Universidad del País Vasco
Euskal Herriko Unibertsitatea

Proyecto de Fin de Carrera

INGENIERÍA INFORMÁTICA

euskadiMoving: sistema de seguimiento de transporte público

Autor
Xabier Valdecantos

Julio de 2013

Índice general

Índice de figuras

Indice de tablas

Resumen

El tema abordado por este proyecto es la información disponible acerca de los horarios del transporte público en las ciudades, concretamente centrado en la ciudad de Donostia - San Sebastián. En esta ciudad, la información ofrecida a los usuarios está centrada en:

- Tiempo que falta para que lleguen los autobuses. Únicamente accesible en las marquesinas.
- Horarios con la primera hora de salida y la frecuencia de la línea.

Estos métodos de información se pueden mejorar mediante un sistema de información en tiempo real, el cual informe de la posición actual de cada autobús de cada línea a petición del usuario. Asimismo, siendo esta información accesible a través de un servicio web, se evita el tener que desplazarse a las marquesinas para informarse.

Por tanto, estos son los objetivos que se consideran para el proyecto. La realización de un servicio web que muestre a petición del usuario las líneas de transporte público disponibles y este pueda seleccionar cuales visualizar, de manera que al seleccionar una línea se muestren todos los vehículos de dicha línea y su posición actual.

1. CAPÍTULO

Introducción

Hoy en día el acceso a internet en el primer mundo está ampliamente consolidado. De ello que constantemente se consulte a través de internet todo tipo de información, como podría ser el horario de las sesiones de un determinado cine, el horario de atención del centro de salud mas cercano o el precio de nuestro coche preferido.

Ha llegado un punto en el que existe la sensación que aquello que no esté en internet no existe. En este aspecto, los medios de transporte público, han adaptado la información existente de horarios y por norma general, simplemente los han adaptado para que sean accesibles a través de internet en un formato prácticamente igual al físico.

Observado por tanto el potencial acceso de la población a internet y las oportunidades que este ofrece, nace la idea de ofrecer a los usuarios del transporte más información de la que han tenido accesible hasta el momento. Ofrecerles una información a nivel de línea en la cual se informe de la posición actual y en tiempo real, del autobús, tren o tranvía de turno.

A través de la oferta de este servicio, se pretende acercar el transporte urbano, entre otros, a toda persona que sea reticente a su uso por desconocimiento y dificultad de entendimiento de los horarios. Así como también facilitará el uso de los mismos al turismo, ya que de un simple vistazo se podrá saber, por ejemplo, que autobús nos puede llevar cerca de nuestro destino, y cuanto falta para que este llegue a la parada.

Asimismo, los horarios ofrecidos actualmente por las compañías de autobús, generalmente, son difíciles de comprender y ofrecen poco margen en materia de información acerca de retrasos o adelantos por parte del medio de transporte. Es decir; en caso de que un autobús

tenga una hora de llegada aproximada de 20 minutos, es posible que dicho tiempo se retrase por mal tráfico, tiempo lluvioso o un accidente en la carretera, entre otros, dejándonos sin la valiosa y útil información que indique que el autobús se retrasa.Es por ello, que utilizando el servicio podremos observar que el autobús no esta avanzando y que por tanto, no llegará en los 20 minutos que estaban planificados.

Por último, dado que el sistema originalmente está basado en el territorio histórico del País Vasco, el nombre elegido para el sistema resultante ha sido *euskadiMoving*, cuyo equivalente en español sería *Euskadi en movimiento*. Para el acceso al sistema se ha adquirido el dominio *www.euskadimoving.com*, que estará accesible durante el tiempo de desarrollo del proyecto y su acceso se garantizará hasta finales del año 2013.

1.1. Objetivos del proyecto

El objetivo principal del proyecto es el de diseñar y desarrollar un sistema de seguimiento y posicionamiento en tiempo real para el transporte público. Este sistema estará destinado a la monitorización del lugar en el cual se encuentran los vehículos en cada momento y representar dicha información a través de un mapa, que a su vez muestre la línea que sigue, o debe seguir, dicho transporte.

El sistema será desarrollado siguiendo el concepto de distribución SaaS, es decir *Software como servicio* (del inglés: Software as a Service[1]). Esta aproximación, coloca al cliente como mero consumidor de un servicio para el cual únicamente necesita un medio que pueda acceder a el, como podría ser un ordenador o un smartphone con acceso a internet. Por tanto, la lógica del sistema se encontrará en el lado servidor.

La metodología de desarrollo utilizada ha sido la de XP o *programación extrema* (del inglés: *eXtreme Programming*, XP[2]). Ésta ha sido la metodología elegida debido al tamaño del equipo, que consiste en una única persona y a la necesidad de obtener un producto terminado en un corto periodo de tiempo. Al utilizar esta metodología se relega gran parte de la documentación al código del proyecto en sí, el cual debe ser fácilmente comprensible y ejercer como si fuese documentación.

[1] http://es.wikipedia.org/wiki/Software_como_servicio
[2] http://es.wikipedia.org/wiki/Programacion_extrema

1.2. Estructura del documento

La memoria del proyecto está estructura del siguiente modo: en el capítulo *Documento de Objetivos del Proyecto* se presentan los objetivos generales del proyecto, junto con la descripción de las tareas correspondientes y su planificación. También se detalla la metodología a utilizar y el análisis de riesgos del proyecto. A continuación, en el capítulo *Desarrollo Técnico* se describen los pasos realizados durante el desarrollo del proyecto. Por último en el capítulo *Resultados, Conclusiones y Líneas Futuras* se evalúan los resultados obtenidos y se extraen las conclusiones del proyecto, incluyendo una breve descripción de las posibles líneas de trabajo futuras.

La memoria incluye dos anexos. En el primero, *Anexo: Prototipo localizador GPS*, se hace una introducción a un posible dispositivo GPS para las compañías de transporte. El segundo anexo, *Anexo: Instalación y configuración del servidor*, se indica cómo realizar la instalación del servidor para el proyecto.

2. CAPÍTULO

Documento de Objetivos del Proyecto

2.1. Introducción

En este apartado se van a detallar aquellos objetivos que engloban el proyecto. No obstante, no todos los objetivos tratados en este apartado están dentro del alcance del proyecto. Asimismo, también se tratará la motivación para el desarrollo del proyecto, un análisis de factibilidad , alternativas y antecedentes del proyecto y por último, la arquitectura planteada para el sistema.

En la sección de Motivación se hará un breve descripción de la razón por la cual se opto por desarrollar este proyecto. Por otro lado, en la sección de objetivos, se detallarán aquellos objetivos que se han ideado para el sistema, tanto los que son abordados por este proyecto final de carrera, como los que se plantean de manera teórica para un sistema de mayor envergadura. En el siguiente apartado, se realizará una análisis de la situación actual de sistemas de similares características o que cumplen un propósito similar y la factibilidad que tendría el sistema a desarrollar en este proyecto. Por último, se hará un análisis de distintas arquitecturas que existen para desarrollar sistemas de esta índole y se hará hincapié en el las razones por las cuales se ha optado por uno de ellos.

2.2. Motivación

En este apartado se detalla la motivación por la que surgió la idea a ser realizada y a través de la cual se decidió llevar a cabo este proyecto.

La ciudad de Londres cuenta con un sistema de *tracking*[1] de algunas de sus líneas de metro. Este sistema fue desarrollado por un concursante del campeonato de desarrollo rápido *Science Hackday* de 2010. Este sistema hace uso de una interfaz de programación de aplicaciones o API es decir del inglés Aplication Programming Interface[2] que facilita la misma compañía de transporte. Esta API permite que el sistema de tracking muestre la ubicación estimada de los vagones de metro en las líneas que había disponibles en 2010.

Por tanto, inspirado en este sistema, se pretende desarrollar un sistema similar, este sistema se desarrollará tomando como ciudad de funcionamiento Donostia-San Sebastián y preparado para funcionar en Bilbao y Vitoria-Gasteiz. No obstante, el sistema permitirá la inclusión de nuevas ciudades, ya que se plantea un sistema que no esté limitado únicamente a las ciudades previamente mencionadas. Esto es así, ya que en la creación del sistema se tendrá en cuenta que pueda ser un sistema para varias ciudades, es decir, el diseño no estará centrado para su funcionamiento en Donostia-San Sebastián.

Por otro lado, si bien en el diseño y desarrollo del sistema se tiene en cuenta la futura anexión de nuevas ciudades, está fuera del alcance del proyecto realizar un análisis de tiempo, costo y esfuerzo que supondría la anexión de nuevas ciudades.

2.3. Objetivos

A diferencia del sistema de Londres, para el cual se cuenta con una API pública facilitada por la administración de transportes, en España a nivel estatal y más concretamente en Euskadi a nivel regional no existe semejante tecnología disponible. No obstante, algunas compañías de transporte urbano cuentan con sistemas de localización de algún tipo, característica indispensable para el proyecto, ya que cada medio de transporte deberá disponer de su localizador, y este a su vez ser capaz de transmitir dicha información a un servidor que procesará los datos.

Debido a la situación que se plantea con respecto a la disparidad en la información de aquellos casos en los que una ciudad ya tuviera un sistema de localización previo, se entiende como un objetivo a tratar el desarrollo de un servidor intermedio que tratará la información y la adaptase al formato que se utilizará en este proyecto. No obstante, el análisis y desarrollo de este servidor, sistema o servicio, está fuera del alcance del proyecto.

Por otro lado, para aquellas compañías que no cuenten con dichos sistemas de localización, el

[1] http://traintimes.org.uk/map/tube/
[2] http://es.wikipedia.org/wiki/Interfaz_de_programacion_de_aplicaciones

proyecto incluye un prototipo teórico de localizador GPS diseñado utilizando la placa Arduino[3] y componentes necesarios como antenas y receptores GPS[4], sistema de posicionamiento global (del inglés: Global Positioning System) y GSM[5], etc. Este localizador únicamente se plantea a modo de orientación y una descripción mas detalla del mismo esta recogido en el *Anexo: Prototipo localizador GPS*.

El diseño y desarrollo final de un sistema de posicionado e información de las posiciones no es, por tanto, un objetivo a ser considerado en el ámbito del proyecto. Estas posiciones en el proyecto serán simuladas a través de un simulador de posiciones recogido en la sección *Anexo: Instalación y configuración del servidor*.

El principal producto resultante del proyecto será una servicio que trate la información recopilada y le de formato de tal manera que pueda ser después accesible para su uso. De esta información se nutrirá un portal web que ofrezca el servicio que muestre la información de posicionamiento de los medios de transporte de las ciudades añadidas en el sistema. Este servicio deberá contar con las siguientes características:

- El nivel de detalle mínimo para que el se desarrollará el proyecto será el de línea de autobús/tren/tranvía.
- Cada línea mostrará aquellos autobuses que estén circulando por ella.
- La información se debe recibir en tiempo real.
- Se habilitará una opción que localice al usuario y muestre en caso de haberlo la información para dicha posición.
- La tasa de refresco tendrá que ser suficiente para mostrar correctamente el movimiento de los medios de transporte
- La interfaz del sistema deberá ser adaptable a distintos tamaños de pantalla, a saber: Pantallas de ordenador, tablets o smartphones.

En este punto cabe destacar que el sistema de Londres, en el cual se basa este proyecto, utiliza un sistema distinto al que se plantea en este proyecto. En el sistema de Londres, aproximadamente cada 2 minutos se adquiere la información de absolutamente todas las posiciones de las líneas. En nuestro caso, ésta información se adquirirá constantemente y únicamente se recibirán los datos de aquellas líneas deseadas. Por tanto, el sistema que se abarca en el proyecto

[3] http://es.wikipedia.org/wiki/Arduino
[4] http://es.wikipedia.org/wiki/Gps
[5] http://es.wikipedia.org/wiki/GSM

cuenta con una granularidad menor por tanto la cantidad de datos que debemos enviar/recibir será menor, no obstante, los requerimientos de información serán más frecuentes.

Debido a que la comunicación en tiempo real requiere una latencia lo mas pequeña posible y dado que el intercambio de información será constante, el tamaño de la información a enviar será lo más reducida posible. El proceso de generación de esta información será del servidor de ubicaciones de cada ciudad, servidor que se detalla más adelante.

Debido a que se desea desarrollar un servicio web accesible y utilizable desde gran variedad de dispositivos, con lo que ello conlleva; varios tamaños de pantalla, distinta manera de mostrar la información, etc, el servicio será desarrollado utilizando formas que sean adaptables al tamaño de la pantalla, así como también se tendrá en cuenta que el servicio muestre una página lo mas parecida posible a través de todos los dispositivos.

Por último, el servicio ofrecido, dará la opción al usuario de seleccionar aquellas líneas que desee visualizar, de forma que no será necesario cargar todas las líneas, obteniendo de esta manera un mayor rendimiento.

2.4. Análisis de factibilidad, estudio de alternativas y antecedentes

En este apartado se podrá encontrar en análisis de factibilidad realizado, así como las distintas alternativas tanto en los detalles del desarrollo y al proyecto en sí.

Como posibles alternativas al proyecto se han estudiado las características de los siguientes servicios:

- El sistema actual de mostrar en las marquesinas el tiempo faltante para la llegada del próximo autobús/tranvía/metro.
- Un servicio similar al de la ciudad de Londres, en el cual se estima la posición del transporte.
- El servicio actual con el que cuenta la empresa municipal de transporte de Donostia dBus.
- El servicio de similares características a la idea de este proyecto, ofrecido por la ciudad de Pamplona.

La primera alternativa que se ha considerado, es en principio la referencia en sí para este proyecto. Este servicio se basa en mostrar en las marquesinas el tiempo faltante para la llegada de los próximos autobuses de las distintas líneas que paran en dicha marquesina.

Si bien es un servicio que está bastante extendido y su utilidad es amplia, tiene grandes inconvenientes entre ellos:

- Falta de precisión, generalmente se basan en heurísticos que no son correctos al 100 % y acumulan aun más fallos en días lluviosos, de muchos tráfico, etc.
- Únicamente se pueden consultar en la misma marquesina, parada de tranvía o parada de metro. Obligándonos a trasladarnos hasta el lugar para poder acceder a la información
- No es accesible desde ninguna otra plataforma que no sea la misma parada.

Por tanto, ya que se entiende este sistema como el referente a través del cual partiría el proyecto, se comprende que sería un complemento al servicio ofrecido por el proyecto ofreciendo de esta manera mas información al usuario final e incluso como línea futura se plantea la fusión de ambos servicios.

Como segunda alternativa, nos encontramos con la posibilidad de ofrecer un servicio similar al ofrecido por la ciudad de Londres para sus líneas de metro. En este caso, el servicio ofertado recibe información real enviado por los vagones cada 2 minutos aproximadamente, durante el resto de tiempo el servicio estima la posición de los vagones. Por contra en este proyecto, se pretende realizar un sistema que informe en tiempo real.

A lo largo de los dos minutos aproximados en los que no se recibe información, se realiza una estimación de la posición en la que se debe encontrar el vagón. Este método es posible e incluso muy fiable tal y como se ha podido constatar tras las observaciones realizadas al funcionamiento del sistema de Londres. No obstante, esta precisión y fiabilidad esta supeditada a que el medio de transporte que utiliza la estimación siga una camino prefijado, en el cual, es difícil que se presenten anomalías que hagan cambiar el ritmo habitual. Por tanto, un metro es ideal para la aplicación de este método, no así una línea de autobús en la cual cada autobús debe enfrentarse a un tráfico distinto según la zona en la que se encuentre, la franja horaria o el tiempo que haga, entre otras variables.

Como tercera alternativa, se presenta la aplicación web móvil facilitada por la empresa municipal dBus que es la explotadora del transporte urbano en San Sebastian. Ésta aplicación, permite conocer la información de las líneas, tiempos de llegada de los autobuses a sus paradas, planificar una ruta y localizar las paradas cercanas a la posición del usuario.

En sí, esta aplicación adapta los servicios ofrecidos por la web a un formato más accesible a través de los teléfonos móviles, por tanto, al igual que la primera alternativa comentada podría utilizarse como referencia o complementaría al proyecto.

Por último, existe la alternativa actualmente ofrecida por la ciudad de Pamplona llamada *TuVillavesa*; en Pamplona se conoce como Villavesa a los autobuses urbanos. Ésta aplicación es la idea mas cercana a la que se plantea por este proyecto. La aplicación ofrece las siguientes características:

- Tiempos de llegada: informaciones en tiempo real sobre las estimaciones de llegada de los autobuses a las paradas
- Sistema de alertas de tiempos de llegada
- Favoritos: selecciona las paradas que utilices de forma habitual para acceder directamente a la información
- Cómo llegar: planifica la ruta óptima para llegar a tu destino viajando en transporte público
- Paradas cercanas: localiza la parada más próxima a tu posición (en lista, en mapa o en Realidad Aumentada)
- Información de las líneas: rutas, paradas, frecuencias...
- Incidencias de servicio

Como se puede observar, en líneas generales esta alternativa cubre los objetivos del proyecto, así como también aborda más objetivos no planteados en el alcance de este proyecto. No obstante, a diferencia de este proyecto, el servicio se ofrece únicamente a través de plataformas móviles, a diferencia de este proyecto el cual pretende crear un servicio web el cual podrá ser accesible a través del portal web recogido en el apartado de objetivos, o como se recoge en el apartado de líneas futuras, por aplicaciones móviles.

2.4.1. Análisis de Factibilidad

Este proyecto supone un avance en lo referente a disponibilidad de horarios de transporte público, no obstante, existen ejemplos en los cuales se puede observar, que no siempre una tecnología nueva y diferente a la ya existente fracasa estrepitosamente. Claro ejemplo de este caso, fue la consola Dreamcast, donde en un mercado que tenía peor competencia fue estala que perdió. Para evitar esta circunstancia, como objetivo fuera del alcance del proyecto habría

que realizar un análisis de los factores fundamentales para el éxito del proyecto, como pueden ser: la plataforma de distribución, la seguridad del sistema, la calidad del servicio, la velocidad de instalación y adaptación de las líneas, la publicidad del servicio, etc.

Por último, como gran baza a favor de la viabilidad del proyecto, contamos con que la mayoría de dispositivo móviles cuentan con acceso a internet, facilitando así a los usuarios tener acceso al servicio desde cualquier lugar, sin la necesidad de depender de un ordenador. Por otro lado, cumpliéndose el propósito de controlar más certeramente la frecuencia del transporte público el uso del mismo aumentará inevitablemente.

2.5. Arquitectura del Sistema

La arquitectura planteada para el sistema, es una arquitectura distribuida, esta contará con un servidor central que contendrá el servicio web y servidores locales. Esta arquitectura la podemos observar en la figura *Arquitectura distribuida del sistema*.

Estos servidores locales recibirán la información de los dispositivos GPS de los vehículos de transporte público de cada ciudad. Asimismo, estos servidores estarán conectados con el servidor central de información. Una vez que un cliente ha solicitado la información sobre una o varias líneas de transporte, el servidor central informará al cliente de la dirección de red del servidor local correspondiente y será este el que facilitará la información actualizada al cliente.

El servidor central, por otro lado, será el encargado de proveer el acceso a el servicio web.

La decisión de crear una infraestructura distribuida de este tipo, en la cual hay uno o varios servidores locales, es debido a que el tiempo de comunicación es un factor de gran importancia. Al enfrentarnos a un sistema que intentará mostrar en tiempo real donde se encuentra cada vehículo, debemos tener por un lado, una gran velocidad a la hora de obtener la posición de los vehículos y por otro lado, gran velocidad a la hora de servir la información.

Se entiende, que generalmente, cuando alguien solicite la información de cierta línea, esa persona este ubicada en la ciudad de la cual se solicita la información y por tanto, si bien queda fuera del alcance del proyecto, se deberá realizar un estudio para saber donde ubicar los servidores de locales.

Aunque la arquitectura planteada para el sistema es esta, el proyecto será desarrollador en un único servidor que simulará la arquitectura planteada en este apartado.

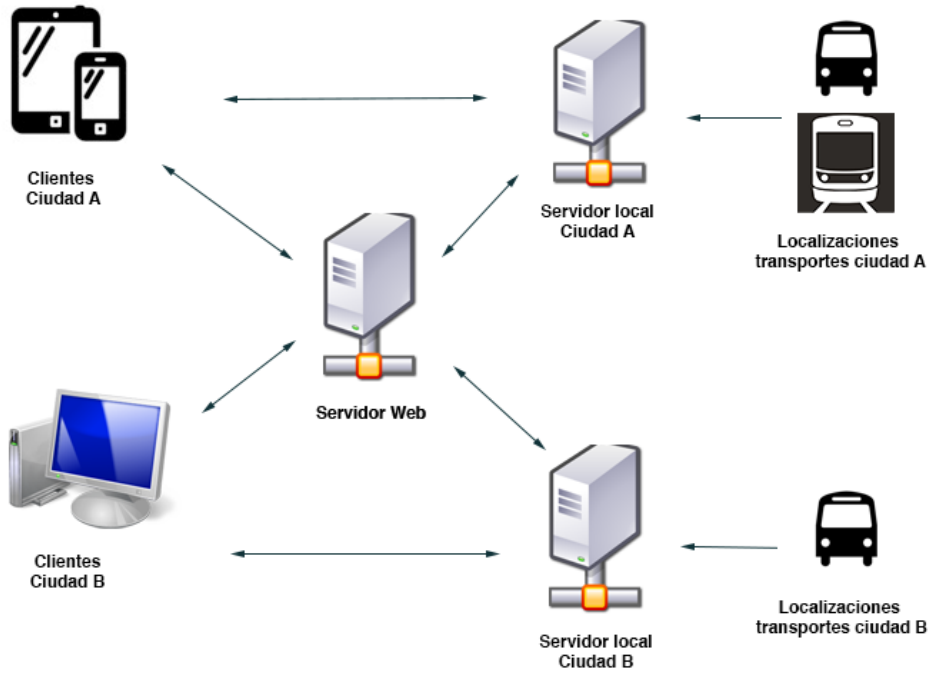

Figura 2.1. Arquitectura distribuida del sistema

2.6. Planificación del Proyecto

En este apartado se va a desarrollar la planificación del proyecto. Contará con lo siguientes sub-apartados: gestión del alcance del proyecto, estructura de desglose del trabajo, planificación, análisis de riesgos, plan de calidad, el plan de comunicación, las adquisiciones y por último, el seguimiento y control del proyecto.

Este documento contiene aquellas partes de una planificación que se han considerado oportunas incluir, dejándose de lado la estimación de costes y el plan de recursos humanos ya que se considera que siendo este un proyecto final de carrera, desarrollado únicamente por una persona, la inclusión de estos documentos no es necesaria.

2.6.1. Gestión del Alcance

Alcance

El proyecto cuenta con apartados diferenciados; el servicio web, el servidor de localizaciones y el dispositivo GPS.

El servicio web deberá contar con las siguientes características:

- Una página principal que mostrará Donosti como ciudad inicial, pero preparada para mostrar cualquier ciudad.
- Sistema de geo-localización para mostrar la información de los transportes urbanos del lugar en el que nos encontremos.
- Líneas de autobús de Donostia y pre-configuración de Vitoria-Gasteiz y Bilbao.
- El sistema cargará las líneas de transporte público cercanas a nosotros si así lo deseamos.
- Página de contacto e información sobre el sistema.

En cuanto al diseño del servicio web:

- Se creará un diseño para abarcar el mayor número de dispositivos.

El servidor de localizaciones:

- Contará con un sistema para recoger las posiciones de los autobuses y almacenarla en un sistema ordenado.
- Proveerá a los clientes que así lo soliciten las posiciones de aquellos medios de transporte solicitados.

El simulador de posiciones:

- Se creará un simulador de las posiciones de los autobuses de Donostia - San Sebastián.

Restricciones y limitaciones

- El servicio web deberá ser correctamente visualizable en los siguientes navegadores (Navegadores compatibles con las hojas de estilo de twitter bootstrap):
  - Firefox 10 y posteriores.
  - Chrome 16 y posteriores.
  - Safari móvil.
  - Chrome móvil.
- El servicio web deberá estar adaptada para poder visualizarse correctamente en los smartphones y tablets.
- Mientras no haya un sistema de GPS accesible para la ciudad de Donostia - San Sebastián, y como medio de prueba, se simularán las posiciones de los autobuses.

2.6.2. Estructura de Descomposición del Trabajo

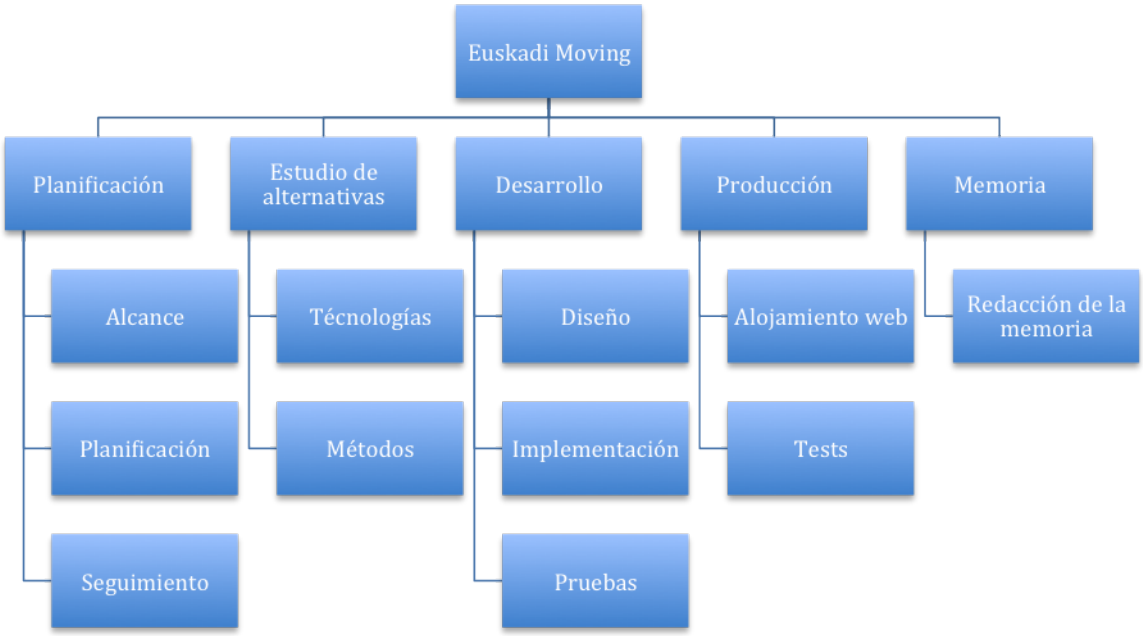

Figura 2.2. Estructura de descomposición del trabajo

Diccionario de la EDT

En este apartado se van a describir cada uno de los apartados recogidos por la Estructura de Descomposición del Trabajo del proyecto, reflejada por la figura *Estructura de descomposición del trabajo*.

Planificación

- Alcance:
  - En esta etapa se realizará un análisis de los requisitos necesarios para el desarrollo de los productos que engloba este proyecto.
- Planificación:
- Esta etapa conlleva la planificación del proyecto, la redacción y revisión del documento de plan de proyecto.

Estudio de alternativas

- Tecnologías:
    - En esta etapa se analizarán las distintas tecnologías disponibles que se pueden utilizar para el desarrollo y se elegirán aquellas que se consideren mejores para el proyecto. Asimismo, aquí se incluye la formación necesaria en dichas tecnologías.
- Métodos:
- En esta etapa se considerarán las distintas formás de solucionar los problemás y los retos de ingeniería y se buscarán para ellos las mejores soluciones.

Desarrollo

- Diseño:
    - Aquí se realizará el diseño del sistema.
- Implementación:
    - Codificación de lo diseñado previamente.
- Pruebas:
    - Por último, realización de las pruebas necesarias para que funcione el sistema de manera correcta.

Producción

- Alojamiento:
    - Puesta en punto de los servidores del sistema y configuración de los mismos para que funcionen correctamente.
- Tests:
    - Pruebas de funcionamiento del sistema una vez instalado todo.

Memoria

- Redacción de la memoria:
    - Redacción, revisión y si fuera necesario, corrección de la memoria del proyecto.

2.6.3. Planificación

En este apartado se detalla el tiempo disponible para el desarrollo del proyecto.

El proyecto cuenta con un tiempo de desarrollo que comprende desde principios de octubre, fecha de asignación del proyecto, hasta el 1 de abril, fecha límite. A lo largo de los meses de octubre, noviembre y diciembre, las horas que se dedicarán al proyecto serán las de las tardes (16:30h-20:30h, 4 horas por día), ya que por las mañanas se deberá acudir a las clases en las que el desarrollador del proyecto está matriculado. En el mes de Enero se entenderá que habrá un periodo de inactividad debido a los exámenes.

Tras los exámenes, el tiempo que se podrá dedicar al proyecto será de 8 horas diarias, hasta la fecha límite previamente dicha. Por tanto, las horas dedicadas al proyecto son aproximadamente las recogidas por la tabla *Horas Proyecto*. Se ha tomado como referencia que un mes cuenta con 20 días laborales. En el primer periodo se considerarán 4 horas útiles por día y en el segundo 8 horas.

Tabla 2.1. Horas Proyecto

Periodo	Horas por día	Días	Suma de horas
Octubre-Diciembre	4 horas	60 días	240 horas
Febrero-Abril	8 horas	40 días	320 horas
Total			560 horas

Por tanto, el resultado total de horas que se puedan utilizar para llevar a cabo el proyecto será de unas 560 aproximadamente. No obstante, habrá que tener en cuenta ciertas desviaciones, habrá días que no se puedan dedicar al desarrollo del proyecto por diversas causas, así como también habrá otros días no planificados como días desarrollo, que se utilizarán como tal.

A lo largo de este periodo de tiempo, se formará en las tecnologías decididas se desarrollarán las tareas listadas más adelante, se redactará la memoria, y se llevarán a cabo las reuniones con los tutores del proyecto. Las actividades de desarrollo del proyecto serán las siguientes:

- Diseño, desarrollo y pruebas del front-end del servico web y la base de datos del sistema.
- Adquisición e incorporación al sistema de recursos e información.
- Diseño, desarrollo y pruebas de los servidores de ubicaciones.
- Diseño, desarrollo y pruebas del back-end del servicio.
- Alojamiento y puesta en marcha del servicio.

2.6.4. Gestión de tareas

En este apartado se detallarán las tareas en las que se ha descompuesto el proyecto. Cada actividad del proyecto esta descompuesta por sus respectivas tareas

Actividad 0: Formación.

Esta actividad engloba todas las tareas de formación que se han llevado a cabo en el proyecto.

- **Formación framework Django**
    - Descripción: Adquisición de conocimiento del funcionamiento del framework django, tanto en el apartado técnico de funcionamiento como del uso de las *templates* de diseño.
    - Recursos:
        - Manual del framework: django book recogido en la bibliografía
        - Documentación oficial: https://docs.djangoproject.com/en/1.5/
- **Formación CSS3 y HTML5**
    - Descripción: Adquisición de conocimiento sobre funciamiento de HTML5 y hojas de estilo de twitter bootstrap. En esta tarea se analizarán las etiquetas de
    - Recursos:
        - Página web de Twitter Bootstrap: http://twitter.github.io/bootstrap/
        - Documentación oficial HTML5: http://dev.w3.org/html5/html-author/
- **Formación servidor**
    - Descripción: Formación en aspectos de administración de servidores.
    - Recursos:
        - Servidor HP MicroServer
- **Formación tecnológica**
    - Descripción: Formación en diversos los lenguajes de programación Javascript y JQuery.

- Recursos:
  - Manuales online

- **Formación BBDD**
  - Descripción: Formación en las peculiaridades del sistema gestor de base de datos PostgreSQL.
  - Recursos:
    - Manuales online

- **Formación Sphinx**
  - Descripción: Formación en el lenguaje para creación de documentos Sphinx.
  - Recursos:
    - Manuales online

En la tabla *Tiempo tareas formación* podemos observar las horas que se han dedicado a las tareas de formación.

Tabla 2.2. Tiempo tareas formación

Tarea	Horas
Framework Django	30 horas
HTML5 y CSS3	10 horas
Servidor	10 horas
JS y JQuery	15 horas
PostgreSQL	5 horas

Actividad 1: Diseño, desarrollo y pruebas del front-end del servicio web y la base de datos del sistema.

En esta actividad se llevarán a cabo las tareas que tengan que ver con el desarrollo de la parte web del proyecto y la base de datos.

En la tabla *Tiempo tareas* podemos observar las horas que se han dedicado a las tareas de la actividad 1.

- **Diseño web**

- Descripción: Creación de las *templates* de Django necesarias para el diseño web. Estas plantillas harán uso de HTML5 y las horas de estilo que provee Twitter Bootstrap. El diseño deberá seguir un layout adaptable a los distintos tamaños de pantallas de los clientes.
- Recursos:
  - Hojas de Estilo de Twitter
  - Navegadores Chrome y Firefox

- **Desarrollo web**
  - Descripción: Desarrollo del código necesario para el funcionamiento de la web. Este desarrollo incluye el proceso de actualización de la información enviada facilitada por los servidores de ubicación. Así como toda las opciones de localización del servicio.
  - Recursos:
    - Google maps
    - JQuery, Javascript
    - Navegadores Chrome y Firefox
- **Desarrollo BBDD**
  - Descripción: Desarrollo de la base de datos del sistema.
  - Recursos:
    - Manual PostgreSQL
- **Pruebas de funcionamiento**
  - Descripción: Realización de pruebas de funcionamiento de la parte web del proyecto.
  - Recursos:
    - Portal web

Tabla 2.3. Tiempo tareas

Tarea	Horas
Diseño web	30
Desarrollo web	80
Desarrollo BBDD	5
Pruebas web	30

Actividad 2: Adquisición e incorporación al sistema de recursos e información.

En esta actividad se engloban todas las tareas que se han llevado a cabo para la adquisición de información, datos y útiles necesarios para el proyecto

- **Adquisición de datos**
    - Descripción: Adquisición de datos de las líneas actuales de San-Sebastian, utilizando para ello la información de la página web de la compañía dBus y sacando las coordenadas
    - Recursos:
        - Pagína web dBus
        - Google Earth
- **Modificación de datos**
    - Descripción: Modificar los datos obtenidos en la tarea anterior para que los datos sean utilizables por el sistema.
    - Recursos:
        - Datos tarea *Adquisición de datos*
- **Adquisición de dominio web**
    - Descripción: Adquisición del dominio www.euskadimoving.com, dominio que se utilizará para entrar al servicio.

En la tabla *Tiempo tareas formación* podemos observar las horas que se han dedicado a las tareas de adquisición.

Tabla 2.4. Tiempo tareas formación

Tarea	Horas
Adquisición de datos	30
Modificación de datos	5
Adquisición de dominio	1

Actividad 3: Diseño, desarrollo y pruebas de los servidores de ubicaciones.

- **Desarrollo de simulador de posiciones**
    - Descripción: Desarrollo del simulador de posiciones de autobuses.
    - Recursos:
        - Datos tarea *Adquisición de datos*
- **Pruebas simulador de posiciones**
    - Descripción: Pruebas del simulador de posiciones realizado en la tarea anterior.
    - Recursos:
        - Simulador de posiciones
- **Creación de un modelo teórico de posicionador GPS**
    - Descripción: Desarrollo teórico de un posicionador GPS basado en tecnología Arduino. Para ello se basará en información acerca de la placa Arduino y sus funcionalidad, así como en añadidos que se le puedan hacer a esta para conseguir el objetivo deseado.
    - Recursos:
        - Esquemas Arduino

En la tabla *Tiempo tareas servidor ubicaciones* podemos observar las horas que se han dedicado a las tareas correspondientes a los servidores de ubicaciones.

Tabla 2.5. Tiempo tareas servidor ubicaciones

Tarea	Horas
Desarrollo de simulador de posiciones	16
Pruebas simulador de posiciones	4
Creación de un modelo teórico de posicionador GPS	10

Actividad 4: Diseño, desarrollo y pruebas del back-end del servicio.

En esta actividad se presentan las tareas que se deben realizar para el servidor back-end.

- **Instalación y configuración de las tecnologías del servidor**
    - Descripción: Instalación y configuración del servidor back-end. Esta tarea está descrita en profundidad en el *Anexo: Instalación y configuración del servidor*.
    - Recursos:
        - Servidor
        - Acceso a los repositorios de las tecnologías
- **Puesta en marcha y configuración**
    - Descripción: Puesta en marcha del servidor y configuración de seguridad del mismo. Esta tarea está descrita en profundidad en el *Anexo: Instalación y configuración del servidor*.
    - Recursos:
        - Servidor
- **Pruebas de funcionamiento**
    - Descripción: Pruebas de funcionamiento del servidor.
    - Recursos:
        - Servidor

En la tabla *Tiempo tareas servidor back-end* podemos observar las horas que se han dedicado a las tareas correspondientes al back-end.

Tabla 2.6. Tiempo tareas servidor back-end

Tarea	Horas
Instalación y configuración de las tecnologías del servidor	16
Puesta en marcha y configuración	20
Pruebas del servidor	16

Actividad 5: Alojamiento y puesta en marcha del servicio.

En este apartado se detallan las tareas para la puesta en marcha del servicio.

- **Despliegue del sistema**
    - Descripción: Desplegar el sistema en el servidor final.
    - Recursos:
        - Proyecto realizado
        - Servidor
- **Configuración del servidor**
    - Descripción: Configuración del servidor para funcionar con el proyecto.
    - Recursos:
        - Proyecto realizado
        - Servidor
- **Pruebas de funcionamiento del servidor**
    - Descripción: Una vez puesto en marcha el servidor se harán pruebas de funcionamiento y de carga.
    - Recursos:
        - Proyecto desplegado
        - Servidor
        - Navegador web compatible

En la tabla *Tiempo tareas alojamiento* podemos observar las horas que se han dedicado a las tareas correspondientes al alojamiento del servicio.

Tabla 2.7. Tiempo tareas alojamiento

Tarea	Horas
Despliegue del sistema	4
Configuración del servidor	24
Pruebas de funcionamiento	16

Actividad Final: Redacción de la memoria

Como última actividad se considera la redacción de la memoria, con sus respectivas tareas.

- **Redacción**
    - Descripción: redacción de una primera versión de la memoria.
    - Recursos:
        - Proyecto desplegado
- **Corrección**
    - Descripción: Corrección de la memoria en base a los comentarios de los tutores del proyecto.
    - Recursos:
        - Proyecto desplegado

En la tabla *Tiempo tareas memoria* podemos observar las horas que se han dedicado a las tareas correspondientes a la creación de la memoria.

Tabla 2.8. Tiempo tareas memoria

Tarea	Horas
Redacción memoria	30
Corrección memoria	30

2.6.5. Plan de calidad

Se han asignado rigurosos requisitos de calidad para asegurar que el servicio que se va a realizar y la web cumplen con el nivel de excelencia exigida por el proyecto. Debido a la intensa interacción que se hará de éste servicio tanto por dispositivos móviles como ordenadores, este tiene que ser muy intuitivo y amigable, es por ello que se ha puesto un especial énfasis en este aspecto. Es tal el énfasis, que el sistema resultante será tan intuitivo que no necesitará ningún manual de uso para el usuario final.

En lo referente al código resultante, ya que se ha seguido una metodología de desarrollo XP, el código en sí será la documentación del mismo. Para ello, todo el código está altamente comentado y explicada su funcionalidad en todo momento.

Por último, se realizarán pruebas de todo el sistema para comprobar que funciona todo correctamente en diversas situaciones de estrés para el sistema.

Aseguramiento de la calidad

Para asegurar la calidad que se intenta conseguir, tanto el estudiante como los tutores del proyecto prestarán gran atención a los aspectos mencionados previamente. Asimismo, en caso de ser posible, se adquirirá un grupo de beta testers para probar el servicio y hacer pruebas de carga y velocidad del servicio.

2.6.6. Plan Comunicación

Se ha realizado un análisis de los posibles interesados, a partir del cual se ha obtenido la lista que se detalla a continuación.

- Alumno y tutores (Equipo)
    - Xabier Valdecantos
    - Roberto Cortiñas
    - Mikel Larrea
- Clientes
    - Ayuntamientos de las ciudades, Gobierno Vasco (en el caso del País Vasco) y Ministerio de Fomento (en el caso del Gobierno de España)
    - Usuarios de la aplicación
- Proveedores de servicios
    - Compañías públicas de transporte
    - Google
    - Twitter
    - GoDaddy

Gestión de las comunicaciones

Equipo

De entre los interesados, se mantendrá una constante comunicación entre miembros del equipo. Estas comunicaciones se llevarán a cabo cada vez que se dé algún hecho de cierta importancia. Para ello, se han establecido varias vías de comunicación, que se detallarán a continuación:

- Email - Comunicaciones oficiales, dudas y concertar citas.
- Bitbucket - Servidor de versiones.

Como medio oficial de comunicación y así permitiendo dejar registro de las comunicaciones, se utilizará el Email. Por otro lado, para intercambiar imágenes o ficheros en general, que pudieran ser útiles en el desarrollo del proyecto, se hará uso del sistema de intercambio de ficheros Dropbox. Por último, para realizar control de versiones del código, se utilizará git sobre el servidor de versiones bitbucket.

Clientes

De entre los clientes, se hará una clara diferenciación entre los usuarios finales y aquellas personas físicas o jurídicas que vayan a adquirir el servicio que se ofrecerá con el proyecto (a partir de ahora contratantes). En el caso de los contratantes, se ofrecerá un servicio rápido de comunicación tanto por email, como por teléfono móvil personal del equipo, facilitado a tal propósito directamente con cada contratante.

Por otro lado, los usuarios finales contarán con la dirección de email oficial del servicio y la cuenta oficial del servicio en Twitter para ponerse en contacto con el equipo. Dado el caso de una expansión territorial del servicio, se estudiarán nuevas formas de comunicación con los usuarios.

Proveedores de Servicios

Con los proveedores de servicios se establecerán los medios de comunicación prefijados mediante los contratos de adquisición, así como las intranet que ofrezcan (en caso de ofrecer) y los servicios de atención al usuario.

2.6.7. Distribución de las comunicaciones

Una vez el servicio esté listo para ser lanzado, sería aconsejable realizar una campaña publicitaria en las marquesinas de los distintos medios de transporte público, para de esta manera darse a conocer entre los usuarios del servicio. De esta manera se facilitará una eventual expansión del servicio a través del boca a boca de los usuarios.

Por otro lado, se concertarán citas con los posibles clientes que pudieran estar interesados en la adquisición del servicio, haciendo demostraciones del mismo, para de esta manera mostrar las ventajas que ofrece con respecto a la tecnología existente.

2.6.8. Análisis de Riesgos

Una vez identificados los riesgos del proyecto se procederá a priorizar los mismos para poder realizar un plan frente a cada uno de los mismos. Los riesgos identificados y ordenados por prioridad han sido los siguientes:

Cumplimiento de plazos

Este riesgo se cataloga como el más importante de todos ya que la fecha marcada como fecha final, 1 de abril de 2013, es la fecha límite que tiene el estudiante para terminar el proyecto, debido a que comienza a trabajar la siguiente semana.

Dada la fecha límite, se realizará una planificación que permita tener una holgura de manera que las actividades puedan desarrollarse antes de las fechas límites. En caso de que durante el seguimiento del proyecto se viera que es inevitable el retraso del proyecto, el desarrollo restante del mismo será llevado a la par con el trabajo, esta situación llevará a un retraso considerable del proyecto.

Desconocimiento de las tecnologías a utilizar

El no tener experiencia previa a la hora de desarrollar el producto con las tecnologías decididas, puede generar que las estimaciones iniciales no sean buenas y que esto lleve a un fracaso del proyecto.

Para hacer frente a este riesgo, la etapa de formación ha sido muy intensa, así como el esfuerzo dedicado a buscar manuales e información de calidad.

Correcto funcionamiento de tecnologías y servicios suministrados por terceros

El servicio resultante depende en muchas de sus funcionalidades de las tecnologías suministradas por terceros, lo que tiene un riesgo muy grande a la hora de garantizar el correcto funcionamiento del servicio en caso de fallar estos.

Correcto funcionamiento del servicio en smartphone y tablets

Si bien para el diseño del sistema, se tendrá en cuenta que se pueda ver correctamente en estos dispositivos, el servicio contará con una gran carga de computación de datos, por lo que, es posible que algunos dispositivos poco potentes no sean capaces de utilizar el servicio de manera fluida y por tanto óptima.

Para solucionar esta situación, en un futuro, si el servicio cuenta con éxito y el respaldo de las instituciones, el desarrollo de aplicaciones nativas para los dispositivos será una prioridad, tal y como se recoge en la sección *Líneas futuras*.

Entrada a un segmento con competencia

El servicio que se está desarrollando, si bien no tiene una competencia directa, cuenta con una competencia indirecta ofrecida por los marcadores de tiempo restante que se pueden encontrar en marquesinas y paradas. Este servicio, lleva más tiempo y esta bastante arraigado, por tanto la lucha contra el mismo en principio podrá ser bastante difícil y habrá que tenerlo muy en cuenta.

No obstante, para superponerse a la competencia, el servicio que se está desarrollando ofrece una mayor información sobre el transporte, así como la posibilidad de crecer en servicios ofrecidos.

Por otro lado, debido al sistema de plantillas ofrecido por Django, se habilita de manera sencilla la traducción del servicio a varios idiomas.

2.6.9. Plan de Adquisiciones

La principal adquisición para este proyecto es la tecnología de mapas a usar. Existen varias alternativas a elegir (Google Maps, Apple Maps, Ovi Maps, Bing Maps, OpenStreet Maps...), cada uno de ellos cuenta con sus puntos fuertes y débiles. Finalmente, la elección para el

producto web ha sido Google Maps, debido a que es el servicio más extendido y el cual cuenta con una ganada reputación, Asimismo, el conocimiento de la API del mismo es mayor. Si en un futuro se abordan las aplicaciones móviles, se deberá estudiar de nuevo qué tecnología usar en cada caso, ya que las restricciones que se aplican son mayores.

El uso de los mapas de Google se realizará sujeto a las condiciones de uso de dicho servicio. Estas condiciones de uso se encuentran recogidas en la página de Google Maps[6].

Por otro lado, la tabla *Adquisiciones del Proyecto* muestra el resto de adquisiciones realizadas para el proyecto.

Tabla 2.9. Adquisiciones del Proyecto

Descripción	**Detalle Producto**	**Vendedor**	**Precio**	**Nota**
Licencia de uso	Google Maps	Google Inc.	Consultar condiciones de uso	
Servidor de Ubicación	Servidor	UPV/EHU	Consultar condiciones de uso	Pruebas finales
Servidor Web	Servidor	Servidor propio	0€	
Datos GPS	Coordenadas GPS	Compañías públicas de transporte	Consultar condiciones	
Dominio	dominio .com	GoDaddy	10€	

De entre las adquisiciones realizadas, las coordenadas GPS serán simuladas durante el desarrollo del proyecto ya que no existe ninguna API pública y no se dispone de ningún acuerdo de colaboración con ninguna empresa de transporte público. No obstante, estas están recogidas en el proyecto ya que se contempla un futuro con ellas incluidas en el funcionamiento del sistema.

Futuras adquisiciones

En caso de éxito del proyecto, como previamente se ha comentado, se crearán aplicaciones nativas para smartphones y tablets. Estos dispositivos, cuentan con sus propias plataformas de distribución de aplicaciones, las cuales cuentan con una cuota de acceso que habría que pagar.

[6] http://en.wikipedia.org/wiki/Responsive_web_design

Asimismo, sería necesaria la adquisición de un servicio publicitario para dar a conocer el servicio.

Por último, conforme más poblaciones se unieran al servicio, habría que gestionar el acceso a sus datos de posiciones GPS.

Metodología y Herramientas a utilizar

Como lenguaje de desarrollo de lado servidor, se ha utilizado Python y más concretamente el framework *Django*[7]. Mediante el uso de este framework, se ha podido desarrollar a gran velocidad obteniendo una gran calidad en el producto obtenido. El desarrollo con Django sigue la idea de *No te repitas* (DRY, del inglés *Don't Repeat Yourself*), de forma que la re-utilización de código es constante en todo el desarrollo, permitiendo más velocidad y la gran escalabilidad que se ha comentado en el apartado anterior. Asimismo, se hace uso de servidores Apache sobre Linux para tener en servicio el sistema.

Se ha elegido XP (programación extrema) como metodología de desarrollo debido a que el proyecto debe ser realizado en un periodo corto de tiempo, para ello es ideal una metodología ágil en la cual se puede empezar a trabajar desde primer momento. Por otro lado, se entiende que los requisitos del proyecto pueden modificarse a lo largo del ciclo de desarrollo, ofreciéndonos para ello un gran adaptabilidad al no tener que enfrentarnos a engorrosos cambios de requisitos definidos desde el inicio del proyecto. Asimismo, este enfoque libera en gran medida la documentación a ser realizada, siendo esta documentación derivada al código del proyecto en sí.

El servicio web hace uso del futuro estándar HTML5 y utiliza los CSS que facilita twitter llamados *bootstrap*[8]. Estos CSS a parte de ofrecer gran variedad de opciones y contar con un gran número de funciones útiles, permiten hacer un diseño como el indicado en el apartado anterior. Para realizar el control de versión, se ha utilizado el cliente *git* con el servidor de versiones *bitbucket*.

Se ha desarrollado el sistema utilizando los gestores de bases de datos *SQLite* y *PostGreSQL*. Estos sistemas gestores son completamente compatibles con Django y es este último el encargado de gestionar las bases de datos.

Para crear la memoria del proyecto, se ha utilizado el generador de documentación Sphinx[9] que convierte ficheros en formato *ReStructured Test* a una gran variedad de formatos como

[7] http://es.wikipedia.org/wiki/Django_(framework)
[8] http://twitter.github.com/bootstrap/
[9] http://sphinxsearch.com/

PDF usando LaTeX para ello, ePub o un sitio HTML entre otros. El utilizar este sistema para desarrollar la memoria nos permite abstraernos del formato del texto, concentrándonos únicamente en la redacción del mismo, siendo después Sphinx, el encargado de generar el formato para cada una de las opciones de salida.

3. CAPÍTULO

Desarrollo Técnico

A continuación, se describirán los detalles concernientes al desarrollo del sistema planteado por el proyecto, así como las alternativas tecnológicas planteadas. Como previamente se ha comentado, el desarrollo del proyecto está dividido en cinco tareas, dos de las cuales se desarrollan en paralelo, más concretamente las tareas tres, cuatro. El resto de tareas se desarrollarán secuencialmente, siendo requisito indispensable el haber terminado la tarea anterior para afrontar la siguiente.

3.1. Alternativas Tecnológicas

Como tecnologías de lado servidor se plantearon el uso de las siguientes tecnologías:

- PHP
- Java - Framework Struts
- Python - Framework Django

Por un lado, tenemos dos alternativas usando frameworks de desarrollo web, los cuales facilitan en gran medida el desarrollo web, y por otro lado, está la alternativa de utilizar PHP como lenguaje de desarrollo. De entre estas posibilidades, el proyecto inicialmente iba a ser desarrollado utilizando PHP, no obstante, tras estudiar los frameworks se opto por el uso de uno de ellos. Concretamente la elección final fue la de Django, debido a la gran potencia del mismo, así como a la capa extra de simplicidad que ofrece a la hora de diseñar y desarrollar.

Por otro lado, el hecho de no haber desarrollado desde un inicio aplicaciones para dispositivos iOS, Android o Windows Phone entre otros, se debe a que únicamente una persona se ha encargado del desarrollo del sistema y el tiempo para el desarrollo era limitado. Es por ello que desarrollar una página web HTML5 con CSS3 y javascript ha sido la decisión adaptada, ya que de esta manera se posibilita que cualquier dispositivo de los mencionados anteriormente sea capaz de acceder al servicio, si bien la fluidez del servicio no será la misma que la de una aplicación nativa.

Para el desarrollo del servicio se han planteado varias arquitecturas para el sistema:

- Uno o varios servidores centrales en los cuales se cuenta con todo el servicio: Recopilación de información por parte de los transportes, servicio de información de las ubicaciones, portal web...
- Servidores distribuidos, compuesto por:
  - Uno o varios servidores centrales encargados del portal web
  - Uno o varios servidores encargados del tratado de los datos y posterior formateo para su uso.

Por un lado, la mayor ventaja que ofrece utilizar una arquitectura de un nivel, es la facilidad de uso ya que contamos todas las partes accesibles en el mismo punto, así mismo su gestión y administración son mas sencillas por el mismo motivo. No obstante, le ventaja que ofrece tener todo el sistema en un mismo punto, también es su mayor desventaja ya que si falla el servidor fallaría todo el servicio. Así mismo, en este caso el servidor o servidores centrales son los encargados de la gestión de todo, creando de esta manera una gran carga de trabajo.

Por otro lado, las arquitecturas distribuidas ofrecen un nivel de carga menor ya que separa hacia fuera el reparto de la carga, habilitando de ésta manera que los servidores tengan que manejar menos datos. En contra de esta arquitectura se encuentra la mayor dificultad de gestión del sistema a nivel global debido a que cada servicio es ofrecido por una o varias maquinas distintas.

Con todo, la arquitectura elegida para el proyecto es una arquitectura distribuida, la cual es detalla en el punto *Arquitectura del Sistema*.

3.2. Tarea 0: Formación

La tarea de formación ha sido probablemente la tarea mas importante del proyecto y a la que se le ha prestado una gran atención, siendo la única tarea que realmente no termina durante

todo el proyecto.

Debido al desconocimiento por parte del desarrollador del proyecto del lenguaje de programación python y del framework Django, se le ha dado máxima prioridad a la formación en estos dos aspectos. Para ello, se han utilizado los manuales *python para todos* y el libro *Django book*, libros que se detallan en la bibliografía. Para la lectura, comprensión y realización de pruebas, se han utilizado dos semanas aproximadamente.

Así mismo, debido a que el sistema se ha desarrollado usando los CSS bootstrap de Twitter y HTML5, se ha profundizado bastante en las peculiaridades de ambas tecnologías, indagando más aún en las posibilidades de diseño relacionadas con los CSS, para ello se han visitado ejemplos de páginas desarrolladas con esta tecnología, para así poder conocer mas profundamente las posibilidades que ofrecen los mismos.

Por otro lado, también ha sido necesario un mayor uso de javascript y jQuery del que se conocía y para mejorar el conocimiento sobre ello y debido a las características de estos dos lenguajes, se dedico un día a repasar los conceptos básicos de ambos lenguajes y tras esto, el mero hecho de desarrollar utilizándolos ha sido suficiente para aumentar el nivel de conocimiento sobre ellos.

Por último, también ha sido necesario el formarse en el uso de la API v3 de Google maps, así como las peculiaridades que surgen al utilizarla en conjunto con los CSS de Twitter y javascript. Para ello, se ha utilizado la documentación suministrada por Google, así como también sus foros de consulta.

3.3. Tarea 1: Front-end y BBDD

En esta tarea se ha llevado a cabo el desarrollo del front-end y la base de datos de la página web. Debido a que estamos utilizando Django como framework, este paso irremediablemente lo debemos realizar a la par. Aún pudiendo parecer un movimiento algo extraño viniendo del desarrollo tradicional web, una vez nos acostumbramos al funcionamiento de Django, se puede ver que es un paso bastante sensato a dar.

Debemos tener en cuenta que Django utiliza el patrón MVC o Modelo Vista Controlador[1], por tanto se separan la lógica de la página, los datos y la capa de presentación. Al crear un proyecto con Django, deberemos especificar una serie de parámetros, entre ellos qué base de datos vamos a utilizar, en nuestro caso PostgresSQL.

[1] http://es.wikipedia.org/wiki/Modelo_Vista_Controlador

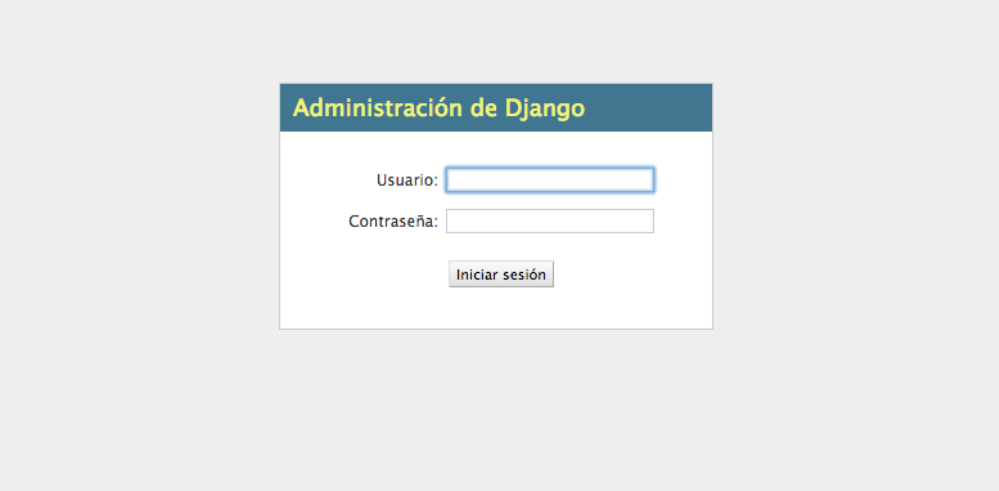

Figura 3.1. Página de inicio de sesión en administración del sistema

A la hora de crear la base de datos, Django ofrece la posibilidad de crear automáticamente la intranet del sistema, como se puede apreciar el resultado en la figura *Página de inicio de sesión en administración del sistema*. Al crear la página de administración, Django automáticamente creará las tablas necesarias para que esta funcione. Esta página automáticamente nos creará grupos, usuarios y los respectivos permisos. También nos permitirá administrarlos, así como también podremos manejar la información de todas las tablas que incluyamos en el proyecto.

Tras esto, al sistema únicamente habrá que facilitarle un fichero escrito en python, llamado models.py. Este fichero, tal y como indica su nombre, representa la capa del modelo de datos. En este fichero habrá que especificar el formato de la base de datos, para ello se utilizará la sintaxis de python, que después Django se encargará de traducir al SQL especifico para nuestra base de datos, en nuestro caso Postgres. El código utilizado para este proyecto lo podemos observar a continuación:

```
from Django.db import models
from Django.utils.translation import ugettext_lazy as _
class ciudad(models.Model):
        nombre = models.CharField(max_length=100)
        latitud = models.CharField(max_length=100)
        longitud = models.CharField(max_length=100)
        def __unicode__(self):
                return self.nombre
        class Meta:
                verbose_name= _('Ciudad');
                verbose_name_plural= _('Ciudades');
```

```
class tipo_transporte(models.Model):
        tipo = models.CharField(max_length=50)
        def __unicode__(self):
                return self.tipo
        class Meta:
                verbose_name_plural= _('Tipos de transporte');
class linea(models.Model):
        nombre = models.CharField(max_length=100)
        identificador = models.CharField(max_length=3)
        activa = models.BooleanField()
        tipo_transporte = models.ForeignKey(tipo_transporte)
        ciudad = models.ForeignKey(ciudad)
        class Meta:
                verbose_name= _('Linea');
                verbose_name_plural= _('Lineas');
```

Este código tras ser ejecutado en Django, creará la base de datos relacional deseada incluyendo todas las restricciones, claves y demás especificaciones indicadas. Una vez realizado este paso, si nos introducimos en la intranet del sistema, podemos ver como están disponibles las tablas que acabamos de introducir, podemos ver esto en la figura *Página principal de administración*. Con esto ya tenemos terminada la parte del modelo de datos.

Figura 3.2. Página principal de administración

Para realizar el diseño de la página web, Django cuenta con un sistema de plantillas que nos facilita la reutilización de código. Para ello, contamos con la posibilidad de la herencia y extensión de las mismas, un sistema muy similar al de la programación orientada a objetos. En nuestro caso, se ha creado una plantilla maestra que incluye la barra de navegación y el pie

de página. De esta plantilla maestra colgarán el resto de páginas de manera que únicamente cambiarán el contenido central de ellas.

3.3.1. Barra de navegación

Debido a que el diseño de la página es uno de los puntos mas importante del proyecto, se ha decidido optar por un diseño sencillo, intuitivo y que siga la tendencia actual de las páginas web. Por ello, se ha incluido una cabecera que contendrá en el lado izquierdo: el nombre del proyecto que hará de ancla al inicio de la página y los enlaces al resto de pantallas de la página. En el lado derecho, estarán los botones que traducirán el servicio a los distintos idiomas disponibles (función no integrada en el sistema) y enlaces a las redes sociales.

Figura 3.3. Barra de Navegación

El resultado lo podemos ver en la figura *Barra de Navegación*. En todo caso, aparecerá resaltado aquel idioma que estemos visualizando, en este caso únicamente podremos seleccionar el Español, el resto de idiomas mostrarán un mensaje informativo indicando que la traducción no esta realizada.

3.3.2. Pantalla principal

Por otro lado, el "cuerpo" de la página será lo que variará entre pantallas. Originalmente están planteadas las pantallas Home, About (Acerca de) y Contacto. Estas pantallas podrán ser ampliadas, modificadas, o se podrán añadir más sin ningún problema y sin tener que hacer grandes cambios, gracias al sistema de plantillas y el modelo MVC.

En la figura *Pantalla Home*, podemos observar el resultado de la pantalla principal, con Donostia - San Sebastián como ciudad a mostrar. Esta pantalla mostrará dos elementos: a la izquierda, ocupando aproximadamente el 70 % de la pantalla, se mostrará el mapa de la ciudad en la cual queremos operar.

A la derecha del mapa nos encontramos con 3 pestañas, a saber: Autobús, Metro y Tranvía. Cada una de estas pestañas cargará las líneas de transporte público específico con las que cuente la ciudad en cuestión.

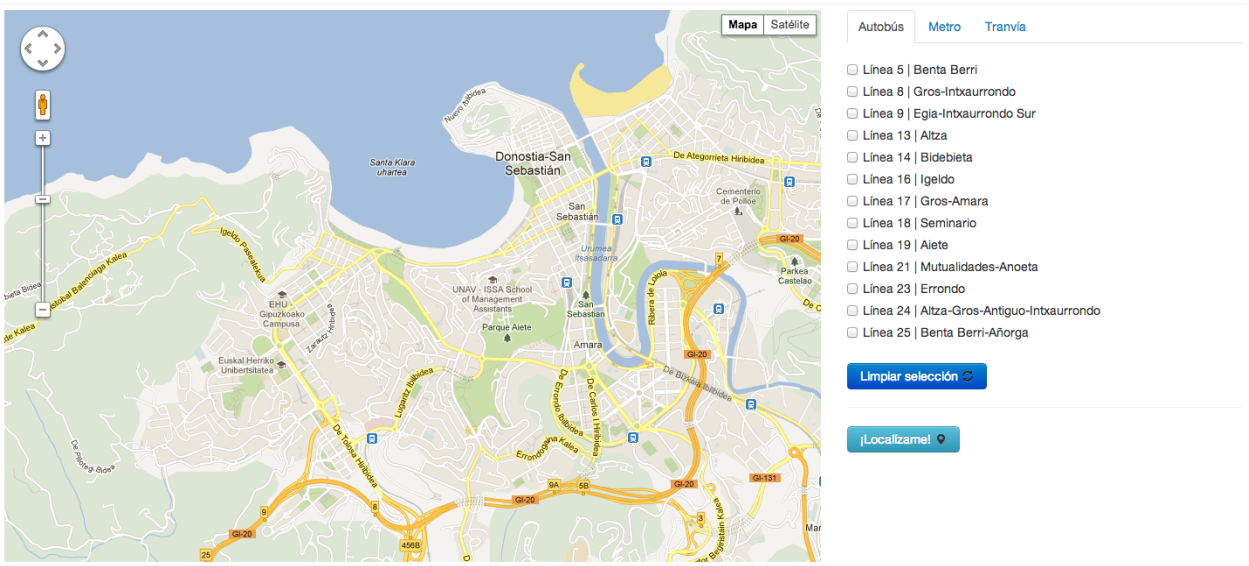

Figura 3.4. Pantalla Home

Estas listas que se cargarán con la página contarán con *checklist*, que al seleccionar una línea específica cargará la ruta de la misma y en caso de las líneas de autobuses, mostrará los autobuses en circulación de la línea seleccionada. Por otro lado, el lado derecho de la página también contará con un botón *Limpiar Selección*, que quitará el inspector a las líneas que hayamos seleccionado.

Por último, el botón *¡Localízame!* utilizará las propiedades de geolocalización de HTML5 para situar el mapa allá donde nos encontremos. Así mismo, a la vez que nos localiza, el sistema comprobará si existe alguna línea de transporte en la base de datos cercana al punto donde nos encontremos. En caso afirmativo, cargará las líneas existentes en el sistema.

3.3.3. Pantallas Contacto y About

Las otras dos pantallas que contiene el servicio son las recogidas por las figura *Pantalla contacto* y *Pantalla about*. La primera de estas pantallas se utilizará para acercar el servicio al usuario, de forma que éste se pueda poner en contacto con los gestores del servicio mediante varios métodos, para así expresar sus dudas, sugerencias, quejas, etc.

Por otro lado, la pantalla About, mostrará una breve descripción del concepto que representa el sistema. También se podría pensar como una línea de marketing, o publicidad de lo que ofrece el servicio Euskadi Moving.

Tal y como se ha comentado previamente, estas pantallas pueden ser modificadas sin nin-

Figura 3.5. Pantalla contacto

gún problema, así como también se podrán añadir pantallas adicionales si se viera que fuera necesario.

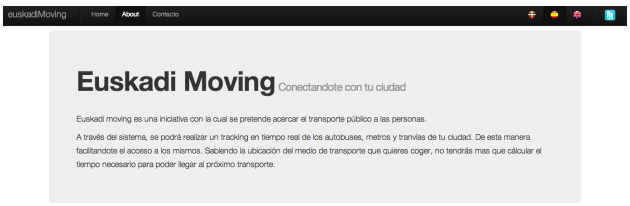

Figura 3.6. Pantalla about

3.4. Tarea 2: Adquisición de datos

La adquisición de datos (rutas, líneas, horarios...) se ha realizado de manera manual ya que no se ha tenido acceso a la información de las líneas de transporte público de Donostia - San Sebastián. Como paso principal, se ha llevado a cabo la obtención de las rutas del transporte público. Para ello, se ha hecho uso de la página de la compañía dbus[2], que muestra las rutas existentes de autobuses. Podemos observar el resultado en la figura *Ejemplo de rutas obtenidas a través de dbus*.

Basándonos en la información suministrada por la página de la compañía dbus y utilizando a la vez el programa Google Earth con la herramienta *regla-ruta*, tal y como podemos observar en la figura *Consiguiendo coordenadas de ruta*, se han obtenido las listas de coordenadas necesarias para el dibujo de las líneas de transporte sobre el mapa de Google Maps. Así mismo, la línea dibujada resultante tendrá el color representativo de dicha línea.

Para hacer mas liviana la aplicación, se creará un fichero javascript por cada ciudad que cuente con el sistema. De esta manera, conseguimos que únicamente se descargue del servidor aquel

[2] http://www.dbus.es

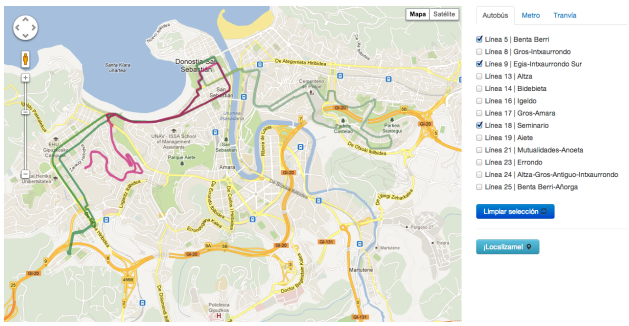

Figura 3.7. Ejemplo de rutas obtenidas a través de dbus

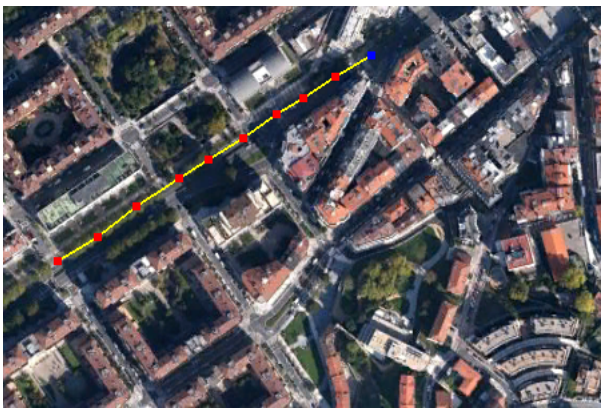

Figura 3.8. Consiguiendo coordenadas de ruta

fichero necesario en cada momento. Esto se ha hecho de esta manera ya que estos ficheros cuentan con un considerable tamaño, por tanto, si descargáramos todos sin criterio alguno, el funcionamiento de la página empeoraría.

Cada uno de estos ficheros contará con la información necesaria de las rutas de cada ciudad, así como la URI, identificador uniforme de recursos (del inglés: Uniform Resource Identifier) del servidor de ubicaciones correspondiente a dicha ciudad.

Como añadido fuera del alcance del proyecto, se plantea un método automatizado para la obtención de las coordenadas. Este sistema sería capaz de obtener las rutas de manera automatizada, bien mediante la información suministrada por una empresa externa o a través de *web scrapping*[3], es decir extrayendo información de otras web. Así mismo, este sistema mantendría actualizadas las líneas ya existentes.

3.5. Tareas 3 y 4: Back-end y servidores de ubicación

El primer servidor de ubicación que se va a crear es el de la ciudad de Donostia - San Sebastián, para ello como servidor de pruebas se creará una máquina virtual utilizando la versión de servidor de la distribución de Linux Ubuntu 12.04 LTS o Servicio de largo plazo (del inglés: Long Term Service)[4]. Esta maquina que vamos a virtualizar, después se exportará a un servidor final en el cual se realizarán las pruebas de funcionamiento.

La ventaja que tenemos al utilizar una máquina virtual es que esta máquina puede ser exportada y modificada con suma sencillez, permitiéndonos de esta manera adaptarla a las necesidades de cada ciudad, de cada servidor, etc.

Por tanto, como se ha explicado previamente, se va a optar por configurar una máquina virtual lo mas estándar posible y que sea capaz de ajustarse a las necesidades de la mayoría de los futuros clientes.

3.5.1. Características de los servidores

Los servidores llevarán instalado el servidor web Apache, que se utilizará para suministrar la información a los clientes. La estructura que seguirá aproximadamente (dependerá de la ciudad y sus líneas) se puede observar en la figura *Estructura de directorios del servidor de*

[3] http://es.wikipedia.org/wiki/Web_scraping
[4] https://wiki.ubuntu.com/LTS

ubicaciones. Tal y como se ha planteado en las intenciones del proyecto, lo ideal sería que se construyeran APIs públicas, por tanto el acceso a la información suministrada por nuestro ejemplo será abierto.

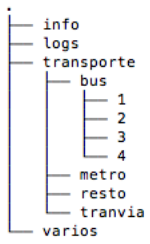

Figura 3.9. Estructura de directorios del servidor de ubicaciones

Así que si un cliente solicita la información de la línea número 1 de la ciudad de Donostia - San Sebastián se deberá seguir la siguiente ruta:

- http://dirección_ip_servidor_donosti/transporte/bus/1/fichero.js

En este fichero se encontrará la última información recabada sobre todos los autobuses de la línea 1. Este fichero se estará constantemente actualizando con la información que se vaya recibiendo por parte de los dispositivos GPS instalados en los transportes públicos. Este fichero y los de su tipo tienen que estar constantemente actualizados, para lo cual se contará con un programa que utilizará la última información recabada y actualizará el fichero.

Si bien está fuera del alcance del proyecto el análisis de las velocidades de transmisión de la información, se estima que el uso del protocolo UDP en el envío de datos desde los dispositivos GPS hasta el servidor de ubicaciones será de gran utilidad debido a la velocidad que nos otorga.

3.5.2. Seguridad de los servidores

La seguridad de los servidores está fuera del alcance del proyecto, debido en parte al desconocimiento sobre técnicas de seguridad. No obstante, se intentará ofrecer cierta seguridad. Para ello se hará uso de un firewall que solamente permitirá el acceso al servidor a través de ciertos puertos, en nuestro caso:

- Puerto 80 -> Web
- Puerto 22 -> FTP
- Puerto 23 -> SSH

- Puertos necesarios para la recepción de información por parte de los dispositivos GPS

Una vez el sistema estuviera instalado de manera final, sería recomendable realizar una auditoria de seguridad sobre los servidores de ubicaciones, para de esta manera evitar el ataque y por tanto posible rotura o hackeo de los mismos.

3.5.3. Simulación de posiciones

Dado que no se tiene acceso a datos reales de posiciones, en el proyecto estas serán simuladas. Para ello se hará uso de los datos que se han obtenido en la Tarea 2 *Adquisición de datos*.

Los datos obtenidos en dicha tarea están almacenados en forma de tabla hash, donde el índice de la tabla es la línea de transporte y el contenido es una lista de tamaño N con pares de números que representan latitud y longitud de un punto en el mapa. Este formato lo podemos observar en el siguiente fragmento de código (recortado para el ejemplo):

```
coordenadas_linea_bus['5'] = ["43.32165,-1.98497","43.320057,-1.983897",
      "43.319589,-1.984584","43.318746,-1.984112","43.318465,-1.984884",
      "43.317856,-1.984559","43.31756,-1.985224","43.316514,-1.986276",
      "43.315296,-1.987928","43.3145,-1.992262","43.314625,-1.996704",
      "43.315546,-1.999172","43.315405,-2.000715","43.314984,-2.00035",
      "43.314718,-2.000533","43.314109,-2.001895","43.312454,-2.004588",
      "43.311799,-2.005393","43.310128,-2.008794","43.309917,-2.008944",
      "43.309667,-2.008955","43.308668,-2.008","43.305202,-2.004749"];
```

El proceso que se llevará a cabo para este caso en particular será el siguiente: cada vez que se actualice el fichero de posiciones de la línea de autobuses número 5, se cogerá la siguiente posición a la actual, de esta manera se hará una simulación bastante aproximada del avance de los autobuses.

3.5.4. Tarea 5: Puesta en marcha del sistema

Para la puesta en marcha del sistema, se deben llevar a cabo los siguientes pasos:

- Pasar el proyecto desde *desarrollo* a *producción*, para ello debido a que estamos utilizando Django, se han de seguir ciertos pasos que se detallarán a continuación.
- Configurar en el servidor final las tecnologías a utilizar (esta parte se detalla en el *Anexo: Instalación y configuración del servidor*

Si bien, tal y como se ha planteado a lo largo del proyecto, lo ideal sería hacer uso de una arquitectura distribuida, debido a los limitados medios con los que se cuenta, únicamente se hará uso de un servidor Linux y será este también el que tenga el fichero de posiciones de los autobuses de Donostia - San Sebastián.

3.5.5. Configuración de Django

El framework Django, por defecto, viene configurado en modo *debug*, esto es, preparado para el desarrollo de páginas web, de forma que nos muestre grandes cantidades de información en caso de fallos para de esta manera poder depurarlos mas fácilmente. Por ello, antes de pasar el proyecto al servidor final de producción, habrá que cambiar todas estas variables y configuraciones para cambiar el framework y que no muestre esta información en caso de fallo, ya que podría derivar en un gran problema de seguridad.

Ya que hemos utilizado git con bitbucket como sistema de control de versiones, bastará con actualizar el repositorio y realizar un *clone* en el servidor de producción. Así mismo, una vez tengamos la copia del proyecto habrá que realizar un *dump* de la base de datos, para poder pasar la información de la base de datos desde el proyecto en desarrollo al de producción. Para ello usaremos la herramienta *dumpdata* proporcionada por Django y tras esto la herramienta inversa *loaddata* para cargar la información.

4. CAPÍTULO

Resultados, Conclusiones y Líneas Futuras

A continuación, en esta sección de la memoria, se presenta el resultado del proyecto y se van a comentar las conclusiones que se han obtenido tras el desarrollo del proyecto. Así mismo, también se plantean las amplias líneas de futuro que tiene el proyecto.

4.1. Resultados

Una vez finalizado el proyecto, se constata que de los objetivos planificados recogidos por el punto *Objetivos* de la memoria, se han abordado todos ellos y llegando correctamente al desarrollo y cumplimiento de los mismos, a saber:

- Se ha desarrollado correctamente el servicio web para que ofrezca la información desglosada a nivel de línea de transporte público.
- Cada línea muestra correctamente los autobuses que el simulador de posiciones genera para cada línea.
- La información se actualiza en el periodo de tiempo establecido por el sistema, por tanto generando la visualización en tiempo real.
- El servicio web cuenta con el sistema de geolocalización, que es capaz de localizar la posición de nuestro punto de acceso a la red, y centrar el mapa del servicio en dicho lugar. En caso de que para dicho lugar existan líneas insertadas se carga la información sobre ellas.

- El diseño resultante ha seguido la estructura de *flow layout* de las hojas de estilo de twitter bootstrap. Al utilizar ésta opción, el diseño se adapta dinámicamente al navegador del cliente, permitiéndonos así adaptarnos a cada usuario.

Por otro lado, el diseño generado para enviar las posiciones se ha creado siguiendo el formato de intercambio de datos JSON, acrónimo de JavaScript Object Notation, el cual utiliza un subconjunto de sintaxis basada en javascript para la generación de ficheros de intercambio de datos. Esto nos lleva a que el objetivo de facilitar la adaptabilidad a servicios de posicionamiento de otras compañías se haya cumplido, ya que una de las características de JSON es su facilidad de uso.

Por último, en un principio no se planteo el desarrollo del sistema en varios idiomas, no obstante, a lo largo del desarrollo se decidió la inclusión de la opción de ser visualizada la página en varios idiomas, pero sin llegar a incluir la traducción de la misma. La inclusión de este objetivo, no ha supuso una gran desviación temporal debido a que únicamente se ha planteado la opción del cambio de idioma, sin llegar a aplicar la traducción en sí.

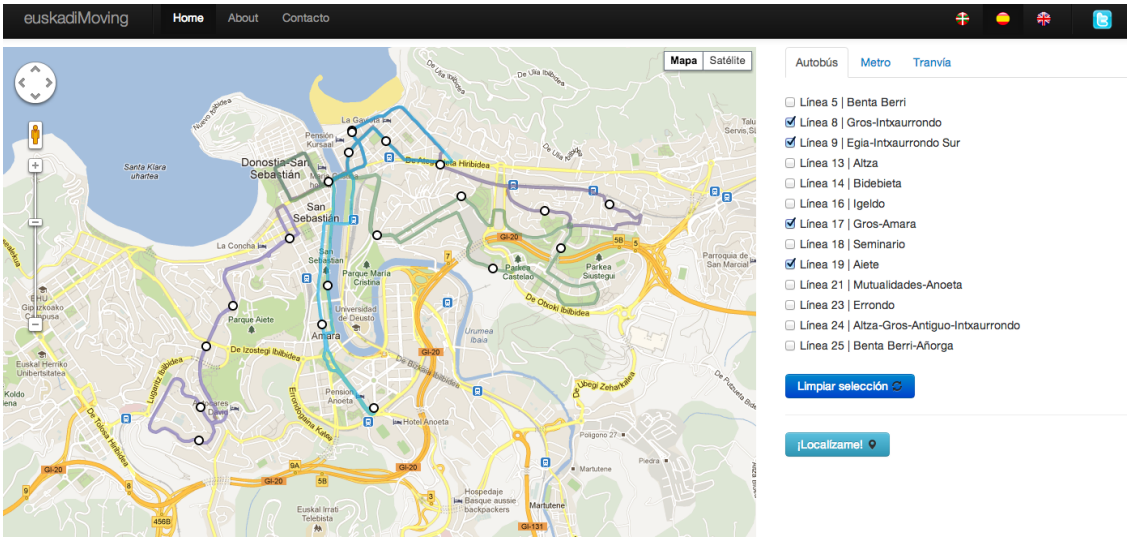

Figura 4.1. Servicio en funcionamiento

El resultado final del servicio lo podemos observar en la figura *Servicio en funcionamiento* y también a través de la página www.euskadimoving.com. En el resultado final, se ha decidido actualizar las posiciones de los transportes cada 2 segundos. Para ello, se borran todos los marcadores dibujados en el mapa, y se procede después a dibujarlos de nuevo en la nueva posición. Este tiempo de actualización puede ser modificado para en una futura optimización del sistema.

4.2. Conclusiones

En esta sección se recogen las conclusiones alcanzadas a nivel profesional tras la realización del proyecto.

En primer lugar, se destaca que se han alcanzado todos los objetivos planteados en el alcance del proyecto de manera satisfactoria. En gran medida debido a la planificación realizada para el desarrollo del proyecto, ya que como se indica en el apartado *planific* el tiempo con el que se contaba para desarrollar el proyecto era bastante ajustado, ya que el mismo se ha llevado a cabo a la vez que se cursaba varias asignaturas, con sus respectivas cargas de trabajo.

Así mismo, también se considera que si bien se han planteado objetivos que se han dejado fuera del alcance del proyecto, estos han sido correctamente no incluidos en el alcance del proyecto ya que el tiempo de desarrollo necesario para llevarlos a cabo hubiera supuesto una desviación temporal demasiado grande.

Finalmente, cabe destacar que no se considero en un principio que el servicio debía mostrarse en varios idiomas, para así aumentar la utilidad del mismo. Esta funcionalidad fue incluida durante el desarrollo, no obstante, para no tener una gran desviación temporal finalmente no se tradujo el servicio al resto de idiomas.

4.3. Líneas Futuras

En este apartado se van a tratar las posibles líneas futuras que se han considerado para el proyecto.

4.3.1. Aplicaciones para móviles

Como paso a tomar en el proyecto si éste tuviera éxito, sería el de ampliar horizontes creando las aplicaciones nativas para los dispositivos móviles. Para ello, se podrá hacer uso de cada una de las peculiaridades de los sistemas móviles. Como es el sistema GPS que llevan incluido, esto nos podría servir para añadir filtros a las búsquedas de líneas, como podría ser buscar líneas cercanas, o autobuses que estén acercándose a mi posición, etc. Podemos observar una simulación en la figura *Ejemplo aplicación móvil*.

Figura 4.2. Ejemplo aplicación móvil

4.3.2. Versión para marquesinas

Una interesante línea de futuro es esta que se presenta a continuación. La idea sería añadir a las marquesinas de autobús/metro/tranvía una pantalla táctil que incluyera una versión limitada del sistema. Esta versión limitada, incluiría únicamente las líneas que pasarán por dicha marquesina, ofreciendo también el sistema táctil para poder elegir por ejemplo algún autobús en ruta y que éste nos mostrara la información relevante sobre el mismo.

4.3.3. Mejora y corrección de posiciones

Dado que la calidad de un dispositivo GPS de uso público puede variar, con un margen de error de hasta 100m, sería una interesante posibilidad la de añadir un sistema de corrección de la posición mediante algún método.

4.3.4. Fusión con el sistema actual de referencia

Dada la infraestructura que ya hay desarrollada en varias ciudades y con el flujo de información que generan, se podrían analizar maneras de combinar los datos que ya tienen con el sistema que se ha creado en este proyecto. Por ejemplo, al pinchar en el icono de un autobús, podría emerger una ventana que mostrara, mediante cual fuera el método usado hasta ahora, el tiempo estimado en llegar el autobús.

4.3.5. Análisis de la eficiencia del sistema

Ha quedado fuera del alcance del proyecto considerar la eficiencia final del sistema debido a la dificultad que supone, ya que no se cuentan con los medios necesarios para hacer una prueba real. Por tanto, se considera necesario en un futuro analizar la eficiencia del envío de datos, tiempos de carga de la página, calidad del javascript y en general todos los aspectos que pudieran influir en la eficiencia del sistema sería necesario.

4.3.6. Generación de una API

Generar una API pública al estilo del sistema de localización de Londres podría dar pie a nuevas iniciativas hacia el transporte público que podrían ser de gran utilidad.

4.3.7. Iconos, marca y logotipo

Otro punto a considerar como posible mejora, sería la inclusión de un logotipo que identificase inequívocamente el servicio, así como el desarrollo de iconos para representar los transportes que recorren el mapa. Pudiendo de esta manera diferenciarlos por línea, ciudad o tipo.

Así mismo, se podría considerar la modificación del nombre *euskadiMoving* por uno mas global o en su defecto crear una marca por cada ciudad o ayuntamiento que quisiera el servicio.

4.3.8. Expansión a otros escenarios

El sistema desarrollado, podría modificarse para el sector privado y así incluir el sistema de *tracking* a sus necesidades. Por ejemplo, una compañía de aviación podría incorporar sus sistema y mostrar a nivel global la localización actual de cada uno de sus aviones. Como otra alternativa, se podría instalar un sistema similar en el prestamos de bicicletas municipales para evitar posibles robos.

5. CAPÍTULO

Anexo: Prototipo localizador GPS

Este anexo cubre el diseño esquemático básico de un emisor de posición GPS, así como la definición de las características que este dispositivo al menos debería tener, para funcionar correctamente con el proyecto. Para ello se hará uso de la tecnología Arduino [1] debido a su extensa comunidad y basándose en proyectos existentes anteriores.

Al ser Arduino una placa electrónica para la creación de diversos sistemas, la placa básica incluye los conectores necesarios para añadir aquellos complementos necesarios para cumplir las necesidades de cada uso.

En este caso, serán necesarias las siguientes partes:

- Algún sistema de detección de posición global. Las opciones disponibles actualmente serían GPS o GLONASS.
- Algún sistema de telecomunicación para enviar la información a los servidores locales. Esta parte podría ser cubierta a través de conexión mediante redes telefónicas, o en caso de contar con Internet como medio de transporte, se podría hacer uso de dicha tecnología previa.
- Algún display del sistema para dar información del funcionamiento actual del dispositivo.
- Caja para empaquetar todo el sistema.

[1] http://http://www.arduino.cc/es/

5.1. Funciones necesarias del sistema

Como previamente se ha remarcado, el sistema deberá contar con un número mínimo de funciones para funcionar correctamente con el proyecto. Estas funciones deberán ser programadas en el firmware del sistema Arduino, objetivo que está fuera del alcance del proyecto.

El sistema deberá ser capaz de informar periódicamente, en tiempo real de la latitud, longitud, hora y de un identificador único. Este proceso se basará en las siguientes etapas:

- Obtención de la latitud y longitud a través del añadido localizador.
- Encapsulamiento de la información, uniendo para ello un identificador único asociado a la línea y al medio de transporte. Así mismo se unirá al paquete la hora de obtención de la posición
- Envío por medio de alguno de los métodos disponibles de la información al servidor asignado al dispositivo.

Debido a que la información provista a través del sistema localizador pierde importancia conforme avanza el tiempo, y dado que la velocidad y el tiempo de transmisión son vitales, todo este proceso deberá realizarse en un periodo de tiempo máximo establecido, cumpliendo así el objetivo de la información en tiempo real.

6. CAPÍTULO

Anexo: Instalación y configuración del servidor

Si bien el proyecto ha sido ideado para ser utilizado a través de una arquitectura distribuida, tal y como se podía ver en la figura 1 del capitulo 2 *Documento de Objetivos del Proyecto*, para la realización del proyecto y posteriores demostraciones del mismo, éste ha sido desarrollado utilizando únicamente un microservidor.

Este anexo cubre la instalación y configuración de dicho servidor, partiendo desde una máquina con la distribución de Linux Ubuntu 12.04 LTS o Servicio a largo plazo (del inglés: Long Term Service)[1]. Estas versiones tienen la características que contarán con soporte por parte de Canonical por al menos 5 años.

6.1. Software necesario

Para la instalación del software necesario, siempre que sea posible haremos uso del gestor de paquetes *apt*[2],este gestor de paquetes nos permite instalar el software deseado de manera sencilla, no obstante, esta instalación se podría hacer de cualquier manera y en caso de no ser posible, se detallará cómo se ha instalado el software.

Como primer servicio a instalar, procederemos a instalar el servidor web Apache, para ello ejecutaremos la siguiente instrucción en la línea de comandos:

[1] https://wiki.ubuntu.com/LTS
[2] http://es.wikipedia.org/wiki/Advanced_Packaging_Tool

```
sudo apt-get install apache2
```

Una vez instalado el servidor, para poder ejecutar código python, podemos optar por instalar o bien *mod_python*, o bien *mod_wsgi*. En este caso, se ha optado por *mod_wsgi* debido a que *mod_python* se declaro obsoleto a partir de la versión 1.3 de Django. Para proceder a la instalación ejecutamos el siguiente comando:

```
sudo aptitude install libapache2-mod-wsgi
```

Tras esto, reiniciamos el servicio apache para que detecte la nueva instalación:

```
sudo service apache2 restart
```

Como siguiente paso, procedemos a instalar la base de datos, en este caso, PostgreSQL. La instalación de este programa, requiere una configuración en la cual se solicita nombre de usuario y contraseña. En nuestro caso es irrelevante los valores de los mismos, por tanto obviamos este paso. Comando de instalación:

```
sudo aptitude install postgresql-8.2 postgresql-client-8.2 pgadmin3
```

Para poder descargar y utilizar el código desarrollado para el proyecto instalamos el controlador de versiones Git:

```
sudo apt-get install git-core
```

Debido a que queremos utilizar el dominio adquirido **www.euskadimoving.com** y la máquina que va a contar con el servicio está conectada a Internet a través de una IP dinámica, haremos uso del servicio de *noip2* para poder conectarnos a través de nuestro dominio, para ello:

```
sudo apt-get install wget
wget -c http://www.no-ip.com/client/linux/noip-duc-linux.tar.gz
tar xvzf noip-duc-linux.tar.gz
cd noip-2.1.9-1
make
make install
```

y para iniciar el servicio:

```
sudo noip2
```

Por último, tenemos que instalar el framework Django, para ello ejecutamos los siguientes comandos:

```
wget https://www.Djangoproject.com/download/1.4/tarball/
tar xzvf index.html
cd Django-1.4
sudo python setup.py install
```

6.2. Configuración del servidor

Para obtener el código del proyecto, que recordemos se encuentra en el repositorio de bitbucket, crearemos un directorio en nuestro equipo con el nombre del proyecto, para dejar aquí el proyecto y que será el directorio al que después accederá el servidor web apache:

```
cd /home
mkdir euskadimoving
```

y tras esto, se hará uso del siguiente comando de git:

```
git clone git@bitbucket.org:valdeca/euskadimoving.git
```

El framework Django hace uso de la librerías mod_wsgi para poder ejecutar código python. Para indicar a Apache que nuestro servidor virtual hará uso de este módulo debemos añadir las siguientes líneas en el fichero *httpd.conf*:

```
WSGIScriptAlias / /home/valdecaserver/Django/euskadimoving/apache/Django.wsgi
PythonPath "['/home/valdecaserver/Django/euskadimoving'] + sys.path"
```

Con estas líneas indicamos al servidor dónde tenemos nuestro script de ejecución y se la añadimos al path de Python. Este script tendrá la siguiente forma:

```
import os
import sys
path = '/home/valdecaserver/Django/euskadimoving'
if path not in sys.path:
    sys.path.append(path)
os.environ['Django_SETTINGS_MODULE'] = 'dMoving.settings'
import Django.core.handlers.wsgi
application = Django.core.handlers.wsgi.WSGIHandler()
```

Django únicamente se encarga de gestionar el contenido dinámico de la página web, por tanto, el siguiente paso, será configurar el servidor a utilizar para servir los ficheros estáticos. Como

buena práctica se recomienda utilizar un servidor web distinto a aquel que esté utilizando Django, no obstante, en este caso debido a la infraestructura disponible, se ha utilizado el mismo servidor.

Así mismo, en el fichero de configuración tenemos que indicar dónde se encuentra la página web para que Apache pueda llegar a ella, aquí también indicaremos la ruta donde se encuentran los ficheros estáticos y se le asignará el alias que tenga en el código de la página, en este caso **/static/**. La configuración final resultante del fichero *httpd.conf* será la siguiente:

```
<VirtualHost *:80>
    ServerName www.euskadimoving.com
    ServerAdmin contacto@euskadimoving.com
    DocumentRoot /home/valdecaserver/Django/euskadimoving
    Alias /static/ /home/valdecaserver/Django/euskadimoving/static2/

   <Directory /home/valdecaserver/Django/euskadimoving/static2/>
      Order deny,allow
      Allow from all
   </Directory>

    WSGIScriptAlias / /home/valdecaserver/Django/euskadimoving/apache/Django.wsgi
    PythonPath "['/home/valdecaserver/Django/euskadimoving'] + sys.path"

    LogLevel warn
    ErrorLog /home/valdecaserver/logs/eMoving/error.log

</VirtualHost>
```

La última entrada del fichero de configuración, indica el nivel de log que queremos para nuestro servidor virtual. En este caso nivel Warning.

Una vez el fichero esté completado habrá que reiniciar apache para que se haga uso de las nuevas incorporaciones al fichero correctamente.

6.3. Actualizador de posiciones

Debido a que las posiciones de los autobuses en este proyecto van a ser simuladas, para ello se ha creado un script que simula las posiciones de los autobuses de la ciudad de Donostia

- San Sebastián. Este script utilizará las coordenadas obtenidas para dibujar las rutas de los autobuses, e irá iterando las posiciones para simular el movimiento del autobús.

```
sumador= 0
socketServer = socket.gethostname()
while 1==1:
	sumador =sumador+1
	now = datetime.datetime.now()
	if socketServer == 'vaxaServer':
		fichero = open("ruta_al_fichero/datos_donostia.js", "w")
	fichero.write("{\n\t")
	fichero.write("\"actualizacion\":")
	fichero.write("\""+str(now)+"\",")
	fichero.write("\n\t")
	fichero.write("\"codigo\":\"1\",")
	fichero.write("\n\t")
	fichero.write("\"nombre_ciudad\":\"Donostia\",")
	fichero.write("\n\t")
	fichero.write("\"buses\":[")
	for linea_autobus in coordenadas_linea_bus:
		fichero.write("\n\t\t{")
		fichero.write("\n\t\t\"id\":\""+linea_autobus +"\",")
		fichero.write("\n\t")
		fichero.write("\t\"titulo\":\""+str(coordenadas_linea_bus[linea_autobus][0]) +"\",")
		fichero.write("\n\t")
		fichero.write("\t\"sig\": [\n")
		longitud= len(coordenadas_linea_bus[linea_autobus])
		pos = longitud/5
		for i in [1,2,3,4,5]:
			posicion=pos*i
			posicion=posicion+sumador
			while posicion >= longitud:
				posicion = posicion-longitud
			if posicion == 0:
				posicion = posicion + 1
			fichero.write("\t\t\t{")
			fichero.write("\n\t\t\t\t\t")
			hora = "7:00"
```

```
                if i == 2:
                      hora = "7:20"
                if i == 3:
                      hora = "7:40"
                if i == 4:
                      hora = "8:00"
                if i == 5:
                      hora = "8:20"
                fichero.write("\"salida\":\""+hora+"\",")
                fichero.write("\n\t\t\t\t")
                fichero.write("\"id\":\""+str(i)+"\",")
                fichero.write("\n\t\t\t\t")
                fichero.write("\"position\": \
                new google.maps.LatLng("+coordenadas_linea_bus[linea_autobus]\
                [posicion].partition(",")[2] +","+   \
                coordenadas_linea_bus[linea_autobus][posicion]\
                .partition(",")[0]+"),")
                fichero.write("\n\t\t\t")
                if i == 5:
                      fichero.write("}")
                else:
                      fichero.write("},")
                fichero.write("\n")
          fichero.write("\t\t\t]")
          fichero.write("\n\t\t},")
    fichero.write("\n\t")
    fichero.write("]\n")
    fichero.write("}")
    fichero.close()
    time.sleep(1.5)
```

El script se actualizará cada 1,5 segundos, de esta manera representando aproximadamente el tiempo que tardarían los emisores GPS de los autobuses en enviar su posición al servidor de posiciones y este a su vez computar los datos y generar un fichero similar al producido por este script.

Una vez configurado el script a nuestra necesidad los ejecutaremos con el siguiente comando:

```
actualizador.py &
```

De esta manera, dejaremos el script ejecutándose en segundo plano actualizando constantemente las posiciones de los autobuses.

Bibliografía

[Django-book] Adrian Holovaty, Jacob Kaplan-Moss, et al.(2012). *The Django book*.

[Sphinx-Docu] Georg Brandl. *Sphinx documentation*. http://sphinx-doc.org/. The Pocoo Team, Python community.

[Google-Maps-Api] Google Team. *Google Maps API v3.0*. https://developers.google.com/maps/?hl=es. Google Inc.

[Python-para-todos] Raúl González Duque. *Python para todos*.