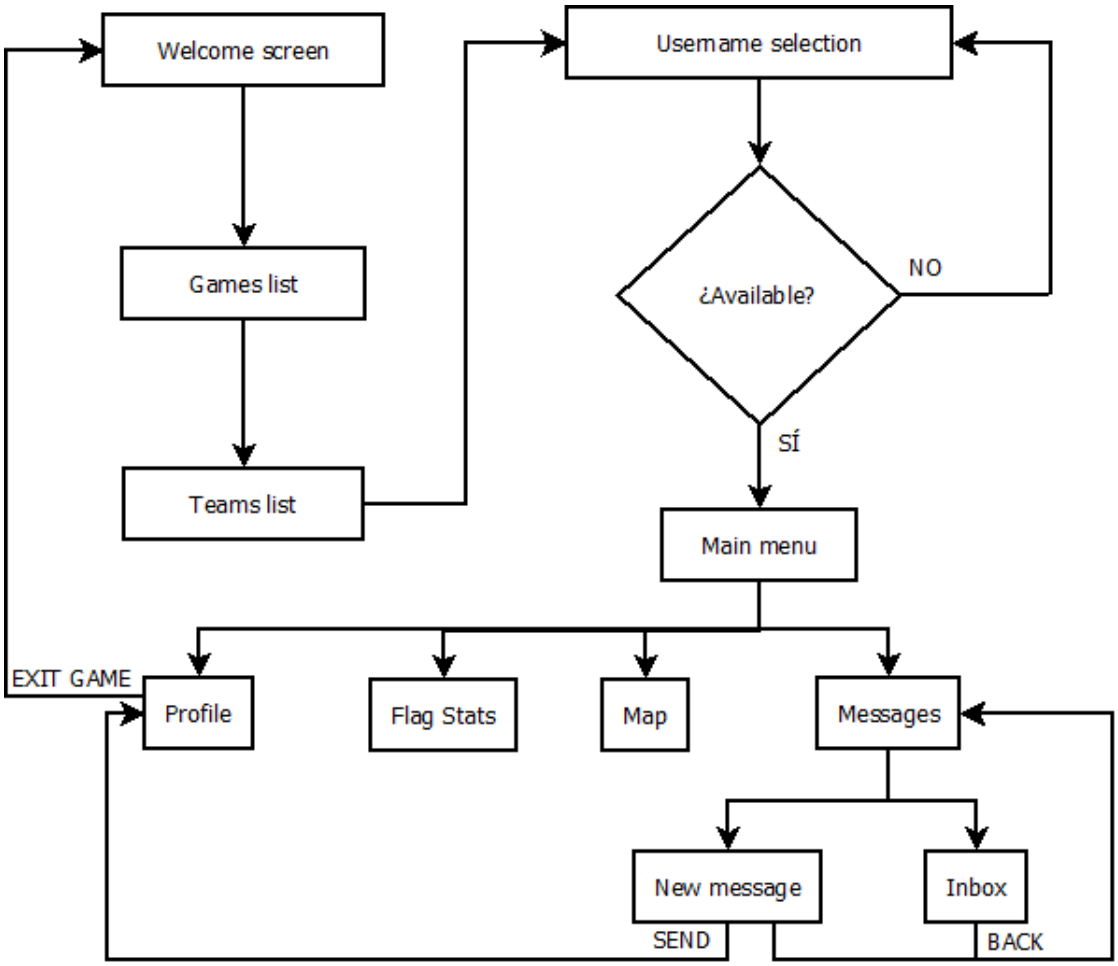

Figura 4.5: Diagrama de flujo de la aplicación Android

que la implementación de la parte servidora se ha realizado con el editor de textos por defecto que incluye el sistema operativo Ubuntu [22], mientras que la aplicación móvil se ha desarrollado en el entorno de desarrollo Eclipse[21].

4.2.1. Implementación del servicio web

La implementación del servicio web es necesaria para que todos los usuarios que accedan al sistema de forma remota como administradores puedan interactuar con él y conseguir realizar sus propósitos. En este apartado se especificará exactamente cómo funciona cada módulo.

Acceso al sistema

En el navegador se realiza la petición GET a la dirección '/reydelacolina/login/', por lo que el servidor devuelve el fichero 'login.html', que muestra la pantalla de acceso al sistema de administración. Una vez que el usuario ha introducido el nombre de usuario y la clave de acceso, se realiza una petición POST al sistema, el cual se encarga de acceder a la base de datos para comprobar que existe un usuario con ese nombre de usuario y que la contraseña introducida coincide.
En caso de que así sea, se mantiene un estado en el navegador en forma de cookie para que el resto de accesos a las diferentes URLs del sistema se hagan mediante esta autenticación, y se redirecciona al usuario a la página 'panel.html', que es la que muestra un menú en el que se recogen enlaces a las páginas que muestran los listados de las partidas y de los managers.
Por el contrario, en el caso de que la autenticación sea errónea, se redirecciona a una página de error en la que se indica que el password o la contraseña son incorrectos. No se detalla más información en este mensaje de error para no dar información al usuario. Si se especificara que la contraseña es incorrecta, se daría a conocer que el usuario existe, pero que esa no es su contraseña, lo cual no es bueno para la seguridad del sistema de administración.
La figura 4.6 muestra la pantalla de entrada a la aplicación, la figura 4.7 muestra la pantalla a la que se redirecciona a un usuario si el acceso es correcto, y la figura 4.8 muestra la pantalla de error a la que se redirecciona al usuario en caso de que el acceso sea incorrecto.

Gestión de managers

Cuando un usuario que ha accedido correctamente a la aplicación es redireccionado al panel principal, puede gestionar los managers que hay en el sistema. Esto implica la creación, listado y eliminación de managers.

Figura 4.6: Pantalla de login

Figura 4.7: Panel de administración

Figura 4.8: Acceso incorrecto de manager

Al acceder a los managers, se le muestra una lista con los managers que hay en el sistema, pudiendo crear un nuevo mánager siempre que pulse sobre la imagen con el icono que se especifica en el margen. En caso de acceder a este enlace, se le redirige a la página con el formulario de creación de managers. El formulario de creación de managers se obtiene mediante una petición GET a la dirección '/reydelacolina/managers/registrar/'. La página que se muestra como resultado de esta petición al servidor es la que se observa en la figura 4.9.

En este caso, se deben rellenar los campos con los datos del nuevo manager. Al pulsar en el botón Registrar, se hace una petición POST al servidor con los datos del formulario. El servidor comprueba que estos datos no están vacíos, en cuyo caso devuelve una página HTML de error indicando que hay algún campo vacío, como se ve en la figura 4.10 Si los parámetros no están vacíos, el servidor comprueba que no existe ya un mánager con el nombre de usuario que se especifica en el campo 'Usuario'. Si el usuario ya existiera, devolvería un fichero HTML indicando esta situación, por lo que habría que elegir otro nombre de usuario. Esto se puede ver en la figura 4.11. En cambio, si el nombre de usuario es nuevo, el servidor crea una nueva instancia de la clase Manager, y le asigna los datos que se han especificado en el formulario. Se redirecciona al usuario a la URL '/reydelacolina/managers/', que muestra una lista con los managers del sistema, como se observa en la figura 4.12. El usuario puede hacer clic en los nombres de los managers para obtener los de-

Figura 4.9: Registro de manager

Figura 4.10: Error por falta de parámetros

Figura 4.11: Error por existencia de usuario

Figura 4.12: Listado de managers

Figura 4.13: Detalles de manager

talles de un mánager, o eliminarlo del sistema. Cada usuario mostrado es un enlace a la URL que representa a dicho manager. En este caso, la dirección URL para el manager 'default2' es '/reydelacolina/managers/2/', ya que en el sistema este manager tiene el identificador 2. Cuando se hace un GET a esta URL, el servidor devuelve un fichero HTML con los detalles relativos al manager con este identificador, como se ve en la figura 4.13. En este caso se muestran los atributos relativos a éste manager. Para garantizar la seguridad, se evita mostrar la contraseña de acceso tal y como es, y se muestra la codificación que se guarda en la base de datos de la contraseña. Lo más seguro sería no mostrar la contraseña, pero dado que a este servicio web sólo pueden acceder administradores que se han autenticado, no se considera un problema de seguridad.
Además, se observa un enlace a la URL /reydelacolina/managers/2/borrar/

Figura 4.14: Listado de partidas

en forma de imagen como la que se observa al margen. Según REST, se debería realizar una petición DELETE sobre la URL, pero en este caso la aplicación lo que hace es GET, dado que el servidor sólo está implementado para atender peticiones GET y POST.
Si se hace la petición GET a la URL para borrar al manager, el servidor accede a la base de datos y elimina la instancia de la clase Manager que tiene el identificador de manager de la URL. Después, redirecciona al usuario al listado de managers, para que se observe que el manager ha sido eliminado del sistema, así como sus recursos asociados.
Para todos los recursos del sistema (partidas, equipos, mensajes, banderas y jugadores), la funcionalidad de los enlaces a crear y eliminar funcionan de manera similar, si bien más adelante se especificarán los campos de cada tipo de datos, no se especificarán los métodos HTML que se utilizan, por ser análogos, salvo por diferencias que sí que haya que destacar.

Gestión de partidas

El administrador de una partida podrá acceder a las partidas del sistema para consultar el estado de los parámetros de la misma, así como crearlas o eliminarlas a su gusto. Dentro del apartado de partidas del panel, podrá visualizar al listado de partidas del sistema. Esto se recoge bajo la URL '/reydelacolina/partidas/'. Un ejemplo de un listado de partidas es el que se observa el la figura 4.14.
De este modo, seleccionando una partida, se accede a los detalles de la misma. La construcción de los enlaces se hace en base al identificador de la partida en la base de datos del sistema. Por ejemplo, el acceso a 'Partida1' será una petición GET del recurso '/reydelacolina/partidas/1/', ya que su

Figura 4.15: Detalles de partida

identificador en el sistema es el 1.
Los detalles de cada partida se muestran tras hacer clic en el nombre de la misma. Un ejemplo de detalles para una partida, sería el que se ve en la figura 4.15. Como se puede observar, se muestra el nombre de la partida que aparecía en el enlace anterior, y se detallan sus parámetros, como son la fecha estimada de juego, el límite de puntos establecido para vencer, el estado de la partida, esto es, si se encuentra activa o cerrada, así como se da la posibilidad de abrir la partida mediante un botón. Para este ejemplo concreto, éste botón realiza un POST sobre la dirección '/reydelacolina/partidas/4/abrir/', de tal manera que el servidor accede a la base de datos y cambia el atributo de la partida.
Además, se permite acceder a la gestión de equipos, banderas y mensajes de la partida, cuyas URLs correspondientes son, para este ejemplo concreto:

- '/reydelacolina/partidas/4/equipos/'
- '/reydelacolina/partidas/4/banderas/'
- '/reydelacolina/partidas/4/mensajes/'

Si se hace clic en la imagen con el símbolo '-', la partida se elimina del sistema, con un funcionamiento análogo a la eliminación de managers del sistema.

Figura 4.16: Creación de partidas

Volviendo al listado de partidas, si el manager hace clic en la imagen con el símbolo '+', se realiza una petición GET a la dirección '/reydelacolina/partidas/crear/', y el servidor devuelve una página HTML que contiene un formulario que permite la creación de una partida nueva en el sistema. Éste formulario es como el que se ve en la figura 4.16. Aquí, el administrador debe especificar un nombre para la partida, una fecha en formato [AAAA-MM-DD HH:MM], siendo A el valor numérico del año, M el valor numérico del mes, D el valor numérico del día, H el valor numérico de la hora, y M el valor numérico del minuto en el que se tiene previsto que la partida dé comienzo.
En caso de que falte algún parámetro en el formulario, el servidor devolverá la página que se observa en la figura 4.17 con un mensaje de error que indica que no se ha podido crear la partida debido a que falta algún parámetro. Si, por el contrario, los parámetros fueran introducidos pero no fueran correctos, es decir, que el formato de la fecha no es válido, o el límite de puntos no sea un número entero positivo mayor de cero, el servidor no podrá crear la partida, y mostrará el error en una página como la que se ve en la figura 4.18.
Hay que destacar la diferencia en el uso de métodos GET y POST para la misma URL, en este caso '/reydelacolina/partidas/crear/'. Si el método es GET, el servidor devuelve una página con el formulario que hay que rellenar para crear una nueva partida. En cambio, si el método es POST, el servidor intentará recoger los datos introducidos en el formulario, y que deberían estar incluidos en el mensaje HTTP que le ha llegado. Con estos datos, creará la

Figura 4.17: Error por falta de parámetros

Figura 4.18: Error por parámetros incorrectos

partida y, en función del resultado, devolverá una página de error en caso de no haber podido crear la partida, o redireccionará al usuario al listado de partidas si ha podido realizarlo con éxito.

Gestión de equipos

Dentro de cada partida, el administrador puede manejar los equipos que participan en ella. Para ello, dentro de una partida, puede hacer clic en el icono de 'Equipos', de tal manera que accederá a listado de equipos para esa partida. Se realiza una petición HTTP con el método GET a la dirección '/reydelacolina/partidas/2/equipos', y el servidor consulta los equipos que participan en la partida con identificador, en este caso 2, y devuelve una página HTML con el listado de los equipos, y los enlaces a cada uno de dichos equipos, con su identificador de equipo, como se ve en la figura 4.19.

Como se puede observar en el ejemplo, se muestran también los puntos que lleva cada equipo, ya que esta es una forma de monitorizar las partidas mientras se están jugando. Del mismo modo que sucedía con la gestión de partidas, se permite al mánager crear un nuevo equipo para la partida, accediendo al enlace con el símbolo '-'. Se realiza al servidor una petición con el método GET a la URL correspondiente, y se muestra una página web con un formulario similar al que se ve en la figura 4.20.

En este caso, se debe especificar un nombre de equipo, y el color que ser-

Figura 4.19: Listado de equipos

Figura 4.20: Creación de equipos

Figura 4.21: Error por falta de parámetros

Figura 4.22: Error por existencia de equipo

virá para identificar a sus jugadores en la partida. En el caso de que no se especifique un nombre de equipo, el servidor devolverá una página de error indicando la razón por la que no se ha podido crear el equipo como la que se ve en la figura 4.21.

Si por el contrario el nombre de equipo ya estuviera registrado en el sistema, se notificará al administrador que ese grupo ya existe, con un mensaje similar al que muestra la figura 4.22.

Del mismo modo que ocurría con la creación de partidas, es diferente el comportamiento del servidor para los métodos GET y POST a la hora de la misma URL para la creación de equipos. En cuanto a los detalles de un equipo, se puede acceder a ellos desde la lista de equipos de la partida. Para ello, se hace una petición al servidor mediante el método GET a la URL correspondiente, la cual incluye el identificador de la partida, y el identificador del equipo dentro del sistema.

A la hora de ver los detalles de un equipo, existe la posibilidad de eliminarlo, del mismo modo que se podían eliminar managers y partidas. El funcionamiento es similar, solo hay que acceder al enlace marcado con el símbolo '-'. Un ejemplo de los detalles de un equipo sería el que se muestra en la figura 4.23.

Se puede observar una imagen que sirve de enlace a la gestión de los juga-

Figura 4.23: Detalles de equipo

dores del equipo, la cual se explica más adelante.

Gestión de banderas

Las banderas son uno de los pilares fundamentales del juego, ya que representan los puntos geográficos en los que los jugadores pueden puntuar, una vez alcancen la posición. Por tanto, para que una partida tenga sentido, los managers deberán crear banderas dentro de una partida para que los jugadores las puedan capturar, del mismo modo que se debieron crear equipos a los que los jugadores se unirán al jugar. Al acceder al enlace de banderas de una partida, se muestra un listado con las banderas de la partida como el de la figura 4.24.

Cada bandera tendrá asociado un identificador de bandera que el servidor devolverá en los enlaces a cada bandera cuando sirva el fichero HTML con el listado de banderas de la partida que corresponda al identificador de partida de la URL. El administrador podrá comprobar el estado de cada bandera, esto es, el color del equipo al que pertenece, y sus coordenadas de latitud y longitud para el posicionamiento GPS. Para ello solo tiene que seguir el enlace, y el servidor le devolverá un fichero HTML con los detalles de esa bandera, como el ejemplo de la figura 4.25.

Se puede monitorizar el estado de cada bandera si se accede a los detalles de la misma en un momento determinado. Si desea eliminar la bandera de la partida, el manager deberá seguir el enlace con el símbolo '-', al igual que

Figura 4.24: Listado de banderas

Figura 4.25: Detalles de bandera

Figura 4.26: Creación de banderas

en ocasiones anteriores. El proceso de creación de banderas es algo peculiar, ya que para ello se necesita acceder a la dirección http://maps.google.com, y poder obtener de forma precisa las coordenadas del punto en el que se desee crear una bandera. En el futuro, se podría mejorar la página de creación de banderas de tal manera que muestre un mapa y se pueda hacer clic en la posición en la que se desea poner una bandera, actualizando los valores de latitud y longitud en las cajas de texto donde se escriben las coordenadas. En este momento, el formulario de creación de banderas es como se muestra en la figura 4.26.

Del mismo modo que en otros formularios de creación, en caso de que falte algún parámetro por rellenar, el servidor devolverá un fichero HTML informando del error. En el caso de que los campos estén correctos y se haya podido crear la bandera, se redireccionará al listado de banderas de la partida.

La manera de introducir las coordenadas de latitud y longitud para una nueva bandera es muy sencilla si se accede al servicio de mapas de Google. Por ejemplo, si se desea crear una bandera que corresponda a la biblioteca de la universidad de Fuenlabrada, los pasos a seguir serían los siguientes:

Primero, habrá que buscar un lugar aproximado al lugar donde se esté desarrolle la partida, como muestra la figura 4.27.

Después, centrar el mapa en el lugar en el que se desee poner la bandera, como muestra la figura 4.28.

Figura 4.27: Búsqueda de un lugar

Figura 4.28: Mapa centrado en la localización

A continuación, centrar el lugar exacto en el lugar en el que se desea la bandera, y hacer clic con el botón derecho del ratón, para seleccionar la opción '¿Qué hay aquí?'. La figura 4.29 muestra un ejemplo.

Por último, aparecerán en la barra superior las coordenadas del punto, separadas por una coma. En este caso, la primera coordenada corresponde con la longitud, y la segunda corresponde con la latitud, como se ve en la figura 4.30.

Tan sólo falta copiar estos valores en las cajas de texto de la página de creación de banderas, para que la bandera tenga las coordenadas correctas, como se muestra en la figura 4.31.

Gestión de jugadores

Por último, los managers podrán ver los jugadores que están participando en una partida en tiempo real, accediendo al equipo de una partida y accediendo al enlace de jugadores. Si bien es cierto que un manager puede crear y

Figura 4.29: Selección de un punto para obtener coordenadas

Figura 4.30: Coordenadas de un punto

eliminar jugadores, en este sistema no tiene mucho sentido hacer esto, ya que los jugadores accederán al sistema mediante la aplicación, momento en el que se creará un nuevo usuario con el nombre que elijan, y éste será eliminado cuando salgan de ella.
Sí tiene sentido poder comprobar que jugadores están participando en una partida, ya que en un futuro puede ser interesante la inclusión de estadísticas personalizadas para cada jugador, o la posibilidad de permitir al usuario registrar un nombre de usuario bajo una contraseña.
En cualquier caso, el manager puede comprobar los jugadores que participan en un equipo, así como ver los detalles de un jugador en concreto. La figura 4.32 muestra un ejemplo del listado de jugadores que están jugando en un equipo. Y la figura 4.33 muestra en detalle los datos de un jugador.

Gestión de mensajes

Otra de las funcionalidades que le permite hacer el servicio web a un manager es la posibilidad de enviar mensajes a los jugadores que están participando en una partida. Del mismo modo, un manager puede ver los mensajes que se han enviado en la partida, y eliminar los que considere que no deben permanecer en el sistema.

Figura 4.31: Creación de banderas

Figura 4.32: Listado de jugadores

Figura 4.33: Detalles de jugador

Figura 4.34: Listado de mensajes

Al igual que se mostraban listados de partidas, equipos, banderas, etcétera, se muestra un listado de los mensajes de la partida. Cada elemento de este listado es un enlace a los detalles de un mensaje, como se observa en la figura 4.34.

Si se sigue el enlace de un mensaje concreto, se puede ver el texto del mensaje, el emisor y el receptor, como muestra la figura 4.35.

El manager podrá eliminar los mensajes que considere mediante el enlace con el símbolo '-', que funciona de forma similar a los vistos anteriormente para otros elementos del sistema.

En el listado de mensajes, se permite la creación de nuevos mensajes, cuya interfaz es similar a la que se muestra en la figura 4.36.

Si el campo de texto del mensaje es vacío, se mostrará una página de error indicando que faltan parámetros como la que se observa en la figura 4.37.

Figura 4.35: Detalles de mensaje

Figura 4.36: Creación de mensajes

Figura 4.37: Error por falta de parámetros

En caso de enviar correctamente el mensaje, se volverá al listado de mensajes, donde aparecerá una nueva entrada en el listado con el mensaje que se acaba de generar.
El funcionamiento del sistema de mensajes es bastante sencillo. Los jugadores o managers accederán a esta URL con los atributos necesarios para la creación de un nuevo mensaje. El servidor se encargará de crearlo y almacenarlo en la base de datos.
Eventualmente, un manager solicitará la lista de mensajes, de tal forma que los podrá leer, así como la aplicación móvil solicitará la lista de mensajes para mostrarla en el terminal de manera periódica. En el caso de los terminales móviles, la lista se recibirá en formato JSON, mientras que las peticiones de un manager desde un navegador web se resolverán enviando un fichero HTML.

4.2.2. Implementación de la aplicación móvil

La aplicación móvil es la parte cliente principal del sistema, junto al servicio web de administración para managers. Esta aplicación será la que ejecute en los teléfonos móviles, de tal manera que los jugadores estarán accediendo al servidor mediante una conexión a Internet. Estas conexiones serán completamente transparentes para los jugadores, porque la percepción que ellos tendrán en el teléfono es simplemente la visualización de cambios de estado en los elementos del juego, sin percatarse de que dichos cambios de estado estarán realizándose debido a la comunicación con el servidor a través de Internet.
La funcionalidad de la aplicación se explicará pasando por todas las pantallas que se muestran durante una partida.

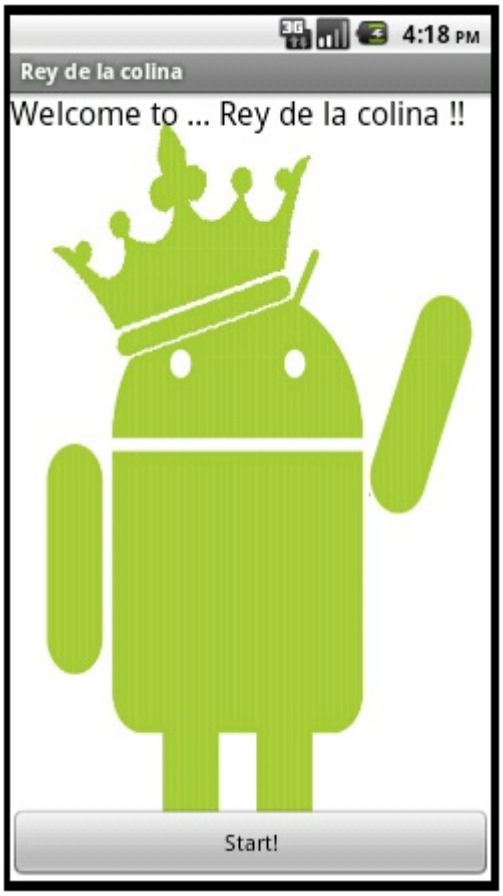

Figura 4.38: Pantalla de bienvenida

Pantalla de bienvenida

Tras acceder a la aplicación en el teléfono, la primera pantalla que un usuario percibe es la que se observa en la imagen. En esta pantalla lo que se muestra es un mensaje de bienvenida a la aplicación, y se observa un botón cuya funcionalidad es la de iniciar el juego. La funcionalidad relativa a esta parte del juego es irrelevante, ya que simplemente consiste en mostrar una imagen de fondo, un texto de bienvenida, y un botón. La figura 4.38 muestra esta pantalla.

Selección de partida

El siguiente paso es seleccionar la partida en la que se desea jugar. De forma transparente al usuario, la aplicación envía una petición GET al servidor

Figura 4.39: Selección de partida

Figura 4.40: Notificación de partida cerrada

a la URL de partidas, del mismo modo que lo hacía en el cliente web para obtener el listado de partidas. En este caso, añade una cabecera 'format=json' en la URL para indicar al servidor de que la petición ha sido realizada por un cliente móvil, por tanto, el servidor no devuelve el fichero HTML con las partidas, sino que devuelve un fichero JSON con dichas partidas. La aplicación procesa este mensaje y genera una lista de partidas que es la que finalmente muestra al usuario.

En el caso de que el servidor no esté funcionando, o que no haya partidas en el servidor, se mostrará un mensaje indicando que no hay partidas disponibles. De lo contrario, se mostrará una lista con el nombre de las partidas similar al que se aprecia en la imagen 4.39.

El jugador seleccionará una partida de la lista, la cual puede estar abierta o cerrada. En caso de estar cerrada, se le notificará mediante un mensaje como el que muestra la figura 4.40. Por el contrario, si la partida está accesible, el jugador pasará a la siguiente pantalla, que es la de selección de equipo.

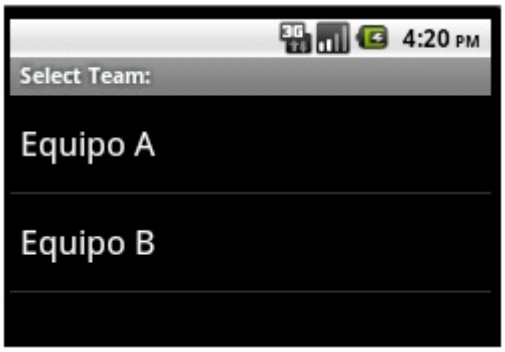

Figura 4.41: Selección de equipo

Selección de equipo

Tras elegir la partida a la que unirse, la aplicación móvil realizará una nueva petición al servidor, en este caso correspondiente a la lista de equipos para la partida seleccionada. Al igual que para las partidas, se realizará una petición GET a la URL de los equipos de la partida, y se añadirá la cabecera indicando que la petición es para recibir los equipos en formato JSON. Tras recibir la lista y procesarla, genera y muestra una lista como la que se muestra en la imagen 4.41.

Si no se hubiera podido realizar la conexión con el servidor debido a algún error de red, o debido a que no hay equipos creados para esa partida en el servidor, se notificará al usuario que no hay ningún equipo disponible, por lo que deberá pulsar el botón de atrás del teléfono y seleccionar otra partida diferente, o reintentar la conexión a la misma partida.

Tras recibir una lista de equipos, el jugador elegirá al equipo al que quiere unirse para la partida. Hecho esto, la aplicación móvil pasará a la siguiente pantalla, que es la de selección de un nombre de usuario.

Selección del nombre de usuario

Llegados a este punto, se presenta una pantalla con una caja de texto, en la cual el jugador deberá escribir el nombre de usuario que desee. Un ejemplo es el que se ve en la figura 4.42.

Tras escribir el nombre de usuario, pulsará el botón para comenzar la partida. La aplicación móvil se comunicará con el servidor para comprobar la existencia del nombre de usuario. Si existe, se mostrará un mensaje indicando que el usuario ya existe, de tal manera que el jugador tendrá que elegir otro nombre de usuario diferente. Si el nombre de usuario no existe en el

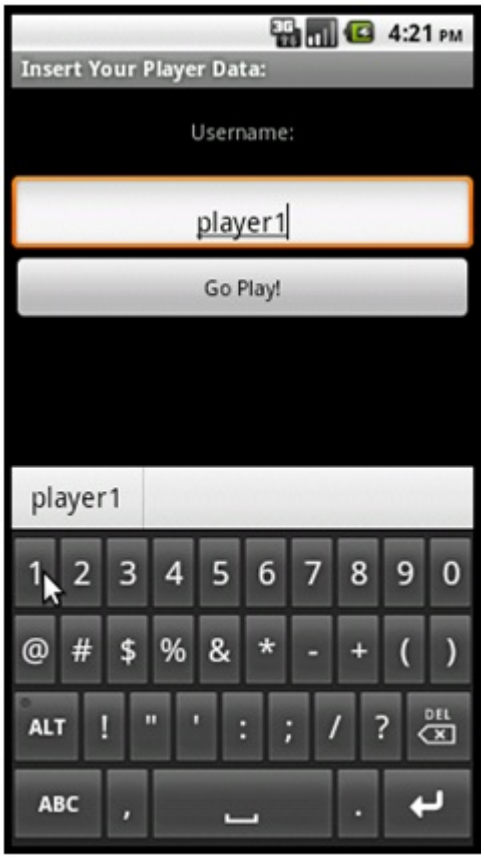

Figura 4.42: Selección de nombre de usuario

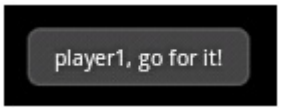

Figura 4.43: Mensaje de bienvenida a la partida

sistema, se crearán las estructuras de datos necesarias en la base de datos, y se notificará al jugador que ha entrado en la partida, mostrándole la pantalla principal del juego. Además, verá un mensaje como el que muestra la figura 4.43.

Pantalla principal del juego

La pantalla principal del juego se compone de cuatro apartados diferentes, con la información necesaria para la participación en una partida. Éstos apartados son los siguientes:

- Perfil: En esta sección se puede consultar la puntuación de la partida, los detalles del jugador, y la opción de salir del juego.
- Estado de las banderas: En esta sección se muestra una lista con el estado de las banderas de la partida.
- Mapa: En esta sección se muestra un mapa con la posición del jugador, así como la de los demás miembros de su equipo. Además se ve la posición de las banderas de la partida sobre el mapa y se muestran con el color en el que están.
- Mensajes: En esta sección se gestiona la bandeja de entrada de mensajes y se permite enviar un mensaje nuevo.

Sección perfil

La pantalla de perfil tiene la apariencia que muestra la figura 4.44.
Como se puede observar, en esta sección se muestra el nombre del jugador, el equipo al que pertenece, el color que identifica a su equipo, el nombre de la partida en la que está jugando, y la puntuación de todos los equipos de la partida.
Además, hay un botón mediante el cual se podrán interrumpir todos los servicios que utiliza la aplicación, se abandonará la aplicación, y se realizará una comunicación con el servidor para indicar que el jugador ha abandonado la partida, por lo que es necesario borrar de la base de datos toda la información

Figura 4.44: Pantalla de perfil

relativa a éste jugador, quedando disponible de nuevo la posibilidad de elegir éste nombre de usuario para futuros jugadores.
De manera transparente al jugador, cada intervalo predeterminado de tiempo, se hace una consulta al servidor para que devuelva una lista con la lista de equipos, de la misma manera que se realizó para mostrar la lista de selección de equipo. En este caso, lo que interesa es procesar el fichero JSON que devuelve el servidor de tal manera que se actualiza la pantalla y se muestran los puntos que lleva cada equipo. El jugador simplemente percibe que con el paso del tiempo cada equipo va aumentando su puntuación, debido a las capturas de las banderas.

Sección estado de las banderas

En esta sección, lo que se puede ver es un listado con los nombres de las banderas de la partida. De esta manera, el usuario puede comprobar de una forma rápida el estado de una bandera en concreto, así como sus coordenadas GPS, sin tener que utilizar el mapa. Para ello, solo tiene que seleccionar una bandera de las que aparecen en la lista, y se le muestra un mensaje con su estado.
De forma transparente al jugador, cada intervalo predeterminado de tiempo se está solicitando al servidor un listado de banderas con una petición GET que incluye la cabecera 'format=json', de tal manera que lo que se recibe es un fichero JSON con los detalles de las banderas. La aplicación procesa ésta información y genera el listado para que lo vea el usuario.

Figura 4.45: Pantalla de estado de banderas

Si algún manager decide eliminar o añadir una nueva bandera cuando una partida está en juego, o si el estado de una bandera cambia, el jugador puede percatarse del cambio.
La figura 4.45 muestra un ejemplo de la consulta del estado de una bandera.

Sección mapa

En esta sección, se muestra un mapa en el cual el centro de atención es el jugador. En este mapa se muestra un icono con forma de hombre y color similar al del equipo al que pertenece en el lugar en el que se encuentra el jugador, y otros tantos iconos iguales en la posición del resto de jugadores de su equipo. Además, se muestra en el mapa un icono para cada una de las banderas de la partida en el lugar en el que se encuentran. Estos iconos tienen también el color correspondiente al equipo que la posee en ese momento.
El jugador podrá en todo momento aumentar o reducir el zoom del mapa para visualizar de una manera mejor todo el área de juego, así como mover

Figura 4.46: Pantalla de mensajes

el mapa. Sin embargo, cada intervalo predeterminado de tiempo, el mapa se volverá a centrar en la posición en la que se encuentre el jugador.
Dado que la visualización de los mapas requiere la descarga de las imágenes de Internet, en ocasiones éstas no están disponibles, lo cual provoca que no se pueda tener una vista satélite del área de juego.

Sección mensajes

En este apartado del juego, se podrá acceder a la creación de un nuevo mensaje que será enviado al resto de jugadores, y también se podrá acceder a la bandeja de entrada de mensajes de la partida. Cada una de estas funcionalidades se ha implementado de forma separada, como se muestra en la figura 4.46.

Escritura de un nuevo mensaje

El jugador tendrá la posibilidad de crear durante una partida en mensaje que llegara al resto de jugadores. Esta capacidad dentro del juego hace posible que la estrategia cobre un factor importante, ya que los equipos pueden coordinar sus movimientos de tal forma que entre todos los jugadores pueden decidir qué hacer en tiempo real.

Figura 4.47: Pantalla de envío de mensajes

La comunicación mediante mensajes es similar a la que realiza un manager al crear un mensaje desde la aplicación web. Aunque lo correcto hubiera sido utilizar el método PUT en la URL de la creación de mensajes, como el servidor sólo implementa los métodos GET y POST, se ha decidido utilizar el método GET.
La figura 4.47 muestra una pantalla de este apartado de la aplicación.
El servidor, simplemente crea un nuevo mensaje y lo almacena en la base de datos. Dado que hay un servicio de mensajes en la aplicación que se encarga periódicamente en cada cliente de pedir los mensajes, pasado un cierto intervalo de tiempo todos los jugadores recibirán el nuevo mensaje en su terminal.
Si el mensaje se ha podido enviar correctamente, se muestra una notificación que así lo confirma en pantalla. Si, por el contrario, no se ha podido enviar, también se notificará, informando de la causa del error. No se permite el envío de mensajes en blanco, ni de mensajes superiores a 200 caracteres.

Bandeja de entrada de mensajes

Durante el transcurso de la partida, lo habitual será que lleguen mensajes, bien de otros jugadores, o bien del sistema. Cuando se reciba un nuevo mensaje, la aplicación avisará al jugador mostrando una notificación temporal en la pantalla, así como también reproducirá un sonido que recuerda a un

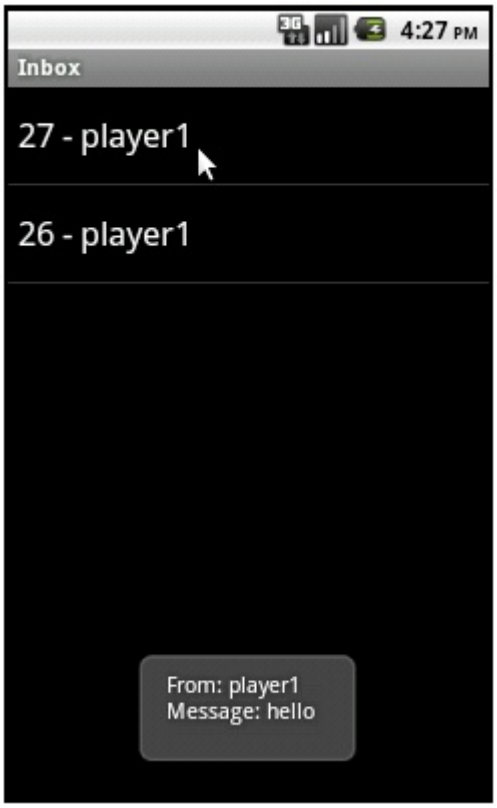

Figura 4.48: Pantalla de bandeja de entrada

claxon. La bandeja de entrada de mensajes es la parte del sistema en la que aparecerán todos los mensajes de la partida que se reciben. Se muestra un listado con los mensajes ordenado por el número de identificador del mensaje dentro del sistema, de más grande a más pequeño. De esta manera, los mensajes más recientes aparecerán antes, y los más antiguos después. En el listado de mensajes, simplemente se puede observar el identificador de mensaje del sistema, y el emisor de dicho mensaje. Para poder leer el texto de cada mensaje, el jugador sólo tiene que pulsar sobre la pantalla en el mensaje que desee, y el texto aparecerá en un globo de información durante un período corto de tiempo, como muestra la figura 4.48.

Servicios transparentes al usuario

De una manera transparente al usuario, hay unas secciones de código que se están ejecutando en la aplicación, y que permiten que tanto la lógica del juego como la interfaz del usuario se puedan estar actualizando en función de

cómo se va desarrollando una partida. Están implementados como actividades de Android, y son las siguientes:

- Controlador de banderas: Se trata de un servicio que periódicamente se encarga de solicitar al servidor las banderas de la partida, y recibe dichas banderas en formato JSON. Tras procesar este fichero y generar las instancias de cada bandera, las almacena, para poder ser mostradas con su estado en la sección de 'estado de las banderas', y ser mostradas en la sección 'mapa'.

- Location Listener: Se trata de un servicio que accede al GPS para obtener la posición del jugador. En lugar de realizar esta operación periódicamente, sólo se realiza cuando se detecta un cambio significativo en la posición del jugador. Este cambio está predeterminado a una distancia de 5 metros, es decir, cuando un jugador tenga una posición que esté a una diferencia de 5 metros de la última conocida, este servicio se encargará de enviar al servidor la nueva posición del jugador y la guardará en la base de datos. El resto de jugadores observará la nueva posición cuando soliciten la posición del resto de jugadores al servidor.

- Controlador de jugadores: Este servicio se encarga periódicamente de solicitar al servidor los datos de los jugadores que estén participando en el mismo equipo del jugador que lo solicita. Tras recibir una respuesta en formato JSON y procesarla, guardará la información de cada jugador, y podrá ver la nueva posición de cada uno de ellos en la sección 'mapa'.

- Controlador de distancias: Este servicio se encarga periódicamente de calcular la distancia del jugador a cada una de las banderas de la partida. Si el jugador se encuentra a una distancia menor de 10 metros de una bandera, mandará al servidor una petición para que cambie el estado de dicha bandera y le ponga el color de su equipo, ya que se considera que ha sido capturada.

- Controlador de puntos: Este servicio se encarga periódicamente de comprobar el estado de las banderas del sistema, y por cada bandera que tenga el color de su equipo, enviará una petición al servidor para sumar un punto a su equipo, ya que las banderas que permanecen en un color, suman periódicamente un punto. De este modo se implementa la funcionalidad consistente es que a mayor número de banderas de un color, mas puntos suma un equipo.

- Controlador de mensajes: Este servicio se encarga de solicitar al servidor periódicamente la lista de mensajes del sistema. Una vez recibida la información en formato JSON, ésta es procesada, y si se detecta que hay nuevos mensajes en comparación con la petición anterior, se notifica al usuario mediante un mensaje en pantalla y un sonido de claxon. La lista de mensajes de entrada 'inbox' también se verá actualizada.

4.3. Pruebas

4.3.1. Pruebas del servidor y servicio web

Hay que diferenciar dos fases a la hora de hacer las pruebas del servicio web. Django proporciona un servidor de desarrollo que es muy útil a la hora de realizar las pruebas del código que se va desarrollando. La primera fase de pruebas consiste en la ejecución del servicio en este servidor proporcionado por Django.
Este servidor se limita a dar servicio en la máquina en la que se está desarrollando. Para ello, sirve el contenido en la dirección de 'localhost' de la máquina, en el puerto que se le indica. Para esta aplicación, este servidor es más que suficiente, ya que cuando todo el sistema funciona correctamente en este servidor, sólo se necesita cambiarlo a un servidor de producción.
En este PFC, el servidor de producción es un servidor apache con el módulo mod_wsgi que compatibiliza la capa Apache con la capa Django que hay debajo. La aplicación ha sido probada paralelamente con el desarrollo del código en el servidor de desarrollo de Django. Sin embargo, es necesario hacer una serie de pruebas para el servidor de producción, en los que los retardos de las peticiones aumentan en relación con los tiempos de respuesta que se obtenían en la dirección de 'localhost'.

4.3.2. Pruebas de la aplicación móvil

Las pruebas para la aplicación móvil también se deben diferenciar en dos fases principales: desarrollo y producción.
En la fase de desarrollo, dado que no se disponía de un teléfono real sobre el que probar la aplicación, las pruebas se han realizado en el emulador de dispositivos Android que proporciona el entorno de desarrollo Eclipse.
Este emulador utiliza también una dirección de 'localhost' para comunicarse con el servidor de desarrollo de Django. De este modo, se han podido probar las interacciones que se producirán a la hora de tener la aplicación en un teléfono real con conexión a Internet.

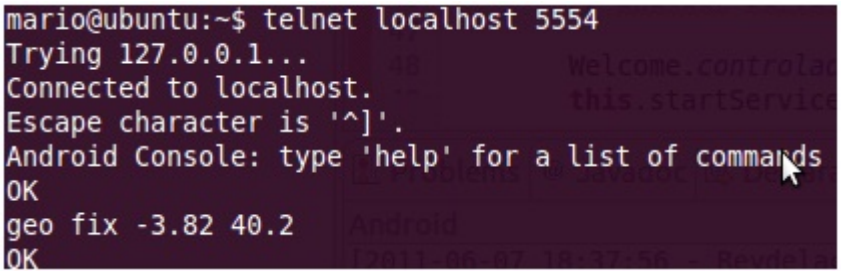

Figura 4.49: Introducción de coordenadas GPS para el simulador

Por otra parte, se ha podido simular la obtención de coordenadas de GPS del teléfono. Para ello se debía hacer una conexión telnet al puerto 5554 de la máquina de desarrollo, y mediante el comando 'geo fix' se podían enviar las coordenadas que se desearan al teléfono emulado, como se muestra en la figura 4.49.

Esta es una buena forma de probar que la lógica del juego se cumple de manera correcta cuando los jugadores alcanzan posiciones GPS determinadas.

El emulador ha permitido probar la aplicación en diferentes versiones del sistema Android, habiendo decidido que funcione a partir de la versión 2.1 del mismo como versión mínima.

Sin embargo, a la hora de llevar el juego a un teléfono real, y conectando a través de Internet al servidor de producción, pueden aparecer otra serie de problemas. Por este motivo, es conveniente realizar una prueba del juego con teléfonos reales y jugadores reales. Además, la realización de pruebas permite obtener información muy valiosa de los jugadores que lo han probado, como sugerencias de nuevas funcionalidades, cambios propuestos para el sistema, etcétera.

4.3.3. Primera prueba de una partida real

El hecho de tener resultados muy satisfactorios en los emuladores, no significa que, a la hora de realizar pruebas con teléfonos y jugadores reales, todo vaya a funcionar perfectamente. Además de encontrar errores en la aplicación, se obtiene también una información muy útil referente a la usabilidad de la aplicación, lo cual hace que los cambios y las futuras versiones tengan menos fallos y una mayor calidad de producto.

El día 14 de Junio de 2011, en el campus de Fuenlabrada de la Universidad Rey Juan Carlos, se realizó una primera prueba con jugadores y dispositivos reales. Dicha prueba tuvo tres fases: explicación del juego y preparación de la partida, desarrollo de la partida, y encuesta final.

En primer lugar, se explicó a los voluntarios el funcionamiento del juego, y se les repartió a cada uno de ellos un teléfono con conexión a Internet. Así mismo, se explicó la forma de operar en el caso de encontrar alguna dificultad o problema en la aplicación. Esto supuso un tiempo de 20 minutos aproximadamente.
En segundo lugar, los voluntarios participaron en la partida por el campus de la universidad mientras un administrador del sistema monitorizaba el desarrollo de la partida en un ordenador. Durante dichas pruebas, uno de los voluntarios presentó problemas en la aplicación y regresó a los pocos minutos. El resto de los jugadores siguieron jugando hasta el final de la partida. La partida tuvo una duración de unos 40 minutos.
Por último, tras acabar la partida y regresar los voluntarios, se pidió que contestaran una encuesta, la cual se puede ver en las figuras 4.50, 4.51, 4.52, 4.53, 4.54 y 4.55. Las conclusiones que se obtuvieron de la prueba fueron las siguientes:

- Para los jugadores que habían jugado a algún juego similar anteriormente en algún formato de videojuego, no eran necesarias explicaciones adicionales, por lo que podemos concluir que la adaptación del juego original a la versión del PFC está bien lograda.

- La interfaz de usuario fue valorada como una interfaz normal dentro de las aplicaciones Android, pero con un toque de originalidad, lo cual hace que la interfaz que presente el juego sea bastante adecuada para la funcionalidad que queremos. Sin embargo, se reseñó que la utilidad de la pestaña 'Flag Stats' no era muy útil, ya que la información que proporcionaba se podía visualizar también en la pestaña 'Map'. Por tanto, es posible que en futuras versiones de la aplicación esta opción desaparezca por no tener utilidad y consumir recursos en el terminal.

- El sistema de GPS presentó problemas a la hora de localizar bien a los jugadores. Este problema aparece ahora por primera vez ya que en el simulador las coordenadas de GPS eran enviadas a través del comando 'geo fix', por lo que concluimos que esta parte del código debe ser revisada.

- También se detectaron problemas con la aplicación, en el sentido de que la interfaz gráfica no respondía en algunas ocasiones y el sistema Android mostraba el mensaje de que la aplicación no responde. Esto sucede porque en algún momento se produce una espera activa, probablemente al esperar una respuesta del servidor y procesarla, por lo

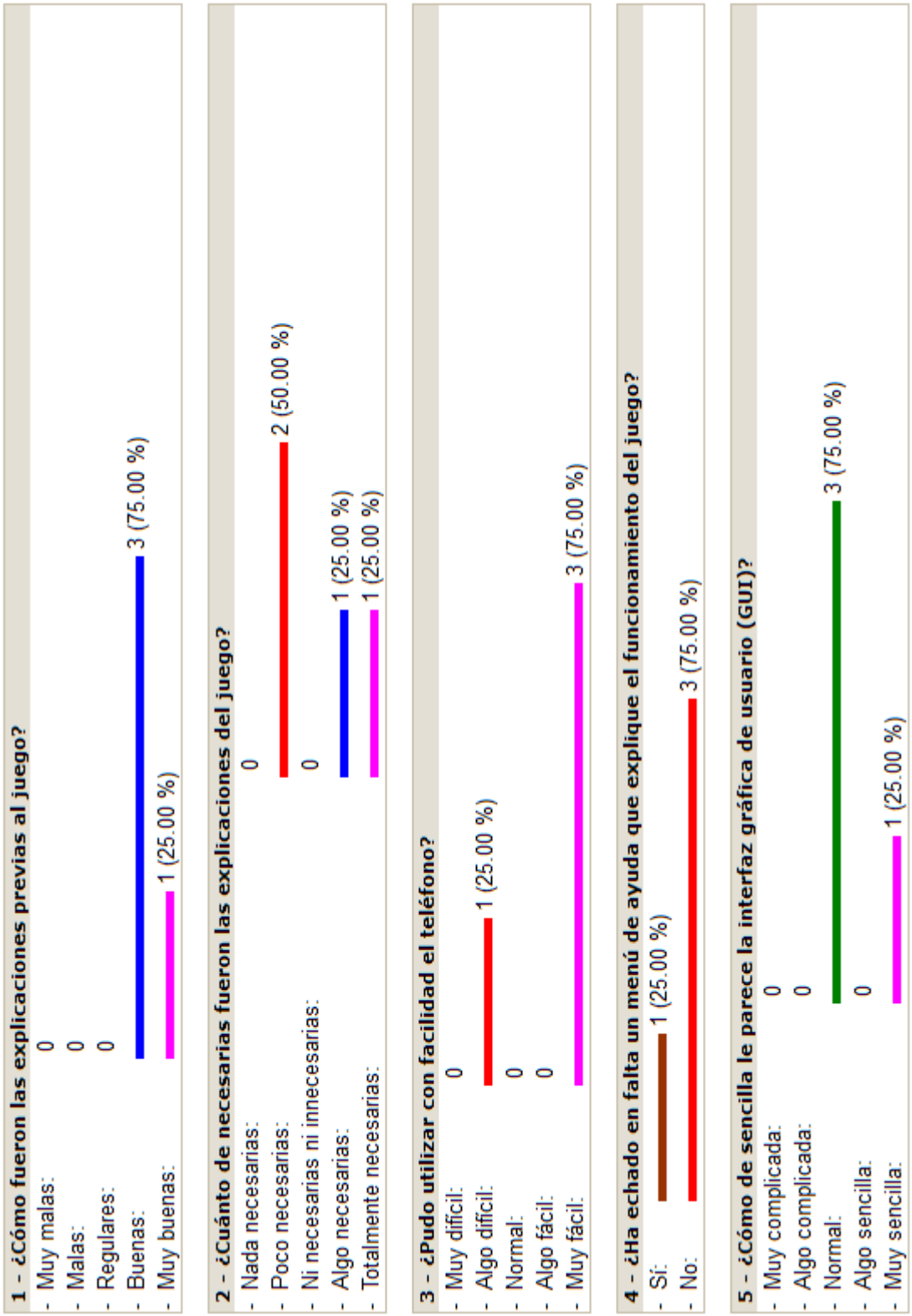

Figura 4.50: Resultados de la encuesta

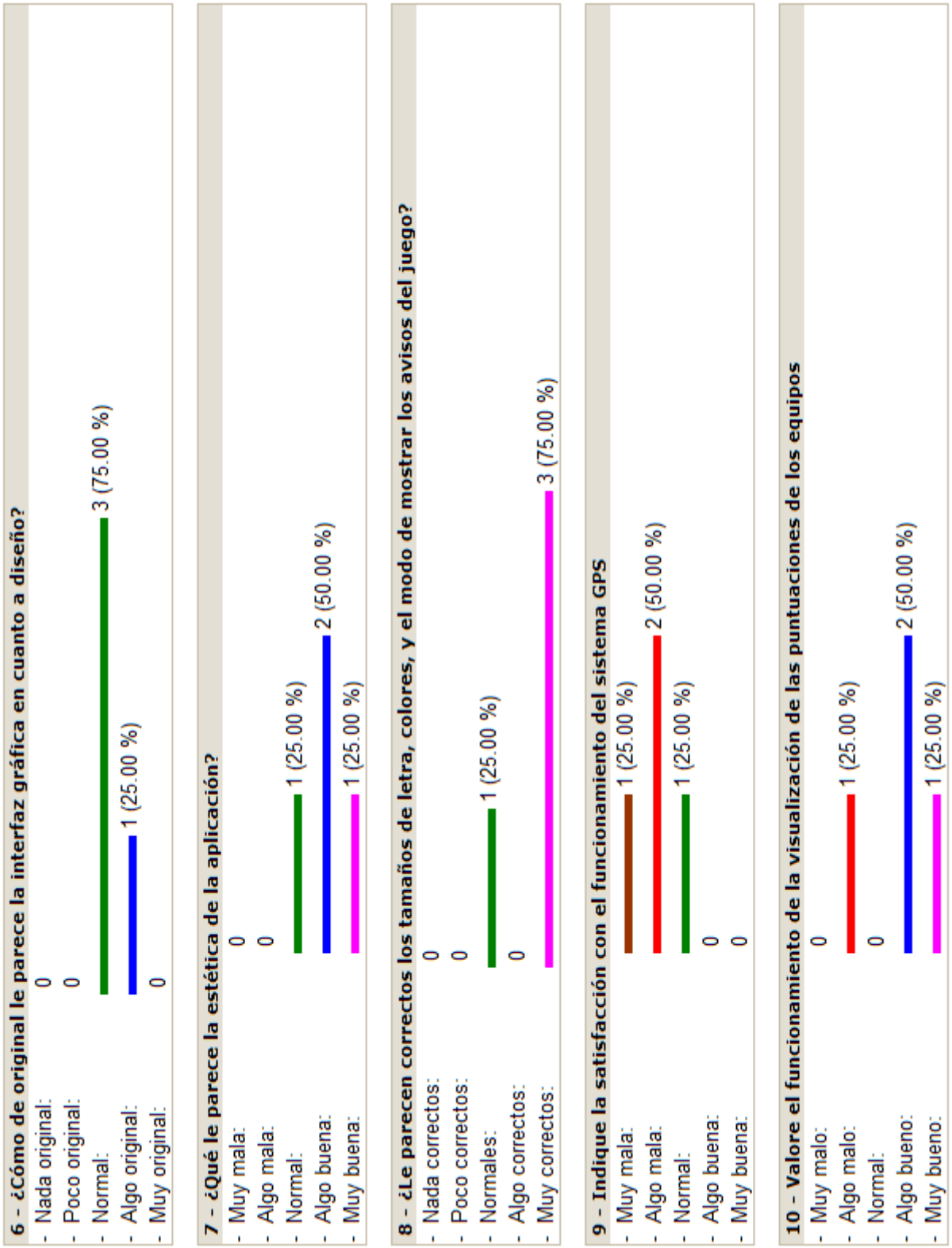

Figura 4.51: Resultados de la encuesta

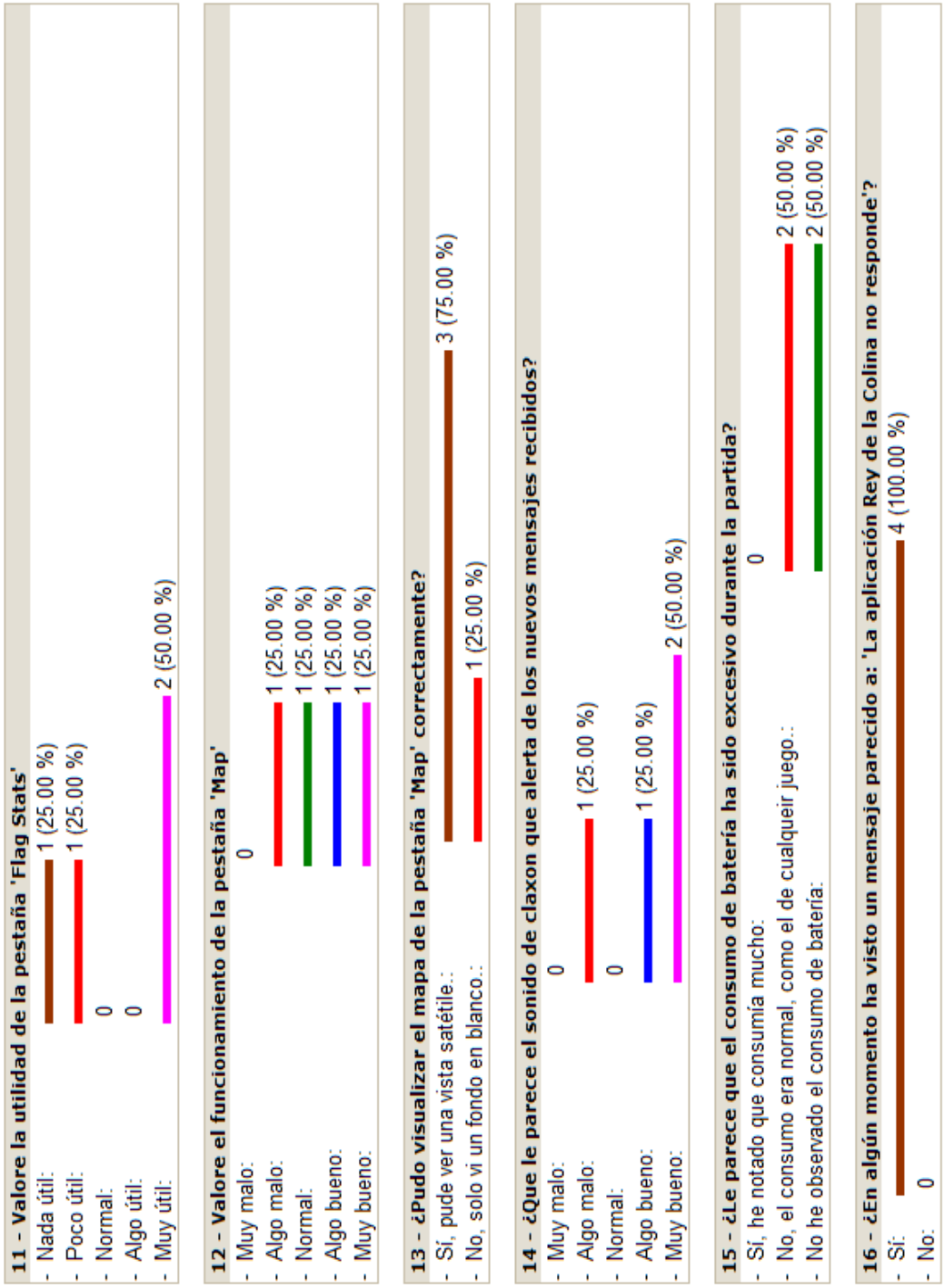

Figura 4.52: Resultados de la encuesta

17 - Indique cuánto está familiarizado con la tecnología Android.

- Es la primera vez que lo oigo: 0
- He oído algo, pero no se bien qué es: 0
- Conozco algo, pero no mucho: 2 (50.00 %)
- Conozco bastante de Android: 2 (50.00 %)
- Soy un experto/desarrollador: 0

18 - ¿Ha sufrido el cierre inesperado de la aplicación?

- Sí: 3 (75.00 %)
- No: 1 (25.00 %)

19 - En caso afirmativo, explique detalladamente qué estaba haciendo cuando la aplicación se cerró.

- Al principio, antes de salir a jugar
- La aplicación intentó cerrarse, casi ninguna pestaña funcionaba excepto la de \"flag stats\", ni recibía los mensajes.
  Salí del juego y tuve que iniciar sesión con un usuario diferente, pues, me decía que \"ya existía\".
  Poco después los mismos fallos se repetían.
- Varias veces han sido intentando ampliar el mapa, aunque la mayoría de las veces ha sido intentando leer los mensajes.
- el cierre se ha producido mientras intentaba mirar la ventana de mensajes

20 - Valore la experiencia con respecto al funcionamiento general de la aplicación

- Muy desagradable: 0
- Algo desagradable: 0
- Ni agradable ni desagradable: 3 (75.00 %)
- Algo agradable: 1 (25.00 %)
- Muy agradable: 0

21 - ¿La aplicación funcionaba de forma fluida?

- Muy poco fluida: 2 (50.00 %)
- Poco fluida: 2 (50.00 %)
- Fluidez normal: 0
- Algo fluida: 0
- Muy fluida: 0

Figura 4.53: Resultados de la encuesta

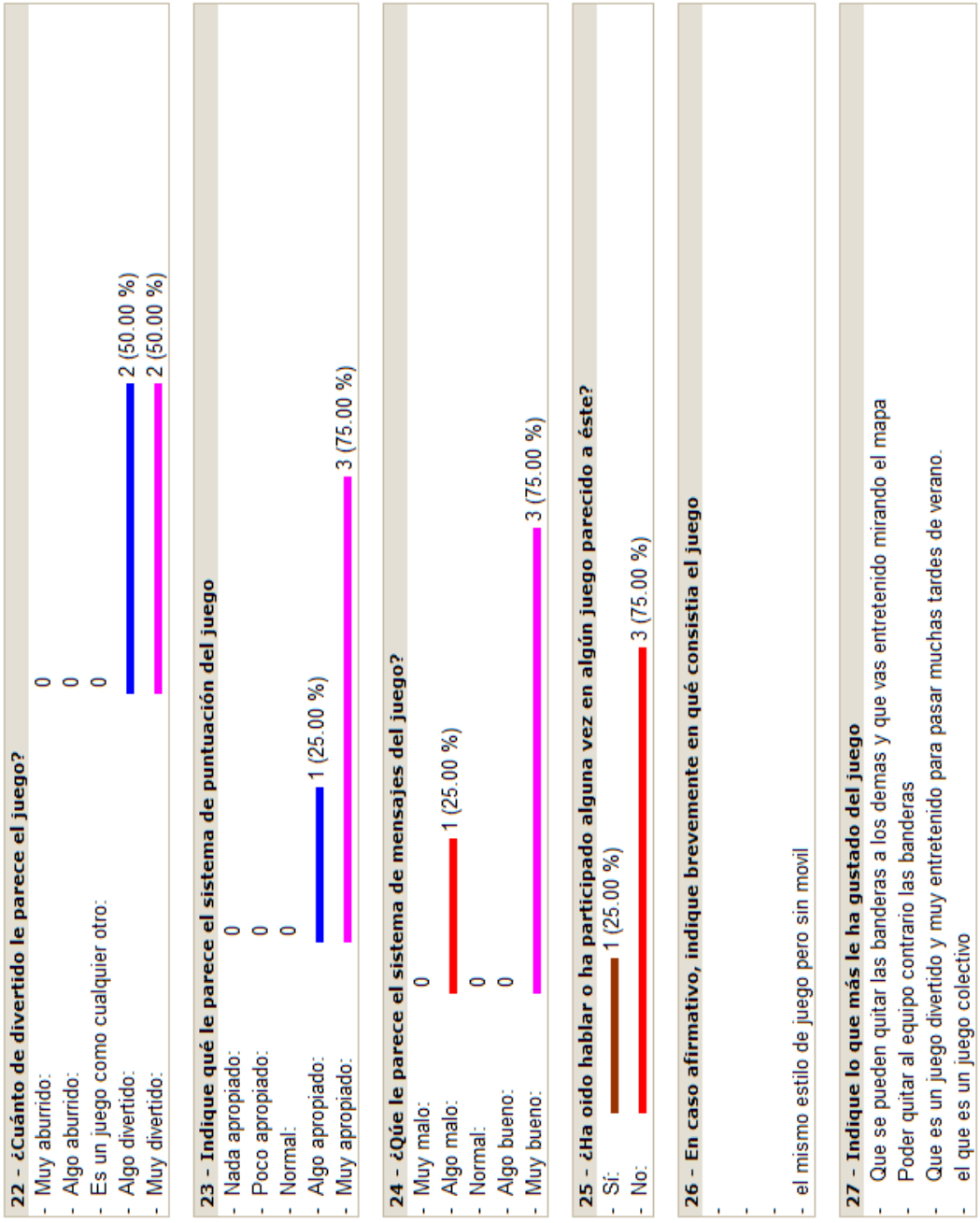

Figura 4.54: Resultados de la encuesta

28 - Indique lo que menos le ha gustado del juego

- Que los puntos de la partida estaban muy lejos y que venian detras de mi para quitarme las banderas jeje
- La lentitud del juego en general, y el gps que no captaba los mapas
- Que se bloquease tanto. Es una pesadez enorme que se bloquee tanto.
- que cuando llega un momento del juego este ya no reacciona

29 - Explique, si las hubiera, las cosas que cambiaría en el juego y por qué

- Si hubiese alguna manera de que fuese mas rapido....
- Ninguna
- Aumentaría el perímetro de alcance, por ejemplo, Fuenlabrada entera.
- en principio no cambiaría nada

30 - Indique las cosas que eliminaría del juego por considerarlas innecesarias

- la segunda pestaña si no te conoces la zona no la usas
- Creo que ninguna
- La pestaña flags, porque en el propio GPS se ve el color.
- nada

31 - ¿Qúe cosas añadiría al juego, o ha echado en falta?

- Alguna animación mas
- El menú de ayuda
- Mayor alcance de perímetro, como puse antes.
- si acaso que las banderas se quedaran un rato fijas

32 - Escriba aquí comentarios y sugerencias que no se hayan recogido en la encuesta, o los posibles errores que haya encontrado.

- Nada
- No tengo sugerencias
- No se si el reiterado bloqueo era por la aplicación o por el móvil, aunque pienso yo que es por el móvil, es decir, que habría que usar móviles mejores que vayan más fluidos.
- nada

Figura 4.55: Resultados de la encuesta

que esta funcionalidad se pasará a realizar en un hilo de ejecución alternativo que no bloquee la interfaz gráfica de usuario. Además, esto provocaba que la aplicación no funcionara de manera fluida.

- En cuanto al sistema de mensajes, y al sistema de puntuación del juego, tuvo un grado mayoritario de aceptación, siendo calificado como muy apropiado por el 75 % de los participantes.

- El juego gustó bastante por tratarse de un juego colectivo que ha sido calificado como entretenido por los jugadores que lo probaron, por lo que la lógica del juego se considera como muy buena.

- Se ha detectado un problema en la lógica del juego, y es que se perjudica al jugador que llega primero a las banderas, ya que otro que llegue más tarde es el que finalmente se lleva los puntos. Por tanto, se debe cambiar la lógica del juego en este aspecto. Una posible solución, será, bloquear la captura de esa bandera durante un tiempo, o dar una cantidad de puntos extra al jugador que llega primero.

- En cuanto al coste de la partida, se observó el saldo de las tarjetas de prepago antes y después de la partida, siendo la diferencia tras el juego de 1.60 euros de media. Teniendo en cuenta que la operadora cobra un euro por el simple hecho de conectar a la red 3G, podemos concluir que el tráfico de datos de la aplicación consumió poco, aproximadamente unos 60 céntimos de euro por 40 minutos de juego. Si los jugadores dispusieran de tarifas de datos mensuales en su móvil, el coste de participar en este juego sería cero. En cambio, para un usuario que no tenga tarifa de datos, tendrá que abonar una media de 2 euros por una partida de una hora, lo cual es bastante económico.

Capítulo 5

Conclusiones

Esta parte consiste en una valoración crítica del éxito o fracaso de los objetivos que se pretendieron conseguir al inicio del PFC. Para ello, en la sección de objetivos logrados se exponen los éxitos obtenidos durante la realización, en la sección de errores y problemas se detallarán las dificultades que han tenido que superarse, y las que no se han podido superar. Finalmente, se exponen las próximas ideas con las que debería seguir el PFC en el desarrollo futuro.

5.1. Objetivos logrados

La mejor manera de explicar los objetivos logrados es respondiendo a las siguientes dos preguntas: ¿Qué he aprendido? y ¿Qué he conseguido? . La respuesta de estas preguntas es la que conforma esta sección del presente PFC:

- Aprender el concepto de 'Outdoor Gaming', y en qué consiste: Hasta que no se empezó con la investigación referida a juegos similares o antecedentes de la idea que se propone para el proyecto, no conocía la existencia de este concepto. Fue a partir de las investigaciones y la profundización en el tema cuando se ha descubierto que ya existen una serie de juegos que han explotado las características de un GPS para utilizarlas en forma de ocio, y que ya existen otras formas de jugar a juegos como 'Rey de la colina' en otras plataformas o versiones anteriores.

- Aprender el funcionamiento del sistema Android: Lo único que se sabía

al comienzo del PFC es que había una serie de teléfonos móviles que utilizaban el sistema Android. Gracias a la realización del PFC, ahora sé que no estamos sólo ante un sistema operativo, sino que además se compone de una serie de aplicaciones básicas, y que se trata de un sistema para el cuál cualquier desarrollador puede crear aplicaciones. He aprendido las características técnicas del sistema, de que se compone, como funciona, y las posibilidades que ofrece. Para ello, ha sido necesaria la lectura de diversos manuales y libros relacionados con la materia, pero sin duda el aprendizaje ha sido muy interesante.

- Aprender un nuevo lenguaje de programación, Python: Con el uso de tutoriales para Django y los numerosos ejemplos que se encuentran en la web, he aprendido a programar en Python, por lo que ha sido una experiencia enriquecedora, ya que durante la carrera no lo había hecho antes.

- Aprender a utilizar el framework de Django: Hasta antes de la realización del PFC, no conocía absolutamente nada de la existencia de Django. La propuesta del profesor para utilizar este framework y ahorrar mucho tiempo en el desarrollo del servidor ha sido otra de las cosas que he aprendido. Esto me ha enseñado que hay una gran cantidad de herramientas para desarrollo software de código libre que son bastante buenas, y que merece la pena utilizar en lugar de tratar de construir todo por cuenta propia.

- Aprender a utilizar libregeosocial: Al igual que sucedía con Django, desconocía el concepto de red social móvil. Libregeosocial es una buena implementación de este concepto, por lo que también he aprendido no sólo a utilizar esta red social, sino que he conocido un concepto que no había imaginado.

- Aprender a utilizar PostgreSQL: Si bien es cierto que ya se tenían conocimientos previos de SQL gracias a los estudios de la carrera, nunca había utilizado este software antes. Además, el hecho de tener una base de datos un tanto especial, al permitir el almacenamiento de datos de carácter de geolocalización, hace que haya sido una experiencia nueva, al tener que haber aprendido el funcionamiento de PostGIS.

- Aprender JSON: Normalmente, a la hora de transmitir información de objetos en formato de texto, lo que se suele aprender es XML. De hecho, durante la carrera, es el único formato que se ha utilizado para las diferentes asignaturas o prácticas. Gracias a este PFC, he tenido

la oportunidad de conocer y aprender a utilizar este otro formato, que cada vez se está extendiendo más y se está haciendo más popular.

- Conseguir desarrollar una aplicación que utiliza tecnología actual: Dada que la incorporación del sistema Android a los teléfonos móviles y la utilización de los sistemas GPS para ocio es algo relativamente novedoso, suponía un reto desarrollar una aplicación que funcionara con estas tecnologías. Del mismo modo, lo habitual en la titulación era desarrollar aplicaciones que típicamente deberían ejecutar sobre un ordenador. Dada la capacidad de procesamiento que están alcanzando los móviles en la actualidad, suponía un reto interesante el hecho de crear aplicaciones que fueran a ser ejecutadas en un entorno en el que la memoria y la capacidad de procesamiento iban a ser una de las limitaciones de la aplicación.

- Conseguir adaptar un juego clásico a una nueva forma de juego: La idea de llevar a escenarios reales el juego que se propone es una novedad. Este tipo de juegos normalmente se han llevado a cabo con personajes y escenarios virtuales en juegos online. En esta propuesta, se ha conseguido llevar una lógica de juego similar al mundo real, en el que los jugadores siguen interactuando entre sí mediante Internet, pero dichos jugadores han pasado a ser poseedores de un teléfono móvil, y los escenarios han pasado a ser lugares en el mundo real.

- Conseguir desarrollar el sistema siguiendo una planificación ingenieril: El método que se ha seguido desde el momento en el que se propuso la idea hasta el momento en el que el sistema ha sido terminado y funciona, ha intentado seguir un modelo de producción de software típico. Para ello, primero se ha expuesto la idea y se ha profundizado en ella, después se han valorado las posibilidades de llevarlo a cabo, después se han establecido los requisitos y se han estudiado las tecnologías que se iban a necesitar, después se pensó en la arquitectura y el diseño del sistema para, finalmente, comenzar a implementar y probar las nuevas partes del sistema, hasta la finalización de la aplicación. Es en este momento cuando se propuso la instalación del sistema en máquinas y dispositivos reales para realizar las pruebas finales. Aun así, el ciclo vuelve a comenzar con el análisis, diseño, implementación y pruebas de nuevas funcionalidades, por lo que el desarrollo que se ha seguido ha sido iterativo e incremental.

5.2. Errores y problemas encontrados

La aparición de errores es un hecho que sucede con frecuencia en el desarrollo de aplicaciones informáticas. En este PFC han aparecido numerosos problemas que se han tenido que resolver, y se han dado errores típicos del desarrollo de aplicaciones para telefonía y del desarrollo de servicios web. Además, se han dado errores propios que derivan del propio uso de las tecnologías implicadas:

- Uno de los primeros errores que se ha tenido en cuenta durante todo el desarrollo del sistema es el que deriva de la utilización del GPS. El sistema de GPS para la localización de elementos en un mapa lleva implícito un margen de error consistente en la precisión del cálculo de la posición. Todos los sistemas de GPS tienen un error de precisión. Algunos sistemas avanzados manejan errores de precisión del orden de centímetros, pero en el caso de teléfonos móviles, el error que se maneja es de unos cuantos metros. Esto ha tenido que tenerse en cuenta a la hora de determinar si un jugador se encuentra en la posición de una bandera. Se ha manejado este error aceptando que un usuario está en un lugar determinado si su posición está dentro de un radio de 10 metros con respecto a la posición de una bandera.

- Otro de los aspectos que ha dado mayores complicaciones ha sido el momento de trasladar la aplicación del servidor de desarrollo al servidor de producción, ya que había que configurar correctamente apache y el módulo mod_wsgi para que se comunicara con Django, además de indicar las nuevas rutas para encontrar el contenido estático del servidor, como imágenes, plantilla CSS, ficheros HTML, etcétera. La manera de solucionar este problema ha sido mediante la consulta de tutoriales oficiales y ejemplos que se encargaban de resolver esto.

- Durante el desarrollo del sistema, se ha producido un problema hardware en la máquina de trabajo. Concretamente, se estropeó la tarjeta gráfica y se tuvo que enviar dicha pieza a reparación. Mientras se reparaba, hubo que repetir la instalación de todo el software en otra máquina nueva y continuar trabajando en otra máquina de características diferentes. En este punto cobran importancia las copias de seguridad. Se perdió el trabajo realizado de un par de días por no realizar la copia de seguridad, por lo que una de las cosas que se han aprendido es que se debe mantener todo el trabajo salvado desde el momento en el que

se termina de hacer. Esto provocó la pérdida de un día para preparar la nueva máquina con todo el software necesario, y además requirió dos días adicionales para volver a restablecer el trabajo perdido.

- Derivado de la nueva máquina de desarrollo, se debieron instalar librerías que no eran necesarias en la primera. El primer computador tenía un procesador y un sistema de 32 bits, mientras que el segundo computador tenía un procesador y un sistema de 64 bits. El entorno de desarrollo Eclipse y el emulador funcionan muy bien en sistemas de 32 bits, no siendo así en sistemas de 64 bits. Se necesitaron descargar e instalar un conjunto de librerías que garantizaran el funcionamiento del emulador en un sistema de estas características, aunque el rendimiento a la hora de ejecutar el emulador era inferior, por lo que a partir de ese momento las pruebas comenzaron a tomar un poco más de tiempo del habitual.

- Con las primeras pruebas en un terminal HTC Magic[25], cuyas capacidades de procesamiento son bajas, se observó que la aplicación no tenía una ejecución fluida. Por tanto, se decidió optimizar la aplicación de tal manera que el procesamiento fuera más ligero. Esto implicó el estudio de buenas prácticas a la hora de programar threads y procesos de Android, de tal manera que la interfaz de usuario de la aplicación no se viera afectada en tanta medida por el procesamiento que requería la aplicación. Aun así, la capacidad de procesamiento del terminal era demasiado baja, por lo que la aplicación no funcionaba como se esperaba. Adicionalmente, se realizaron pruebas en otros terminales más rápidos. Concretamente, un HTC Wildfire[26], Samsung Galaxy 3[27], y Samsung Galaxy Ace[27]. En estos terminales la aplicación funcionaba correctamente y de una manera fluida, por lo que el problema era que la aplicación requería de un terminal con capacidad de procesamiento mayor que el que proporcionaba el primer teléfono de prueba.

- Debido a todos estos problemas, la planificación del proyecto que se ideó en un primer momento no se pudo ir cumpliendo, y se tuvieron que realizar modificaciones según iban apareciendo los diferentes problemas. De esto se ha aprendido que realizar una buena planificación es una tarea complicada, que requiere de mucha experiencia previa en la construcción de sistemas similares, y que aún así, siempre aparecerán problemas que fácilmente harán que es desarrollo se dilate en el tiempo más de lo previsto.

5.3. Posibles trabajos futuros

En esta parte de la memoria del PFC se tratan los siguientes pasos que se podrían seguir. Los principales puntos serían los siguientes:

- Añadir en el sistema una funcionalidad que permitiera a los jugadores de los teléfonos móviles registrarse desde la aplicación y poder participar en las partidas con un nombre de usuario único para cada jugador. Además, se podrían almacenar estadísticas de los jugadores como por ejemplo el número de victorias conseguidas, el número de derrotas, el número de banderas capturadas, la cantidad de puntos conseguidos, etcétera. Esta información serviría para realizar estadísticas y conocer a los jugadores más efectivos a la hora de jugar a este juego.

- Incluir nuevas variantes de juego, como por ejemplo partidas cuyo límite no sea un número determinado de puntos, sino un máximo de tiempo. En este nuevo caso, sería vencedor el equipo que más puntos tuviera al transcurrir un tiempo determinado.

- Incluir un sistema de partidas para un jugador. El sistema propondría una serie de retos en los cuales en cada partida sólo habría un jugador. Éste jugador debería alcanzar los puntos geográficos en un tiempo determinado.

- Construir un simulador de jugadores virtuales. En este caso, podrían existir jugadores controlados por el sistema, de tal manera que un jugador podría jugar una partida contra adversarios que no están físicamente jugando, y que podría servir para el entrenamiento de jugadores reales.

- Incluir banderas especiales que requieran de varios jugadores para su captura, o superar una prueba para conseguir capturarla, además de llegar hasta ella. Por ejemplo, podría haber una bandera que precisara de la presencia de dos o tres jugadores de un mismo equipo para poder empezar a puntuar para ese equipo. Por otra parte, podrían existir banderas que necesitaran de la resolución de algún puzle para poder conquistarla, como por ejemplo contestar a una pregunta relacionada con el lugar en el que está ubicada, o una pregunta de cultura general, que el sistema debería validar para permitir la conquista de la bandera.

- Comercializar el juego en Android Market, de tal manera que la obtención de la aplicación y su instalación en el terminal reportara ingresos al desarrollador. Otra opción sería vender la idea a alguna empresa que estuviera interesada en seguir desarrollando la aplicación, o ceder los derechos de explotación de la misma a cambio de una compensación económica.

Bibliografía

[1] Sitio web de Alleycat Races en Wikipedia. Accedido en Junio de 2011. `http://en.wikipedia.org/wiki/Alleycat_races`

[2] Sitio web de Geocaching en castellano. Accedido en Junio de 2011. `http://www.geocaching-hispano.com`

[3] Sitio web de Geodashing. Accedido en Junio de 2011. `http://www.gpsgames.org`

[4] Sitio web de Highpointing. Accedido en Junio de 2011. `http://www.highpointers.org/`

[5] Sitio web de Letterboxing. Accedido en Junio de 2011. `http://www.letterboxing.info/`

[6] Sitio web de la cadena de televisión Fox. Accedido en Junio de 2011. `http://www.foxtv.es/`

[7] Bungie Studios. Sitio web del videojuego *Halo: Combat evolved*. Accedido en Junio de 2011. `http://halo.xbox.com/en-us/intel/titles/halocombatevolved`

[8] Rare y 4J Studios. Sitio web del videojuego *Perfect Dark*. Accedido en Junio de 2011. `http://perfectdark.retropixel.net/pd/`

[9] Google Inc. Sitio web de Android. Accedido en Junio de 2011. `http://www.android.com/`

[10] Burnette, E. 2010. *Hello Android*.

[11] Haseman, C. 2008. *Android Essentials*. Editorial Apress.

[12] Murphy, M. L. 2010. *Beggining Android 2*. Editorial Apress.

[13] Sitio web de Python. Accedido en Junio de 2011.
http://www.python.org/

[14] Sitio web de Django. Accedido en Junio de 2011.
https://www.djangoproject.com/

[15] Sitio web de Apache. Accedido en Junio de 2011.
http://www.apache.org/

[16] Libresoft, GSYC, URJC. Sitio web de libregeosocial. Accedido en Junio de 2011.
http://www.libregeosocial.org/

[17] Sitio web de PostgreSQL. Accedido en Junio de 2011.
http://www.postgresql.org/

[18] OSGeo Proyect. Sitio web de PostGIS. Accedido en Junio de 2011.
http://postgis.refractions.net/

[19] Sitio web de JSON. Accedido en Junio de 2011.
http://www.json.org/json-es.html

[20] Richardson, L. y Ruby, S. 2007. *RESTful Web Services*. Editorial O'Reilly.

[21] Sitio web de Eclipse. Accedido en Junio de 2011
http://www.eclipse.org/

[22] Sitio web de Ubuntul. Accedido en Junio de 2011.
http://www.ubuntu.com/

[23] Libresoft, GSYC, URJC. Fernández, J. Sitio web de Gymkhana Project. Accedido en Junio de 2011.
http://gymkhana.libresoft.es/indice.html

[24] Google Inc. Sitio web del módulo WSGI para Django y Apache. Accedido en Junio de 2011.
http://code.google.com/p/modwsgi/

[25] HTC. Sitio web de HTC Magic. Accedido en Junio de 2011.
http://www.htc.com/es/product/magic/overview.html

[26] HTC. Sitio web de HTC Wildfire. Accedido en Junio de 2011.
http://www.htc.com/es/product/wildfire/overview.html

[27] Samsung. Siti web de Samsung. Accedido en Junio de 2011. `http://www.samsung.com/es/`