ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
GRADO EN INGENIERÍA INFORMÁTICA

CTPATH: Desarrollo de una aplicación iOS para el cálculo de rutas ecológicas

CTPATH: Development of an iOS application to compute ecological routes

Realizado por
Francisco Navarro Aguilar
Tutorizado por
José Francisco Chicano García
Departamento
Lenguajes y ciencias de la computación

UNIVERSIDAD DE MÁLAGA
MÁLAGA, Junio de 2016

Fecha defensa:
El secretario del tribunal

Resumen: El *cambio climático* se debe en gran parte a la emisión de gases de efecto invernadero como el dióxido de carbono, por tanto, es necesario desarrollar nuevas tecnologías que reduzcan la emisión de estos gases a la atmósfera. Actualmente, las aplicaciones para calcular rutas no tienen en cuenta la relación entre el tiempo de viaje y las emisiones, ni el perfil de conducción del usuario y el tipo de coche. El proyecto *CTPATH* tiene en cuenta estos factores. *CTPATH* para *iOS* tiene como objetivo *reducir* la cantidad de *gases de efecto invernadero* que emiten los vehículos cuyo combustible es un derivado del petróleo mediante el uso de rutas más ecológicas. Cualquier usuario puede utilizar la aplicación *sin necesidad de registrarse*, aunque si lo desea, podrá hacerlo a través de la aplicación web existente. Registrarse en la aplicación ayuda a calcular cuál es la ruta que mejor se adapta a su forma de conducir y por tanto, ofrecer una mejor experiencia al usuario. Cuando el usuario introduce el punto de inicio y el punto de destino se le muestra una lista de *tres posibles rutas* a tomar, donde la *más ecológica* aparece *en primer lugar*, seguido de otras rutas alternativas que son menos ecológicas y posiblemente más rápidas que la primera. Puede darse el caso de que la ruta *más rápida* y la *más ecológica* coincidan, es decir, sean la misma. También puede ocurrir que exista *una única ruta* o que existan sólo *dos rutas diferentes* para llegar al destino, en ese caso, una misma ruta aparecerá varias veces en la lista. Cuando el usuario seleccione una ruta, podrá ver las *cantidades de gases nocivos* emitidos para esa ruta, y desplegar una tabla con las *indicaciones que debe tomar* para llegar a su destino.

Palabras clave: *contaminación, cambio climático, iOS, aplicación móvil*

Abstract: *Climate change* is produced by pollution and gases like carbon dioxide, so it is necessary to develop new technologies to reduce the emission of these gases to the atmosphere. Nowadays, the applications to compute routes do not take into account the relation between trip time and pollution,neither driving profile, nor the type of car. *CTPATH* project takes into account these factors. *CTPATH* for *iOS* tries to *reduce* the amount of *pollution* emitted by vehicles which uses petroleum-base fuel, by using more ecological routes. Any user can use the application *without registering*, but if it is desired by the user, it may be done through the web application. Signing in the application helps us to compute the best route for a specific user. When the user enters the starting point and the destination point, a list is shown with *three possible routes* to take, where the *most ecological appears first* followed by other additional routes that are less friendly with the environment and possibly faster to the first one. It can happend that the *fastest* route and the *most friendly with environment* match, i.e, they are the same. It can also happen that there is just a single path or there are just two different routes to the destination, in this case, the same route appears several times in the list. Once the user selects a route, s/he can see the *amount of harmful emitted gases* for that route, and s/he can also display a table with the *instructions* to reach the destination.

Keywords: *pollution, climate change, iOS, mobile application*

Índice general

Capítulo 1

Introducción

En esta sección se introduce al lector al proyecto CTPATH, al porqué de este trabajo, y se explican los objetivos que se intentan alcanzar a través de la realización del mismo.

1.1 Contexto

La **contaminación del aire** es un problema muy grave que acarrea numerosas consecuencias para nuestra salud, al aumentar el riesgo de padecer enfermedades respiratorias como neumonía o cáncer de pulmón [1]. Además de este gran problema, durante los últimos años, y a consecuencia de la contaminación atmosférica, estamos experimentando un cambio en el clima en todo el mundo y con ello también en nuestra ciudad. El **cambio climático** [2] es un acontecimiento que ha ocurrido siempre desde la existencia de nuestro planeta. El problema que tenemos en la actualidad, es la velocidad con la que se está produciendo en las distintas zonas del mundo.

Un **factor** que eleva las emisiones de gases nocivos a la atmósfera son los vehículos que funcionan mediante derivados del petróleo, como la gasolina o el gasóleo, entre otros. La quema de estos **combustibles** libera *dióxido de carbono* (CO_2)*, monóxido de carbono (CO), óxidos de nitrógeno* (NOx)*, hidrocarburos no quemados* (HC) *y otros gases en menor cantidad* [3]. Tal emisión no es siempre la misma, ésta varía en función del tipo de vehículo y su antigüedad, el tiempo que está parado, la distancia que recorre, y otros muchos factores, de tal manera que un conductor no puede saber de una forma sencilla, qué ruta es la más ecológica para llegar a su destino.

Actualmente existe un proyecto llamado **CTPATH** que tiene en cuenta la **emisión de gases nocivos** a la atmósfera del vehículo y el **tipo de conductor** para calcular las distintas rutas entre dos puntos. Un caso de estudio para el proyecto es la ciudad de Málaga, donde el proyecto está siendo aplicado actualmente ofreciendo a los usuarios **rutas más beneficiosas** para el medioambiente.

1.2 Proyecto CTPATH

Extraído directamente del documento *Technical Design* (ES) [5] del proyecto:

"CTPATH es un planificador de rutas **inteligente** y **holístico** para entornos urbanos. Es **inteligente** por el tipo de tecnología y algoritmos computacionales usados (inspirados en la naturaleza, entre otros) y es **holístico** porque analiza todo el contexto del viaje dinámicamente, no únicamente al usuario ni únicamente la ciudad, como otros competidores".

Los cálculos de rutas se realizan fijando **dos objetivos**: obtener el camino más corto en tiempo y el que menos emisiones de gases nocivos genera. Para abordar este problema se tiene en cuenta el estado del tráfico, el vehículo usado en el trayecto, y el perfil del conductor que solicita la ruta.

1.2.1 Cálculo de emisiones

Daremos una breve explicación sobre los elementos que se tienen en cuenta para calcular la cantidad de gases emitidos en base a la documentación que se proporciona del proyecto [4]. Se considera la ciudad como un **grafo** G = (V,E) donde E es el conjunto formado por las distintas calles de la ciudad, y V es el conjunto de intersecciones entre las calles. Cada calle se etiqueta con cuatro valores:

- Longitud de la calle $l(e)$
- Máxima velocidad permitida en la calle $m(e)$
- Probabilidad de atravesar la calle sin parar a causa del tráfico $p(e)$
- Tiempo de espera en el caso de parar a causa del tráfico $\tau(e)$

Dada una solución σ como una secuencia de aristas, la probabilidad de completar la solución sin parar a causa del tráfico es la siguiente:

$$p(\sigma) = \prod_{e \in \sigma} p(e) \tag{1.1}$$

Una vez definido el grafo de la ciudad se especifica un **perfil de velocidad** para posteriormente calcular las funciones objetivos comentadas anteriormente. Estas funciones se evaluarán para cada conductor, por tanto el perfil de velocidad depende del **perfil de conducción**, teniendo en cuenta la aceleración, frenado, y la velocidad de crucero que tiene el conductor.

1.2.2 Aplicaciones existentes

El proyecto dispone de una **aplicación web** y de una **aplicación móvil** para el sistema operativo *Android*. Ambas aplicaciones realizan una petición *HTTP* a un *API REST*,

también desarrollada por el equipo del proyecto, que realiza los cálculos necesarios para hallar la mejor ruta y devuelve esta información en un texto con formato *JSON*. Cualquier dispositivo puede acceder a esta *API* a través de la URL de la misma, enviar los parámetros necesarios, y recibir la información calculada por el servicio web. *CTPATH* se proporciona como **código abierto** para todo el que esté interesado en utilizarlo o mejorarlo. Como el proyecto carece de aplicación para el sistema operativo *iOS*, vamos a desarrollar la aplicación para este sistema.

1.3 Objetivos

Desarrollaremos una **aplicación iOS** para complementar el proyecto CTPATH, esta aplicación se conectará al servicio web que realiza los cálculos comentados anteriormente y mostraremos los resultados en el dispositivo. De esta manera añadiremos al proyecto raíz el sistema operativo de *Apple* con el fin de dar facilidad al conductor para utilizar la aplicación en cualquier momento y en cualquier dispositivo. El simple hecho de que el usuario pueda conocer la forma de llegar a su destino de una manera más beneficiosa para el medio ambiente, justo antes de arrancar el vehículo, y mediante su teléfono, es una manera de conseguir que utilice cómodamente este servicio web.

Gracias a esta comodidad, habrá **mayor número de usuarios** que la usen, por lo que conseguiremos **disminuir** la **emisión** diaria de gases que contaminan la atmósfera, así como **reducir** el **tráfico**, al aportar al conductor los distintos caminos que puede tomar.

1.3.1 Disminución de emisiones

Aunque nuestra aplicación ofrecerá al usuario la ruta más corta en tiempo, mostrará en primer lugar el trayecto más ecológico (éstos pueden, o no, coincidir). De esta manera, estaremos concienciando al conductor, haciéndole ver que existen unos trayectos que son más favorables para el medio ambiente que otros, con lo cual, estaremos contribuyendo a una limpieza de la atmósfera, y con ello, a una mejora en la salud de las personas y una ralentización del cambio climático.

1.3.2 Reducción del tráfico

Por cada conductor, nuestro sistema ofrecerá distintos itinerarios de ruta (aunque pueden coincidir en determinadas ocasiones) para el mismo punto de inicio y destino. Por lo que la fluidez del tráfico en las carreteras debería mejorar conforme aumenta el número de usuarios que utilicen esta aplicación.

1.4 Fases del trabajo

Con el fin de realizar este proyecto correctamente, es necesario planificar las fases en las que vamos a desarrollarlo y el número de horas que dedicaremos a cada una de ellas:

1. *Análisis del problema.*

   Esta fase consistirá en la toma de requisitos de la aplicación y la posterior elaboración del documento especificación de requisitos. Número de horas: 8

2. *Desarrollo de la aplicación*

   Una vez que tenemos claros los requisitos de la aplicación, realizaremos un modelado de nuestro software que nos ayude a abarcar todos sus casos de uso de manera eficiente, tras esto, procederemos con la implementación de la misma. Número de horas: 110

3. *Testeo*

   Conforme se vayan implementando funcionalidades, iremos realizando las pruebas oportunas para comprobar que el software se está desarrollando sin errores. Número de horas: 136

4. *Documentación*

   Con el propósito de facilitar la comprensión de nuestro software a un alto nivel, realizaremos un informe detallando el diseño de nuestra aplicación y las características que tendrá. Número de horas: 42

1.5 Estructura de la memoria

Esta memoria se divide en siete secciones principales donde se tratan los temas más relevantes del proyecto. Estos capítulos son los siguientes:

- **Capítulo 1**. En este capítulo se introduce al lector al proyecto CTPATH, al porqué de este trabajo, y se explican los objetivos que se intentan alcanzar a través de la realización del mismo.

- **Capítulo 2**. En este capítulo se proporciona una visión general del proyecto software desarrollado comentando el propósito de realizarlo, el ámbito, las restricciones que se deben tener en cuenta y las funcionalidades que debe implementar.

- **Capítulo 3**. Entramos en la fase de modelado de la aplicación. Comenzaremos comentando la distintas arquitecturas utilizadas para el desarrollo de nuestra aplicación, tras esto, mostraremos el diagrama de clases de nuestro proyecto de una forma general, y explicaremos cada una de las clases implementadas.

- **Capítulo 4**. Corresponde con el diseño de la interfaz de usuario, por tanto, explicaremos como se han construído las distintas interfaces gráficas mostrando la disposición de cada uno de los elementos gráficos a partir de *mockups*.
- **Capítulo 5**. En este capítulo se explican las distintas pruebas realizadas para validar el software desarrollado, es decir, los requisitos funcionales requeridos funcionan correctamente.
- **Capítulo 6**. En este capítulo comentaremos las conclusiones que extraemos de todo el proceso de realización de este Trabajo de Fin de Grado, y discutiremos sobre posibles mejoras, y trabajos futuros relacionados con este trabajo.

Capítulo 2

Especificación de requisitos

Para especificar los requisitos necesarios del sistema se ha seguido la norma *IEEE Prácticas recomendadas para la obtención de requisitos software* [6]. Esta norma explica cómo deben redactarse las especificaciones del software para la correcta implementación del mismo. Siguiendo esta guía, se ha redactado el documento de especificación de requisitos (ver Anexo B) donde se detalla toda la información necesaria para comprender sin ambigüedades el proyecto a desarrollar.

Los **requisitos** descritos en este documento se han obtenido a partir de varias reuniones con el director del proyecto donde se comentaba el propósito de este TFG, su ámbito, la perspectiva del mismo, los requisitos funcionales y no funcionales que debía implementar, qué usuarios finales tendría la aplicación desarrollada y algunos comentarios relacionados con el proyecto.

En esta sección se comenta la parte fundamental del documento, para una visión más detallada se puede consultar el documento completo en el anexo B de esta memoria.

2.1 Propósito

La aplicación a desarrollar pretende complementar el proyecto *CTPATH* ya que carece de aplicación *iOS*. Por tanto, este documento va dirigido al desarrollador de la aplicación, el autor de estas líneas, para conocer las especificaciones requeridas y a su vez, va dirigido al grupo de desarrollo del proyecto *CTPATH* para que el software desarrollado pueda ser comprendido correctamente.

2.2 Ámbito del sistema

El futuro sistema, en adelante *CTPATH-iOS* permitirá poder utilizar las rutas proporcionadas por el servidor CTPATH en dispositivos móviles con sistema operativo iOS. Las funciones que se desarrollen en esta aplicación están limitadas a la funcionalidad

del propio servidor. *CTPATH-iOS* permitirá al usuario seleccionar el punto de inicio y el punto de destino, y mostrará las rutas más ecológicas que unen estos dos puntos. Estas rutas han sido previamente calculadas por el servidor CTPATH y podrán ser personalizadas si el usuario inicia sesión en la aplicación, pero no es una acción obligatoria para poder utilizar la aplicación. El registro del usuario en la aplicación no está contemplado, para poder darse de alta, deberá acudir a la página oficial del proyecto *CTPATH* (`https://mallba3.lcc.uma.es/ctpath/`).

2.3 Perspectiva del producto

CTPATH-iOS es un complemento de un proyecto mayor denominado *CTPATH*, el cual posee una aplicación web, una aplicación *Android* y un servidor web, el cual sirve las peticiones solicitadas por estas aplicaciones. Las funcionalidades implementadas en *CTPATH-iOS* deben ser las mismas al resto de aplicaciones al tratarse del mismo proyecto, y a su vez, estas funcionalidades deben ser implementadas a través de peticiones *HTTP* a un servicio *REST*, de tal manera, que la aplicación muestra los datos que el servidor ha calculado, esto implica que la aplicación no realiza cálculos, sino que procesa la respuesta que obtiene del servidor.

2.4 Características de los usuarios

Al ser una aplicación que muestra al usuario las rutas más ecológicas de acuerdo, entre otros factores, al vehículo que utiliza, los usuarios finales de la aplicación serán las personas que dispongan de un vehículo cuyo combustible sea un derivado del petróleo (gasolina, gasóleo...). Además, en las primeras versiones de la aplicación, sólo es posible calcular rutas en la ciudad de Málaga, por tanto, estos usuarios serán ciudadanos de Málaga, turistas, y/o visitantes de la ciudad.

2.5 Suposiciones y Dependencias

Existen dos lenguajes de programación para desarrollar aplicaciones en *iOS*, uno es *Objective-C* y otro es *Swift*. La aplicación se ha programado en *Objective-C* ya que la versión existente de *Swift* en estos momentos no es definitiva y en cambio, *Objective-C* es un lenguaje con mucho más tiempo en el mercado, y por tanto, es más estable. Si se hubiese realizado en la versión actual de *Swift*, una versión futura con grandes cambios podría ocasionar la refactorización del código implementado. No obstante, parece ser que Apple busca hacer de *Swift* el lenguaje principal, por tanto podría ser necesario o conveniente programar la aplicación en *Swift*

2.6 Requisitos funcionales

Una vez dada una visión global del producto, vamos a comentar los requisitos funcionales que debe cumplir. Atendiendo a las reuniones con el director del proyecto, se pueden dividir en tres grupos: cálculo de rutas ecológicas, información al usuario, seguimiento del perfil del conductor.

Cálculo de rutas ecológicas

- **RF01 - Introducir datos de la ruta**. La aplicación debe permitir al usuario introducir el punto inicial y el punto final de la ruta.
  - **RF01.1 - Seleccionar en el mapa**. El usuario podrá seleccionar un punto presionando la posición en el mapa hasta que aparezca una marca. Si no hay punto de inicio, se colocará una marca de inicio. Si hay punto de inicio, se colocará una marca de destino.
  - **RF01.2 - Buscador de calles**. Existirá un buscador con dos entradas de texto, una para el punto inicial y otra para el punto final. Dependiendo de donde se introduzca el texto, se colocará marca inicial o final.
- **RF02 - Modificar datos de la ruta**. La aplicación debe permitir al usuario modificar el punto inicial o el punto final en un momento dado.
  - **RF02.1 - Selección de marca**. El usuario podrá seleccionar una marca y desplazarla a una nueva posición.
  - **RF02.2 - Buscador de calles**. El usuario insertará una calle en la entrada de texto oportuna y la marca respectiva se trasladará a la nueva posición.
  - **RF02.3 - Seleccionar en el mapa**. El usuario presionará una nueva posición en el mapa hasta que aparezca en la nueva posición marcada.
- **RF03 - Calcular ruta**. Cuando el punto inicial y el punto final se especifiquen la aplicación deberá conectarse al servicio web automáticamente y obtener las rutas calculadas en formato *JSON*.
- **RF04 - Manejo de errores**. Cuando se produzca un error en el cálculo de rutas, se mostrará un mensaje al usuario informándole del problema ocurrido.
  - **RF04.1 - Error en el servidor**. Informar al usuario en caso de que el servicio no esté disponible.
  - **RF04.2 - Error en el cálculo de la ruta**. Informar al usuario si el servicio no puede calcular una ruta entre los puntos insertados por el usuario.
  - **RF04.3 - Error de conexión**. Informar al usuario en caso de que no tenga conexión a Internet.

Información al usuario

- **RF05 - Mostrar rutas calculadas**. Una vez responda el servidor con toda la información relativa a las rutas, se debe mostrar al usuario de distintas formas.
  - **RF05.1 - Trazado de las rutas**. Se pintará en el mapa el trazado de las rutas devueltas cada una con un color, siendo la más ecológica pintada de color verde y la menos ecológica pintada de color negro. Las rutas intermedias tendrán un color verde oliva.
  - **RF05.2 - Datos de las rutas**. Habrá una tabla con el nombre que identifica a cada una de la rutas, la duración de la ruta y el color que identifica a esta ruta, siguiendo la misma metodología de colores explicada en el punto anterior. Cada fila de la tabla se podrá seleccionar.
- **RF06 - Mostrar detalles de la ruta**. Al seleccionar una ruta aparecerá una vista con toda la información relativa a la ruta seleccionada, duración, fecha y emisión de gases nocivos. Existirá un botón en esta vista que abra el itinerario de la ruta
  - **RF06.1 - Itinerario de la ruta**. La aplicación debe ofrecer al usuario la posibilidad de ver en formato textual los pasos que debe seguir para completar la ruta correctamente.
  - **RF06.2 - Ocultar información**. Debe existir un botón para volver a la interfaz donde se muestra únicamente el mapa, para que el usuario pueda insertar una ruta nueva.

Seguimiento del perfil del conductor

- **RF07 - Autenticación**. El usuario podrá iniciar sesión en la aplicación con su correo electrónico y contraseña. Siempre puede iniciar sesión con otro usuario para cambiar el perfil de conducción después de haber iniciado sesión anteriormente .
  - **RF07.1 - Autenticación con éxito**. El servidor web devolverá un token de autenticación que deberá ser almacanedao y utilizado en las siguientes peticiones al servidor.
  - **RF07.2 - Error en el servidor**. Se informará al usuario en caso de que el servicio de autenticación no estuviera disponible.
  - **RF07.3 - Error de conexión**. Se informará al usuario en caso de que no tenga conexión a Internet.
- **RF08 - Peticiones al servidor**. En caso de no haber iniciado sesión, se podrá utilizar la aplicación sin impedimentos. Si se inicia sesión previamente, se añadirá a una cabecera AuthenticationToken a la petición *HTTP* con el token devuelto por el servidor.

2.7 Requisitos no funcionales

- **RNF01 - Lenguaje**. El lenguaje utilizado debe ser propio de *Apple*: *Objective-C* o *Swift*.
- **RNF02 - Usabilidad**. La aplicación debe poder visualizarse correctamente en todos los dispositivos móviles y tabletas de *Apple*.

2.8 Casos de uso

Una vez definidos los requisitos funcionales y no funcionales, vamos a elaborar los diferentes casos de uso de nuestra aplicación. Con ellos se pretende dar una visión más detallada de las funciones a implementar. A continuación se muestra un diagrama de caso de uso que comprende los requisitos funcionales **RF01, RF02** y **RF07**. Posteriormente expondremos otro diagrama de caso de uso comentando otras funcionalidades del sistema.

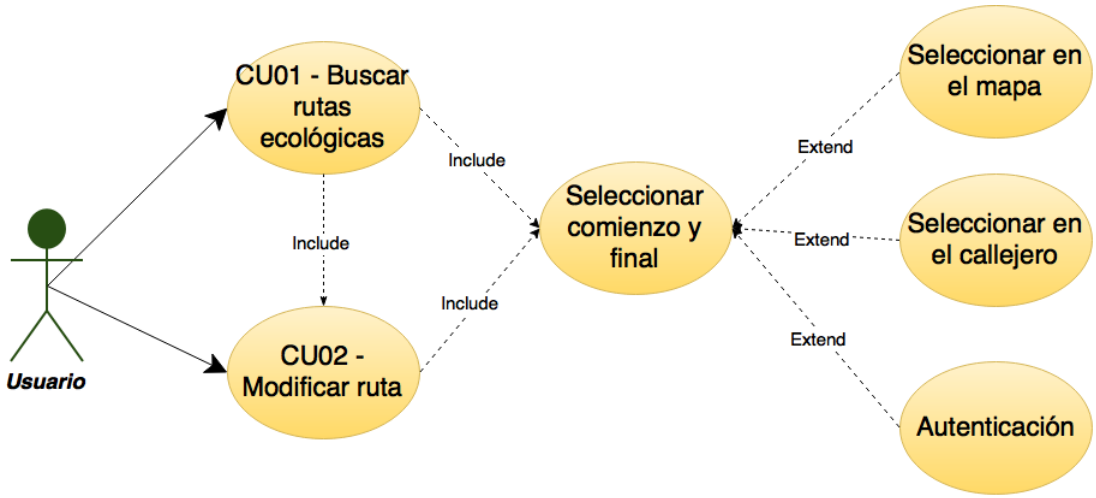

Fig. 2.1: Diagrama de casos de uso conteniendo CU1 y CU2

En la figura 2.1 se muestra un diagrama de casos de uso donde están representados el caso de uso “Buscar rutas ecológicas” (CU01) y “Modificar ruta” (CU02). En cualquiera de las dos situaciones, buscar o modificar ruta, el usuario deberá seleccionar el comienzo y el final de la ruta, a través de un mapa o desde un callejero, si lo desea podrá autenticarse. Vamos a explicar con más detalle cada uno de estos casos de uso y los distintos casos de prueba para dichos casos de uso.

Caso de uso 01 - Buscar rutas ecológicas

Descripción: El usuario podrá buscar rutas ecológicas para llegar al destino deseado.

Precondición: Se deben seleccionar los puntos de inicio y fin de la ruta.

Post-condición: Se mostrará en el mapa las distintas rutas calculadas.

Escenario principal:

1. El usuario selecciona desde el mapa o el callejero el punto de inicio.
2. El usuario selecciona desde el mapa o el callejero el punto final.
3. El sistema realiza una petición al servidor con los datos insertados.
4. Los datos devueltos se muestran en el mapa.

Escenarios alternativos:

4. No conecta con el servidor por un problema:
    a) El sistema muestra un mensaje con el error.
    b) Vuelta al paso 1.

---

2. El callejero no encuentra la calle:
    a) El usuario cancela la búsqueda por callejero.
    b) Vuelta al paso 1.

Casos de prueba:

- **CP01 - Buscar rutas ecológicas**. El usuario selecciona un punto de inicio desde el mapa o el callejero, realiza la misma acción para el punto final de la ruta. El sistema se conecta con el servidor enviando los parámetros de la ruta, se recibe una respuesta y se pinta en el mapa.
- **CP02 - Error en la conexión**. Desconectar el acceso Internet, seleccionar el punto inicial en el mapa o en el callejero, realizar la misma acción para el punto final de la ruta. El sistema muestra un error avisando del error ocurrido.

Caso de uso CU02 - Modificar ruta

Descripción: El usuario podrá modificar el punto de inicio y el punto final seleccionados previamente.

Precondición: Los puntos inicial y final deben haber sido seleccionados previamente.

Post-condición: Se actualizarán los puntos inicial y final a los nuevos valores introducidos por el usuario.

Escenario principal:

1. El usuario modifica el punto inicial, arrastrando la marca existente en el mapa, o seleccionando un punto desde el callejero.
2. El usuario modifica el punto final, arrastrando la marca existente en el mapa, seleccionando un punto desde el callejero, o manteniendo pulsado el dedo en nuevo lugar en el mapa.
3. El sistema realiza una petición al servidor con los datos insertados.
4. Los datos devueltos se muestran en el mapa.

Escenarios alternativos:

2. El usuario cancela el cambio de posición del punto inicial:
    a) La marca se deja en la misma posición.

Casos de prueba:

- **CP03 - Modificar ruta**. El usuario modifica la posición del punto de inicio arrastrando la marca en el mapa o desde el callejero, realiza la misma acción para el punto final de la ruta. El sistema se conecta con el servidor enviando los parámetros de la ruta, se recibe una respuesta y se pinta en el mapa.
- **CP04 - Cancelar modificación**. El usuario elecciona una marca, cuando esté disponible para arrastrarla, el usuario la suelta. El sistema no realiza ninguna acción puesto que la posición es la misma.

Para completar todos los casos de uso del sistema mostramos el siguiente diagrama donde aparece el caso de uso *CU01* y las relaciones de éste con otros.

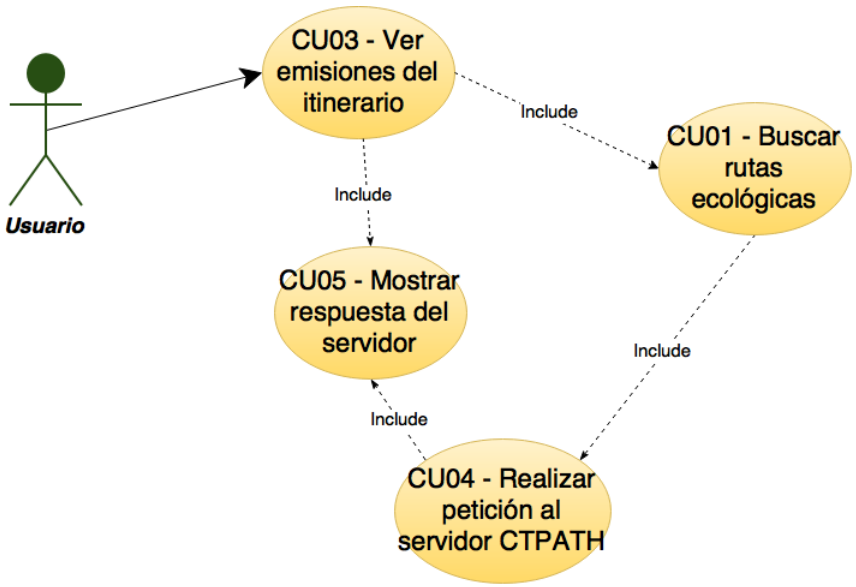

Fig. 2.2: Diagrama de casos de uso incluyendo CU01, CU03, CU04, CU05

En este diagrama se pretende explicar que para que el usuario pueda realizar el caso de uso CU03 - *Ver emisiones del itinerario* debe primero buscar el itinerario implicado. Ya hemos visto en los casos de uso anteriores cómo se ejecuta éste. La novedad del diagrama anterior es la introducción de nuevos caso de usos relacionados con el caso de uso CU01. Tras buscar la ruta el usuario, la aplicación debe realizar la petición al servidor CTPATH y mostrar la respuesta recibida para que el usuario pueda ver las emisiones calculadas. Vamos a explicar con detalle cada uno de los casos de uso implicados en el diagrama anterior.

Caso de uso CU03 - Ver emisiones del itinerario

Descripción: El usuario podrá comprobar las emisiones de gases nocivos que emite la ruta elegida.

Precondición: Se ha realizado una búsqueda de itinerarios.

Post-condición: Aparece una vista con los detalles de una ruta incluyendo sus emisiones.

Escenario principal:

1. El usuario selecciona una ruta de la tabla mostrada.

2. Se abre una pestaña con los datos de la ruta pudiendose ver las emisiones de la ruta.

Casos de prueba:

- **CP05 - Ver emisiones del itinerario**. El usuario realiza una búsqueda de rutas, se obtiene una lista con las rutas calculadas. El usuario selecciona una de ellas. Tras esto, aparece una ventana con los detalles de la ruta y sus emisiones.

Caso de uso CU04 - Realizar petición al servidor CTPATH

Descripción: La aplicación deberá conectarse al servicio que realiza los cálculos de ruta.

Precondición: El usuario solicita una búsqueda de rutas.

Post-condición: Se obtienen las rutas calculadas por el servidor

---

Escenario principal:

1. Se obtienen los parámetros de la búsqueda del usuario:
   - Coordenadas de los puntos inicial y final (parámetros fromPlace y toPlace)
   - Fecha y hora actual (parámetros time y date)
   - Tipo de vehículo (En esta versión sólo es para coche. Parámetro mode)
2. Se crea la URL para realizar la petición, por ejemplo: `https://mallba3.lcc.uma.es/otp/routers/default/plan?fromPlace=36.712,-4.485&toPlace=36.707,-4.434&time=9:18pm&date=06-27-2016&mode=CAR`
3. Se ejecuta la petición.
4. Se obtiene la respuesta del servidor.

Escenarios alternativos:

4. Se obtiene un error en la petición del servidor:
   a) Se muestra un mensaje de error.
   b) Vuelta al paso 1

Casos de prueba:

- **CP06 - Realizar petición**. La aplicación obtiene los parámetros de la ruta, construye la URL de la petición y la realiza. El servidor devuelve los datos de las rutas calculadas.
- **CP07 - Error en la petición**. La aplicación obtiene los parámetros de la ruta, construye la URL. Sin conexión a Internet se realiza la petición. El sistema muestra un mensaje de error.

Caso de uso CU05 - Mostrar respuesta del servidor

Descripción: La aplicación deberá mostrar la información devuelta por el servidor.

Precondición: El servidor ha respondido con los datos de las rutas.

Post-condición: Se mostrará la información de cada una de las rutas.

Escenario principal:

1. Se obtienen los datos de las rutas calculadas por el servidor.
2. Se recoge la información de cada ruta en una estructura de datos:

3. Cuando se seleccione una ruta, se accederá a sus datos y se mostrarán en una ventana.

Casos de prueba:

- **CP08 - Mostrar datos**. La aplicación obtiene la información hallada por el servidor, la procesa y la muestra cuando el usuario selecciona una ruta.

Capítulo 3

Diseño de la aplicación

Tras haber finalizado la fase de análisis del problema con la elaboración del documento de especificación de requisitos, continuamos con la fase de *modelado* de la aplicación. Comenzaremos explicando la arquitectura *cliente-servidor* y el patrón de arquitectura *Modelo*, *Vista*, *Controlador*, tras esto, mostraremos el *diagrama de clases* de nuestro proyecto de una forma general, y explicaremos cada una de las clases implementadas.

3.1 Arquitectura del sistema

En este proyecto hemos implementado dos tipos de arquitecturas diferentes, por un lado una arquitectura cliente-servidor, donde el cliente es la aplicación *iOS* y el servidor está proporcionado por el proyecto *CTPATH*, y por otro lado, una arquitectura Modelo-Vista-Controlador para la correcta estructuración de las clases en el desarrollo de la aplicación.

3.1.1 Arquitectura Cliente-Servidor

La arquitectura cliente-servidor implementada comunica al cliente y al servidor a través de Internet, con la característica de que esta comunicación solo se produce cuando el cliente móvil realiza una petición *HTTP* al servidor y éste le responde. Por tanto, el servidor sólo puede recibir peticiones de la aplicación, y responder con los datos solicitados, pero en ningún momento se iniciará la comunicación por parte del servidor. En la siguiente imagen se muestra la comunicación entre la aplicación y el servidor.

Como se puede observar en la figura 3.1, el cliente móvil empieza la comunicación a través de una petición *HTTP* donde se indican los siguientes parámetros: la URL del recurso, el punto inicial, el punto final, la fecha y hora del viaje, y el tipo de vehículo. El servidor recibe estos parámetros, realiza los cálculos y responde con un texto en formato *JSON* con la información de las rutas. Esta respuesta se procesa en la aplicación y se muestra la información al usuario.

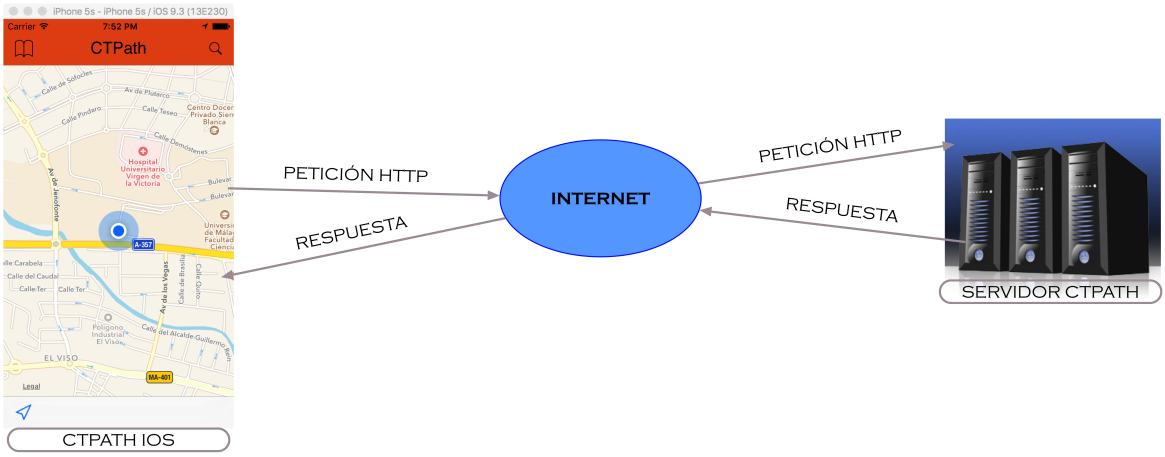

Fig. 3.1: Arquitectura del sistema

3.1.2 Arquitectura Modelo, Vista, Controlador

La mayoría de las aplicaciones para *iOS* siguen el patrón de diseño Modelo, Vista, Controlador (MVC) debido a que *Apple* recomienda utilizar este patrón. Esta forma de organizar las clases del proyecto software asigna a cada objeto un rol: Modelo, Vista o Controlador y la forma en la que estos objetos se van a comunicar. Expliquemos la función de cada uno de estos roles:

Modelo. Este rol es el encargado de definir y encapsular la información que se utiliza, recibe o produce en la aplicación. En nuestro caso, el modelo será las distintas rutas que el servidor calcule.

Vista. Es el conjunto de elementos gráficos que el usuario puede ver en la aplicación, como puede ser un botón, o una caja de texto. Estos elementos son los que permiten al usuario interaccionar con la aplicación.

Controlador. Es el encargado de realizar cambios en el modelo cuando el usuario realiza una acción en las vistas, y al mismo tiempo, cuando se produce un cambio en el modelo, es el encargado de notificar a la vista el cambio producido.

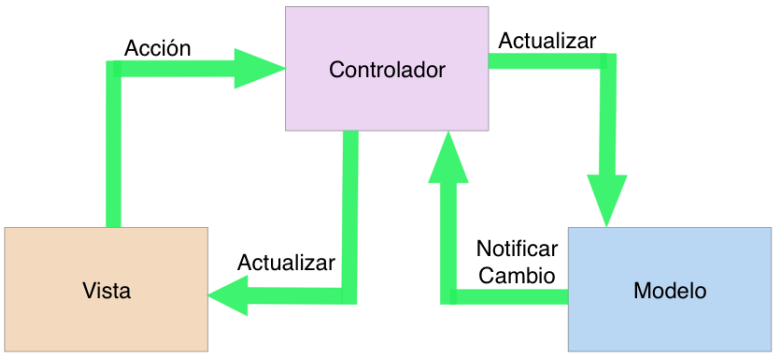

Fig. 3.2: Patrón MVC

En la figura 3.2 se puede observar que el controlador hace de intermediario entre la vista y el modelo, como ya habíamos comentado. Realizar una actualización de la vista o modelo directamente sin comunicar al controlador está prohibido en este patrón de diseño, todo cambio producido debe pasar por el controlador.

A continuación, en la figura 3.3 se muestra cómo se ha organizado el patrón MVC en nuestra aplicación. Normalmente, por cada controlador, hay una vista asociada a ese controlador, el resto de vistas marcadas con un asterisco (*) son vistas auxiliares que han sido creadas para ayudar a mostrar la información al usuario de manera apropiada.

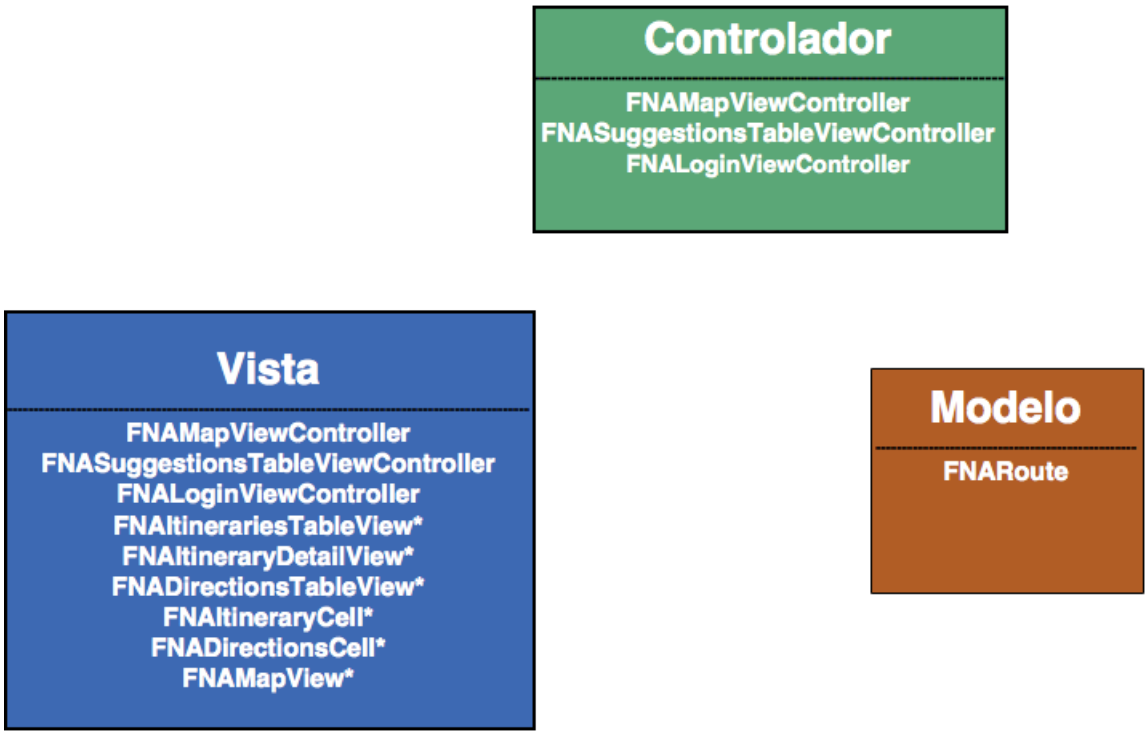

Fig. 3.3: Implementación del patrón MVC

3.2 Diagrama de clases

A continuación vamos a exponer el **diagrama de clases** de nuestro proyecto, teniendo en cuenta la figura 3.3, donde comentamos que cada controlador tiene asociada una vista, de tal manera que los atributos de los controladores que presenten el signo de exclamación (!) especificará que ese atributo es un elemento gráfico de la vista asociada. De esta manera, simplificamos el diagrama de clases para hacerlo más inteligible. Para mostrar el diagrama de clases completo, se muestra en la imagen cada una de las clases sin atributos y métodos. Tras comentar de forma general el diagrama, continuaremos con la explicación de cada una de las clases implementadas.

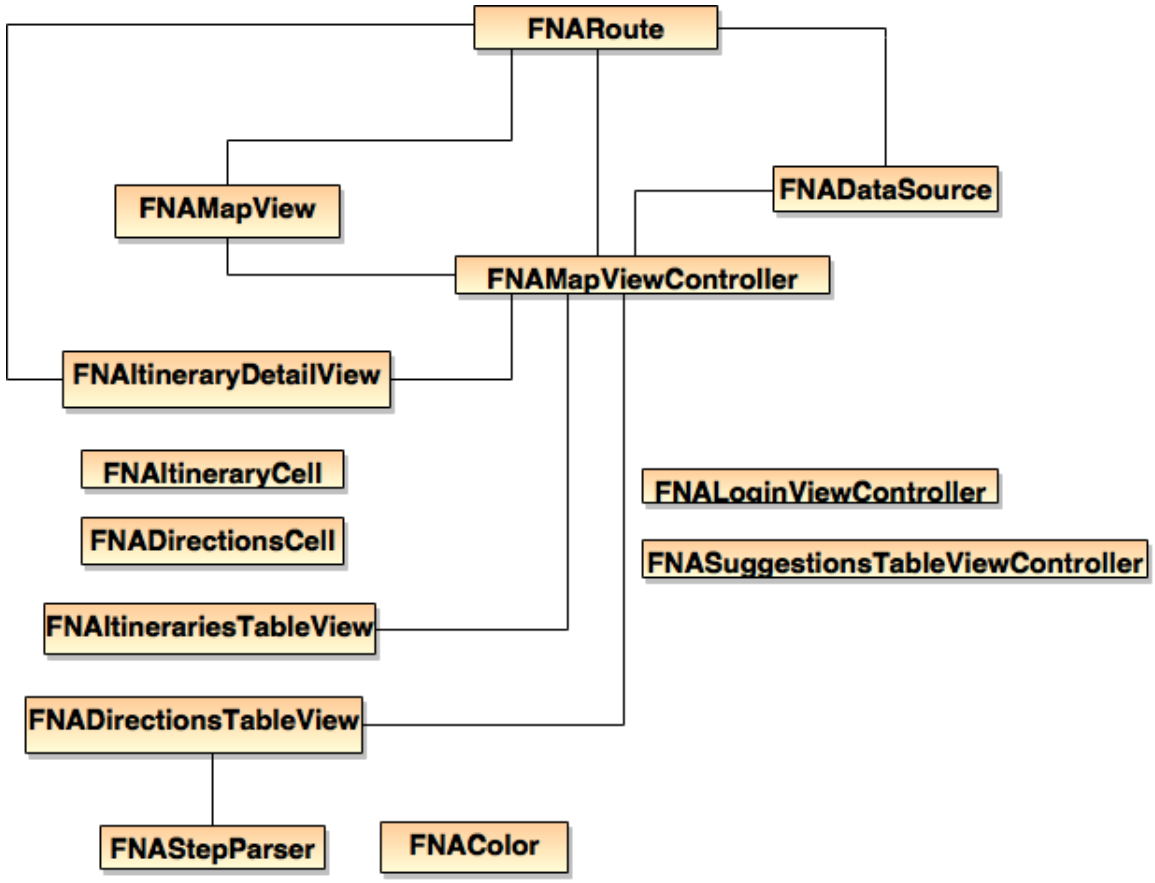

Fig. 3.4: Diagrama de clases de la aplicación

En la parte central del diagrama se encuentra el controlador *FNAMapViewController*, el cual tiene la mayoría del control sobre las vistas con las que puede interaccionar el usuario. Estas vistas aparecen a la izquierda del controlador principal en el diagrama de clases. Son *FNAMapView*, *FNAItineraryDetailView*, *FNAItineraryCell*, *FNADirectionsCell*, *FNAItinerariesTableView* y *FNADirectionsTableView*. A estas vistas se suman las propias de cada uno de los controladores, la vista *FNAMapViewController* del controlador principal y las que se encuentran a la derecha en el diagrama de clases: *FNALoginViewController* y *FNASuggestionsTableViewController*, propias de los controladores con el mismo nombre. En la parte superior, podemos ver nuestro modelo *FNARoute* donde se aloja la información devuelta por el servidor. A parte de las clases implicadas en el patrón MVC, en el diagrama también se incluyen otras clases auxiliares que hemos utilizado como *FNAStepParser*, *FNAColor* y *FNADataSource*.

Modelo

- **FNARoute**. Esta clase almacena los datos que provienen del servicio web REST en un objeto del tipo NSDictionary, ya que es un objeto muy similar a un texto en formato JSON. Ambos tienen el formato (clave, valor). Además de almacenar los datos, implementa los métodos necesarios para acceder a ellos. El contenido de este objeto cambia cada vez que el usuario solicita un nuevo cálculo de rutas.

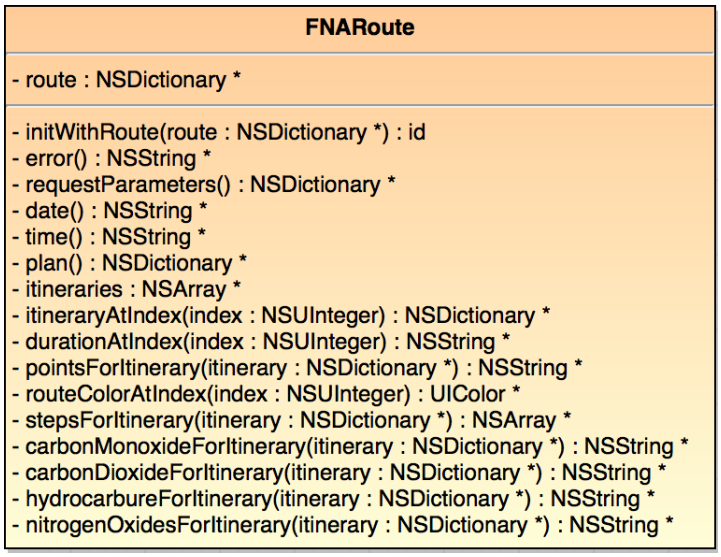

Fig. 3.5: FNARoute

Vista.

Vamos a comentar las vistas que no pertenecen intrínsicamente a los controladores, es decir, las vistas marcadas con un asterisco en el recuadro ”Vista” de la figura 3.3.

- **FNAItinerariesTableView**. Esta vista se utiliza para implementar el requisito funcional **RF05.2**. Es una tabla donde se muestra el contenido especificado en este requisito.

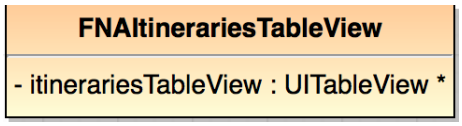

Fig. 3.6: FNAItinerariesTableView

- **FNAItineraryCell**. Esta clase hereda de UITableViewCell y no es más que una celda personalizada que posteriormente formará parte de la tabla FNAItinerariesTableView, no obstante, puede servir para cualquier otra tabla.

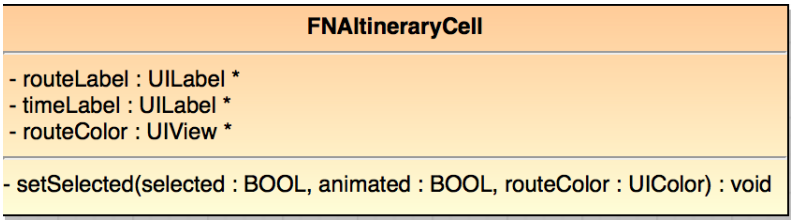

Fig. 3.7: FNAItineraryCell

- **FNAItineraryDetailView**. Esta vista es utilizada para mostrar todos la información relativa a una ruta específica, cumpliendo así el requisito funcional **RF06.1**.

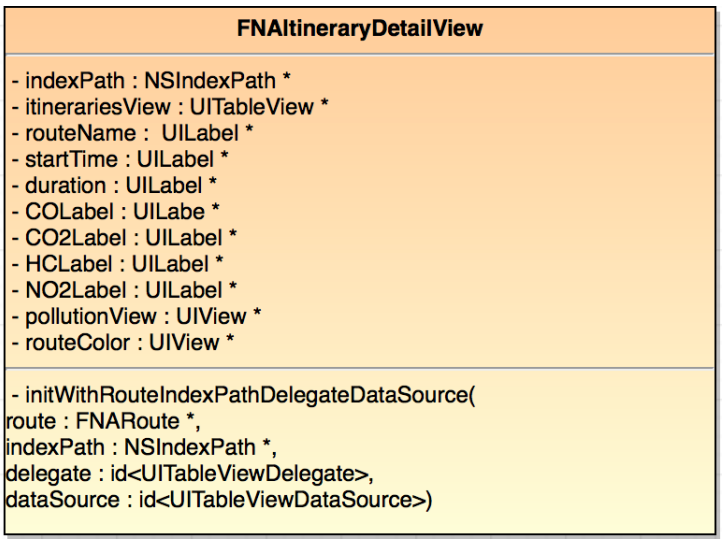

Fig. 3.8: FNAItineraryDetailView

- **FNADirectionsTableView**. Esta vista es la encargada de mostrar al usuario los pasos que debe seguir para completar la ruta previamente seleccionada. Con esta vista mostrarmos al usuario la información comentada en el requisito funcional **RF06.1**.

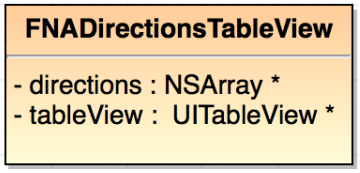

Fig. 3.9: FNADirectionsTableView

- **FNADirectionsCell**. Es la celda personalizada de la tabla anterior, ya que ninguna de las celdas que Cocoa tiene por defecto nos es de utilidad y usando una celda personalizada tenemos más libertad de cambio.

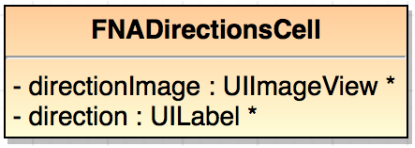

Fig. 3.10: FNADirectionsCell

- **FNAMapView**. Esta clase es una extensión de UIMapView que se crea para adaptar el mapa propio de Apple a las necesidades de nuestro proyecto.

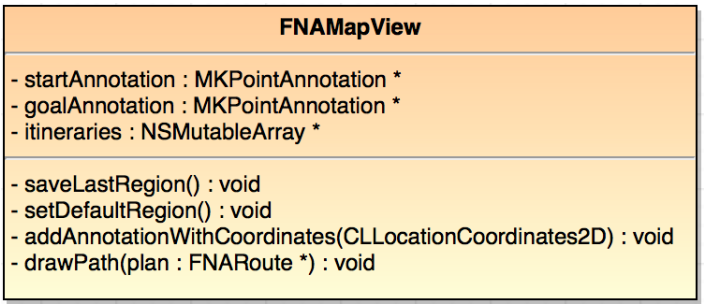

Fig. 3.11: FNAMapView

Controlador

- **FNASuggestionsTableViewController**. Este controlador es el encargado de implementar el requisito funcional **RF01.2**, una vez que el usuario selecciona la calle, éste controlador se comunica con FNAMapViewController para colocar una marca en el mapa. De esta manera el usuario podrá buscar por el nombre de la calle, si no conoce la posición en el mapa.

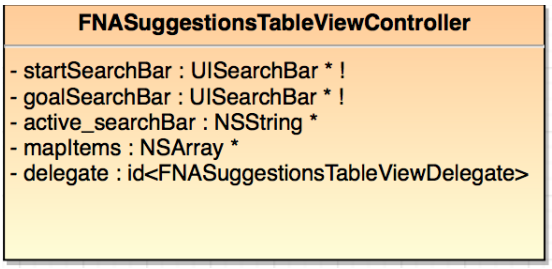

Fig. 3.12: FNASuggestionsTableViewController

- **FNALoginViewController**. Este controlador implementa el requisito funcional **RF07** y envía el token de autenticación al controlador FNAMapViewController a través de un protocolo.

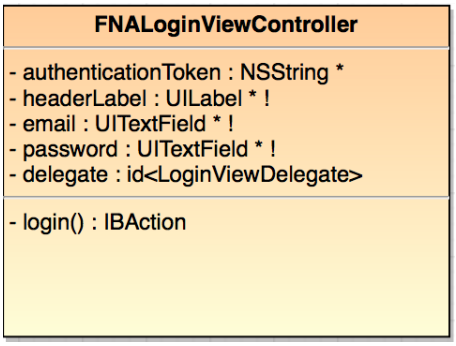

Fig. 3.13: FNALoginViewController

- **FNAMapViewController**. Debido a la naturaleza del proyecto, las acciones del usuario se producen en un mapa, dado que este controlador contiene ese mapa, proporciona la mayoría de la funcionalidad de la aplicación. Este controlador será el encargado de comunicar el modelo con las vistas comentadas anteriormente, y a su vez, presentará en pantalla el controlador de inicio de sesión y el controlador para la sugerencia de calles explicados anteriormente. Implementa los requisitos **RF01.1, RF02, RF03, RF04, RF05 y RF06**.

Fig. 3.14: FNAMapViewController

Clases auxiliares

- **FNAStepParser**. Esta clase se crea para interpretar la información de las direcciones que se deben seguir para completr una ruta. Es una clase auxiliar que nos ayuda implementando métodos de acceso a las direcciones de la ruta devueltas por el servicio.

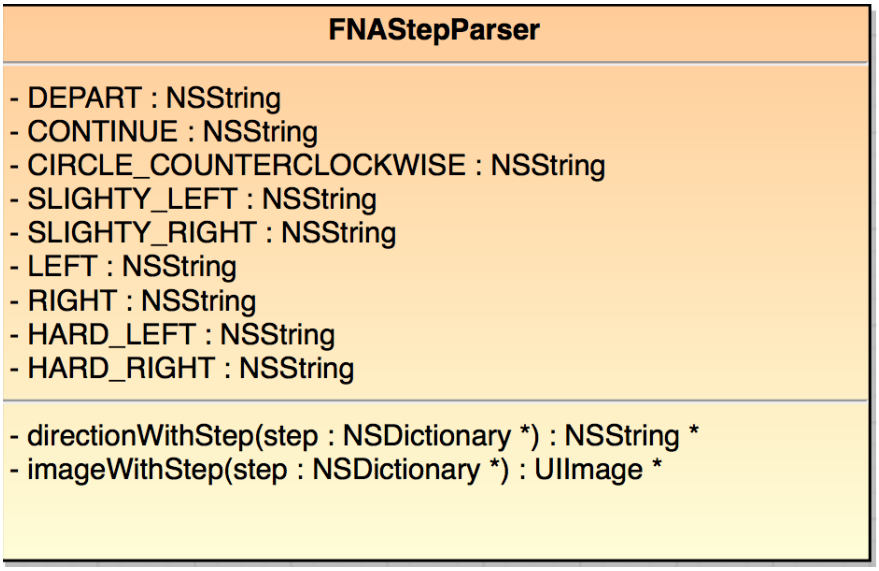

Fig. 3.15: FNAStepParser

- **FNAColor**. Esta clase extiende de UIColor implementando métodos auxiliares que retornan algunos colores. Se crea debido a que se necesitará utilizar estos colores en muchas ocasiones, y por tanto, para no tener que ”crear” estos colores continuamente, se crea esta clase con los métodos que los ”crean”.

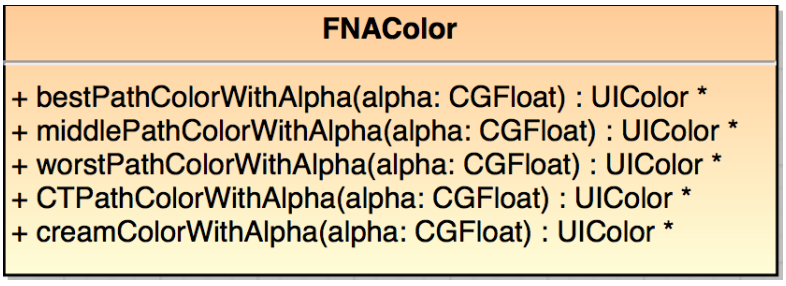

Fig. 3.16: FNAColor

- **FNADataSource**. Debido a que continuamente mostraremos al usuario información en tablas, éstas necesitarán una clase que ejerza de “DataSource” (Una clase que rellene el contenido de cada celda de la tabla). Implementaremos esta clase para unificar cada uno de los DataSource de cada tabla, de este modo, con una única clase, instanciamos las tablas de nuestra aplicación.

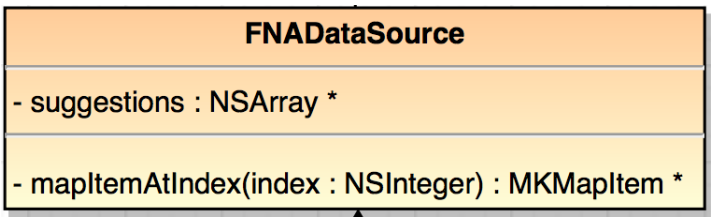

Fig. 3.17: FNADataSource

Capítulo 4

Diseño de la interfaz de usuario

En esta fase del problema explicaremos como se construirán las distintas interfaces gráficas mostrando la disposición de cada uno de los elementos gráficos a partir de *mockups* diseñados con el software *Balsamiq* [8].

Ya que la aplicación se centra en el uso de un mapa, buscamos una interfaz de usuario atractiva y a la vez sencilla de utilizar, aportando al usuario la información exacta y de manera concisa con el mínimo movimiento entre ventanas. Esto hará que el usuario no se pierda entre los elementos gráficos de la interfaz y entienda rápidamente el funcionamiento de la aplicación.

Debido a que tenemos tres controladores, vamos a mostrar el diseño de cada una de las vistas asociadas a estos controladores.

4.1 FNAMapViewController

Veamos el conjunto de vistas asociadas a este controlador, su diseño, y cómo van apareciendo unas y otras dependiendo de las acciones que realice el usuario sobre los elementos gráficos.

4.1.1 Ventana inicial

En la figura 4.1 se muestra la primera pantalla que aparece cuando inicia la aplicación. La disposición de los elementos gráficos es la siguiente.

Se puede observar un mapa en la parte central, en el que el usuario podrá seleccionar los puntos para calcular las rutas. Cada vez que se cierre la aplicación, quedará almacenada la región que estaba a la vista del usuario, y cuando se vuelva a iniciar la aplicación, el mapa se situará en esta región. Por defecto, aparece la región en la ciudad de Málaga, al ser éste el caso de estudio.

En la parte inferior, se observa un botón que nos lleva a la posición del usuario en caso de que nos dé permiso de geolocalización.

En la parte superior, podemos ver tres elementos gráficos: El elemento de la izquierda nos abrirá el controlador de inicio de sesión, en medio vemos el nombre de

la aplicación, y a la derecha tenemos un botón que al pulsarlo muestra el controlador para sugerir calles al usuario.

Fig. 4.1: Pantalla inicial

4.1.2 Mapa con puntos de inicio y fin

En la figura 4.2 podemos ver el resultado de insertar el punto inicial y final de nuestra ruta en el mapa.

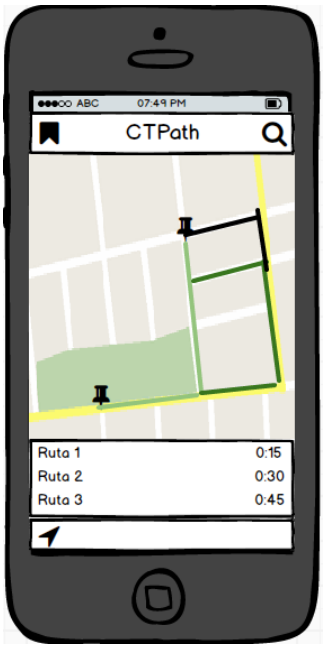

Fig. 4.2: Mapa con puntos de inicio y fin

El objetivo es calcular los itinerarios existentes entre estos dos puntos, y que automáticamente se pinten estas rutas con distintos colores en el mapa y aparezca una tabla con la información de cada una de ellas.

4.1.3 Vista detalle de una ruta

Una vez que el usuario seleccione la ruta que quiere tomar, aparecerá una vista con toda la información relativa a ella. En esta vista "detalle" contendrá la cantidad de gases nocivos que se emiten al utilizar este itinerario, la duración y la fecha como se puede ver en la figura 4.3.

Fig. 4.3: Vista detalle de una ruta

Además, desde esta vista, se podrá seleccionar otro itinerario diferente y por tanto, se modificarán los datos existentes mostrando los de la ruta seleccionada.

4.1.4 Instrucciones del itinerario

También aparecerá en el menú de abajo, dos nuevos elementos gráficos: uno para ocultar esta vista detalle volviendose a mostrar el mapa, y un botón con una flecha que abrirá una tabla con las direcciones que hay que seguir para completar el itinerario (Figura 4.4).

Fig. 4.4: Instrucciones del itinerario

4.2 FNASuggestionsTableViewController

Veamos cual es la vista asociada a este controlador y la forma en la que se produce la interacción con el usuario y el resto de controladores.

4.2.1 Sugerencia de calles

Esta vista mostrada en la figura 4.5 es la encargada de proporcionar los elementos gráficos para que el usuario pueda insertar texto con el nombre de la calle que desea que sea el inicio o el fin de su ruta. Contiene dos cajas de texto y una tabla. Tras escribir en las cajas de texto, aparecerá una lista de sugerencias con el nombre de las calles que son parecidas al texto introducido por el usuario, con el fin de ayudarle en su elección. En cualquier momento se puede cancelar esta acción, haciendo click en el botón “cancelar”, volviendo a mostrar el controlador anterior, es decir, el mapa.

Una vez que el usuario seleccione una calle de la lista, esta vista desaparecerá y se insertará una marca en el mapa.

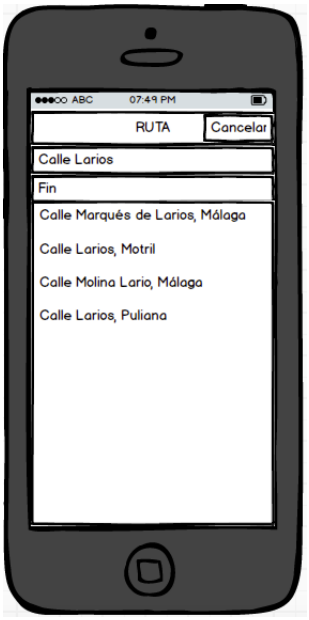

Fig. 4.5: Sugerencia de calles

4.3 FNALoginViewController

Como indica el nombre de esta vista, se encarga de proporcionar los elementos gráficos necesarios para que el usuario pueda iniciar sesión en la aplicación. Veamos como está estructurada.

4.3.1 Vista de inicio de sesión

Fig. 4.6: Vista de inicio de sesión

La vista de inicio de sesión está compuesta por dos cajas de texto donde se escribe el correo electrónico y la contraseña del usuario. Justo debajo tenemos un botón para realizar el inicio de sesión una vez rellenos los campos de entrada. Más abajo se muestra un texto donde se explica que el registro se debe realizar desde la web de CTPATH, y el nombre del proyecto. En el menú superior podemos ver un botón llamado “salir” para ocultar esta ventana y hacer aparecer el mapa de nuevo.

Capítulo 5

Validación y pruebas

En este capítulo se explican las distintas pruebas realizadas para validar el software desarrollado, y así comprobar el buen funcionamiento del sistema.

Aunque la mejor forma de validar el software es a través de pruebas unitarias u otras pruebas automáticas, debido a mi desconocimiento en la realización de este tipo de pruebas en el entorno de programación de *Apple* no ha sido posible realizarlas, por lo que sería recomendable para desarrollos futuros integrarlas en el proyecto. No obstante, sí se han realizado otras pruebas durante el desarrollo de la aplicación para comprobar el buen funcionamiento de la misma conforme se iba avanzando en su implementación. Estas pruebas se han realizado con la ayuda de personas ajenas al proyecto a las cuales se les ha pasado un cuestionario guiado a través de la aplicación conforme ésta avanzaba. De esta forma, el software era validado por futuros usuarios y a través de su experiencia se mejoraba la usabilidad y se corregían los errores con los que se encontraban durante el uso de ella.

Estas pruebas se realizaron ejecutando la aplicación en un simulador. A cada usuario se le asignó un dispositivo (Iphone 5, Ipad 2, Ipad Air, Iphone 6 Plus...) para comprobar que el software se ejecutaba correctamente en todos los distintos dispositivos de *Apple*. El usuario debía seguir las acciones que se especifican en el documento de la figura 5.1 y responder con el resultado obtenido (éxito o error) al realizar la acción sugerida, y en su caso, añadir un comentario extra explicando los problemas con los que se encontró en caso de error, o sugerir alguna mejora.

Una vez finalizado el test se recogía el documento, se estudiaban los problemas que había obtenido el usuario y se aplicaban los arreglos oportunos para solventar las incidencias que aparecían en él.

Test Guiado para probar funcionalidad de CTPATH-iOS				
VERSIÓN				
SIMULADOR				
ACCIÓN A REALIZAR	RESULTADO	COMENTARIOS EXTRA		
Inicia la aplicación				
Acepte o rechace los permisos de ubicación				
Al pulsar en la flecha, el mapa se mueve su ubicación				
Mueve el mapa y cierra la aplicación				
Al abrir la aplicación, se carga la región del mapa anterior				
Aparece una chincheta en el mapa cuando mantiene pulsado sobre un punto				
La chincheta es roja (punto de inicio)				
Vuelva a mantener pulsado sobre otro punto del mapa				
En este caso la chincheta es verde (punto final)				
Automáticamente aparece una tabla de rutas y se pintan las rutas				
Compruebe varias veces con distintos puntos				
Aparecen rutas pintadas en verde claro, oscuro y negro				
El número de colores de la tabla coincide con el nº de colores de las rutas				
Al pulsar en mostrar se abre un detalle de la ruta seleccionada				
Al pulsar en ocultar se oculta la vista detalle				
Al pulsar en mostrar de nuevo, aparecen las rutas anteriores				
Desde la vista de detalle, selecciona otra ruta diferente y los datos cambian				
Abajo a la derecha aparece una flecha				
Al hacer click, aparece una ventana con las direcciones que hay que seguir				
Aparece el botón ocultar				
Al pulsar el botón ocultar se muestra la vista detallle				
Selecciona un nuevo punto en el mapa, se cambian los datos de la tabla				
Al hacer click sobre la lupa se abre una ventana para escribir datos				
Al insertar texto, se sugieren diferentes calles				
Inserta una calle conocida por ti en el punto inicial				
Aparece en las sugerencias, seleccionala				
Aparece una marca roja en el mapa				
Vuelve a abrir las sugerencias				
Inserta una calle conocida por ti en el punto final				
Aparece en las sugerencias, seleccionala				
Aparece una marca verde en el mapa y se calculan las rutas				
Se puede acceder a estas rutas como anteriormente				
Haz click en el libro				
Inicia sesión con un nombre y contraseña equivocadas				
Inicia sesión con el usuario que te voy a dar				
Vuelve a seleccionar punto de inicio y fin				

Fig. 5.1: Ejemplo prueba: Test Final de la aplicación

Como hemos comentado al principio, es muy recomendable realizar pruebas unitarias en un futuro para verificar la correcta implementación del software de manera automática. No obstante, las pruebas realizadas han servido para corregir algunos errores y mejorar la usabilidad del producto, gracias a las respuestas dadas por los usuarios.

Capítulo 6

Conclusiones y trabajo futuro

Para finalizar esta memoria, comentaremos las conclusiones que extraemos de todo el proceso de realización de este trabajo de fin de grado, y hablaremos sobre posibles mejoras, y trabajos futuros a los que puede dar lugar este trabajo de fin de grado.

6.1 Conclusiones

En este documento se han presentado las distintas fases por las que ha ido avanzando el proyecto. Comenzamos con una introducción y contextualización del proyecto, comentando la importancia de la tecnología en aspectos tan actuales como el cuidado del medioambiente, y viendo, cómo a pesar de lo avanzada que está la sociedad actualmente, siempre queda hueco para nuevas innovaciones que ayudan a una mejora de la misma, en este caso, a través del proyecto CTPATH.

Tras situarnos en el contexto, se han especificado los distintos requisitos que debe cumplir nuestra aplicación, los cuales, se basan en el proyecto inicial, y por ello, el crecimiento de esta aplicación en términos funcionales, siempre dependerá del crecimiento del proyecto CTPATH.

En la fase de modelado y diseño hemos podido observar dos tipos de arquitecturas dentro de una misma aplicación.

Arquitectura Cliente-Servidor, para dar soporte al cálculo de rutas, esto implica que la mayor parte del trabajo computacional se realiza en servidor, y a través de peticiones HTTP REST se recibe la información calculada. Y por otro lado la arquitectura Modelo, Vista, Controlador, utilizada para estructurar internamente el comportamiento de la aplicación, y la comunicación entre las distintas clases que forman el proyecto.

Una de las fases que más ha ayudado a la hora de desarrollar la aplicación ha sido la fase de diseño, ya que con la ayuda de *mockups* hemos podido tener una visión más específica de la disposición de los elementos gráficos en la interfaz de usuario. Planificar esta disposición antes y no durante el desarrollo ha servido para reducir el

tiempo empleado en diseñar la aplicación, ya que realizar varios *mockups* y modificarlos es siempre más rápido y más cómodo que implementar el diseño y modificarlo durante el desarrollo de la aplicación.

Por último, hemos visto las distintas pruebas de validación realizadas para depurar la aplicación durante su desarrollo y finalización, con el objetivo de detectar errores, y poder aplicar mejoras al sistema.

El haber realizado este trabajo de fin de grado ha supuesto un aumento de mis conocimientos en el desarrollo de software en general, ya que el trabajo realizado en cada una de las fases me ha ayudado a conocer más a fondo las distintas áreas existentes en el desarrollo de un proyecto software. También ha contribuido a aumentar mis conocimientos en el entorno de programación de Apple debido a la necesidad de utilizar el framework de mapas de Cocoa MKMapKit [7].

Con el desarrollo de esta aplicación iOS se ha conseguido aumentar el alcance de uso del proyecto CTPATH, ya que hasta ahora, sólo existía una versión de la aplicación para Android y una versión para web, de tal manera que los usuarios que no disponían de un teléfono móvil o tableta Android, o consideraban que una aplicación web era incómoda para un teléfono móvil, no hacían uso del proyecto CTPATH.

6.2 Trabajo futuro

Implementación de nuevas funcionalidades. Como ya sabemos, con la realización de este trabajo se añade una plataforma nueva al proyecto CTPATH. En la medida en la que aumente la funcionalidad de este proyecto, esta aplicación podrá aumentar su funcionalidad, acorde a éste. Por tanto, un trabajo futuro sería actualizar la aplicación iOS implementando las mejoras que se vayan produciendo en el proyecto padre del cual se basa éste.

CTPATH-iOS en Swift. Debido a haber programado el código en el lenguaje Objective-C por las razones explicadas en el capítulo 2 (la estabilidad de este lenguaje frente al contínuo cambio en la implementación de Swift), en un futuro podría ser necesario su implementación en el lenguaje **Swift** por lo que un posible trabajo futuro podría ser la adaptación de este proyecto a este nuevo lenguaje de Apple.

Referencias bibliográficas

[1] Consecuencias en la salud de la contaminación atmosférica, (fecha de consulta: 25 Junio 2016)
http://www.who.int/phe/health_topics/outdoorair/databases/health_impacts/es/

[2] Cambio climático, (fecha de consulta: 25 Junio 2016),
http://cambioclimaticoglobal.com/que-es-el-cambio-climatico

[3] Gases emitidos por los vehículos a motor, (fecha de consulta: 25 Junio 2016)
https://es.wikipedia.org/wiki/Control_de_emisiones_vehiculares

[4] Christian Cintrano, Daniel H. Stolfi, Jamal Toutouh, Francisco Chicano, and Enrique Alba, CTPATH: A Real World System to Enable Green Transportation by Optimizing Environmentaly Friendly Routing Paths,63–75 (2016)

[5] Diseño técnico del sistema: CTPATH, (fecha de consulta: 25 Junio 2016)
http://maxct.lcc.uma.es/documents/DisenoCTPATH.pdf

[6] Especificación de Requisitos según el estándar de IEEE 830, (fecha de consulta: 25 Junio 2016)
https://www.fdi.ucm.es/profesor/gmendez/docs/is0809/ieee830.pdf

[7] MKMapKit Framework Reference, (fecha de consulta: 25 Junio 2016)
https://developer.apple.com/library/ios/documentation/MapKit/Reference/MapKit_Framework_Reference/MKMapKit

[8] Balsamiq mockups, (fecha de consulta: 25 Junio 2016)
https://balsamiq.com

Apéndice A

Manual de usuario

A continuación se muestra una guía de las distintas vistas de la aplicación explicando cómo utilizar cada uno de los elementos que aparecen en la ventana y su función.

A.1 Splashscreen

Un *Splashscreen* consiste en una interfaz con una imagen y/o nombre de la aplicación que aparece al arrancar la misma y se muestra el tiempo necesario hasta que la aplicación se inicia completamente.

Fig. A.1: Splashscreen con nombre de la aplicación

A.2 Ventana solicitando permiso de geolocalización

En esta ventana, aparece una alerta solicitando al usuario permisos de geolocalización para que en cualquier momento el usuario pueda saber donde se encuentra. Estos pemisos pueden derogarse en cualquier momento desde los ajustes del teléfono.

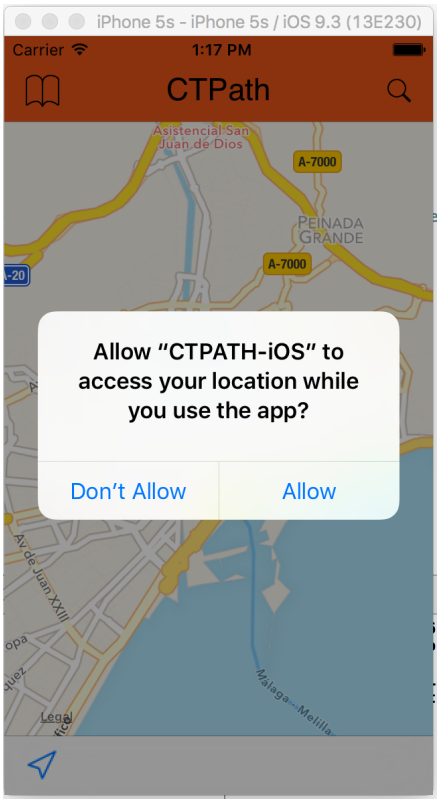

Fig. A.2: Aplicación solicitando permisos de geolocalización

La aceptación o rechazo de los permisos de geolocalización sólo influyen a la hora de mostrar la posición del usuario. Si no están aceptados, no se mostrará ninguna posición.

A.3 Ventana inicial

Una vez iniciada la aplicación aparece la ventana principal, donde podemos ver el mapa donde seleccionaremos los puntos de la ruta, un menú superior con los botones para acceder a los controladores de inicio de sesión y sugerencias de calles, y un menú inferior con un botón para localizar nuestra posición en el mapa, en el caso de que hayamos aceptado los permisos de geolocalización comentados anteriormente.

Fig. A.3: Ventana inicial de la aplicación

A.3.1 Mapa con posición del usuario

Esta interfaz coincide con la anterior, con la única diferencia es que aparece la posición del usuario. Hemos accedido a esta interfaz mediante el botón de geolocalización. Si no es posible ver el punto azul en el mapa, habría que revisar los permisos de geolocalización en los ajustes del teléfono.

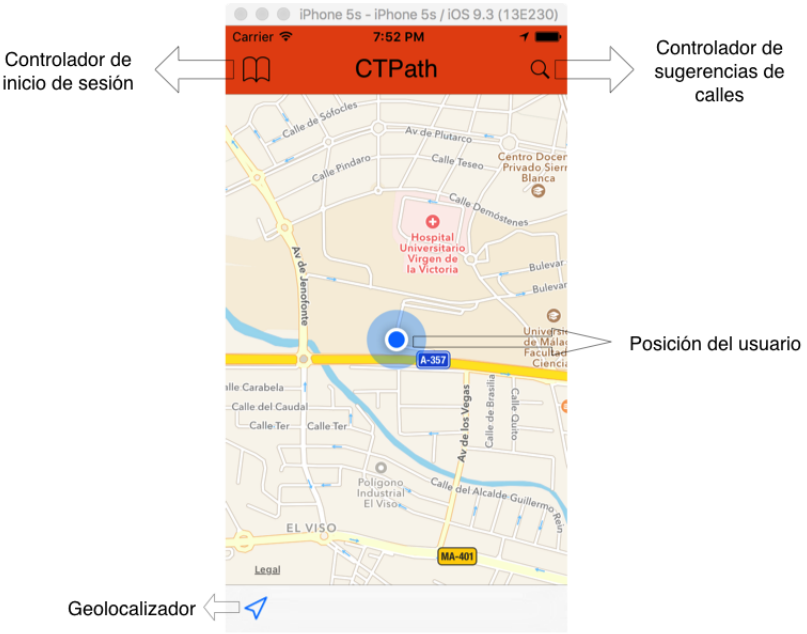

Fig. A.4: Mapa con posición del usuario

A.3.2 Mapa con punto de inicio

Cuando se mantiene el dedo presionado sobre un punto en el mapa aparece una marca en ese punto. Si no tenemos ninguna marca en el mapa, esta marca será roja, indicando el punto inicial, como podemos ver en la siguiente figura.

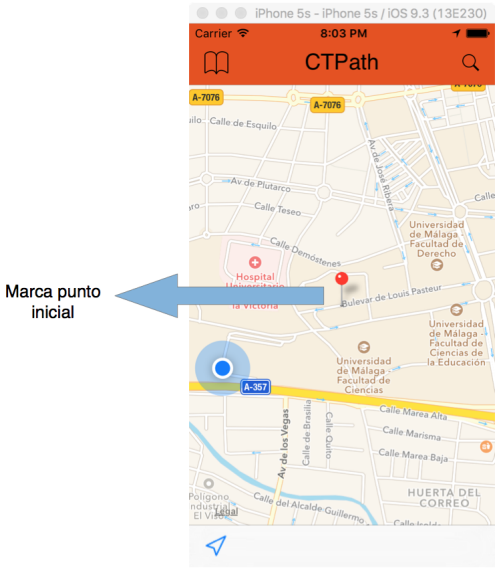

Fig. A.5: Mapa con marca punto inicial

A.3.3 Mapa con puntos de inicio y fin

Por otro lado, si ya tenemos una marca situada en el mapa y mantenemos presionado el dedo en el mapa, aparecerá una marca verde, indicando el punto final, y automáticamente aparecerá una lista de rutas. Si presionamos sobre una ruta nos aparecerá la vista detalle de la ruta seleccionada.

Modificar puntos de inicio y fin

Siempre podremos modificar los puntos seleccionados de tres formas:

- Manteniendo pulsado en otro punto del mapa, en este caso se modifica únicamente el punto final.
- Seleccionando y arrastrando una chincheta del mapa a otro lugar.
- Accediendo al buscador de calles e introduciendo el nombre de la calle en la caja de texto apropiada.

Fig. A.6: Mapa con marca punto inicial y final

A.3.4 Vista detalle de una ruta

En esta vista aparece la información de una ruta, nombre, duración, inicio del viaje, cantidad de gases emitidos por gramos. Además, continúa apareciendo la tabla de rutas del apartado anterior para ver la información de las otras dos rutas proporcionadas por el servidor. Bastaría con seleccionar una de ellas para ver su correspondiente detalle. En la barra inferior aparece un botón para ver las instrucciones a seguir para realizar la ruta.

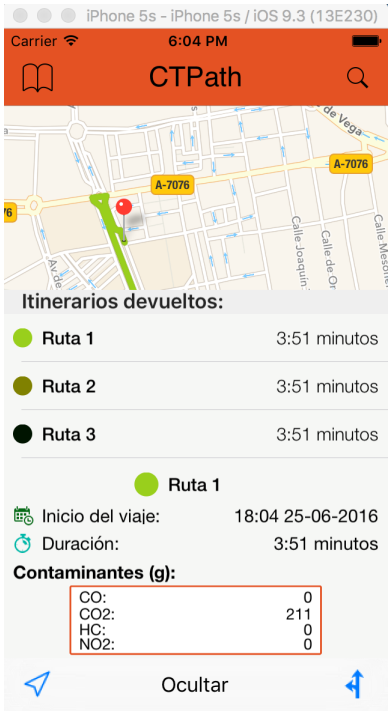

Fig. A.7: Vista detalle de una ruta

A.3.5 Vista con instrucciones de ruta

La vista de la figura A.8 muestra una tabla con cada uno de los movimientos que el conductor debe realizar para alcanzar con éxito el destino marcado, a través de la ruta seleccionada en el paso anterior. Siempre se puede cerrar esta vista pulsando en Ocultar.

Fig. A.8: Vista con instrucciones de ruta

A.4 Ventana de inicio de sesión

Esta es la interfaz donde el usuario puede iniciar sesión utilizando el correo electrónico y la contraseña. Cuando se escriban estos datos en las cajas de texto apropiadas, se deberá pulsar en el botón iniciar sesión. Tras pulsar en este botón pueden ocurrir tres situaciones: se inicia sesión correctamente ya que las credenciales son correctas y disponemos de conexión a Internet, se produce un error inesperado por no disponer de conexión a Internet o el servidor no está en funcionamiento, se produce un error por haber insertado incorrectamente las credenciales. A continuación se muestran cada una de estas circunstancias.

Fig. A.9: Ventana de inicio de sesión

A.4.1 Inicio de sesión realizado con éxito

En el caso de que el correo electrónico y la contraseña del insertadas sean correctas, es decir, hay un usuario registrado con estas credenciales, entonces aparecerá el siguiente mensaje que implicará un inicio de sesión realizado correctamente. Podemos verlo en la figura A.9

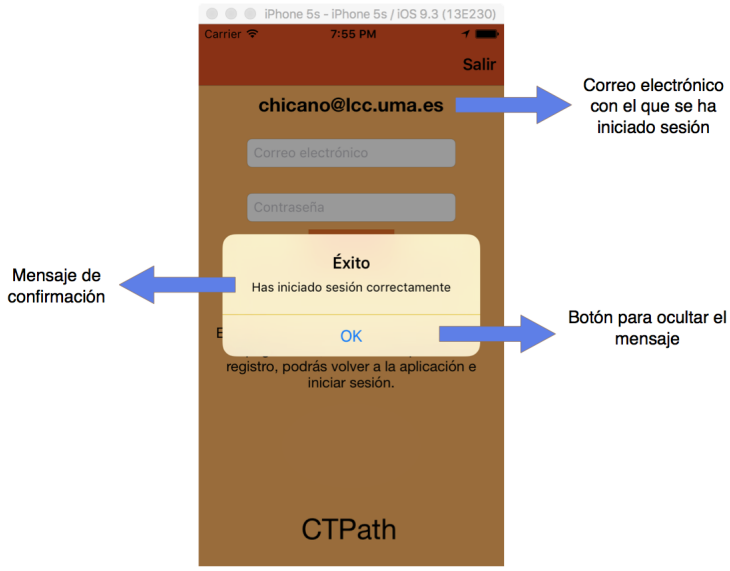

Fig. A.10: Inicio de sesión realizado con éxito

A.4.2 Inicio de sesión con error inesperado

En este caso, se ha producido un error en el servidor, puede que no este en funcionamiento, o que no se haya podido realizar la conexión al servidor debido a la conexión de Internet del usuario.

Fig. A.11: Inicio de sesión con error inesperado

A.4.3 Inicio de sesión con error en credenciales

Es posible que no se produzcan ninguno de estos dos sucesos y simplemente el usuario haya introducido mal sus credenciales.

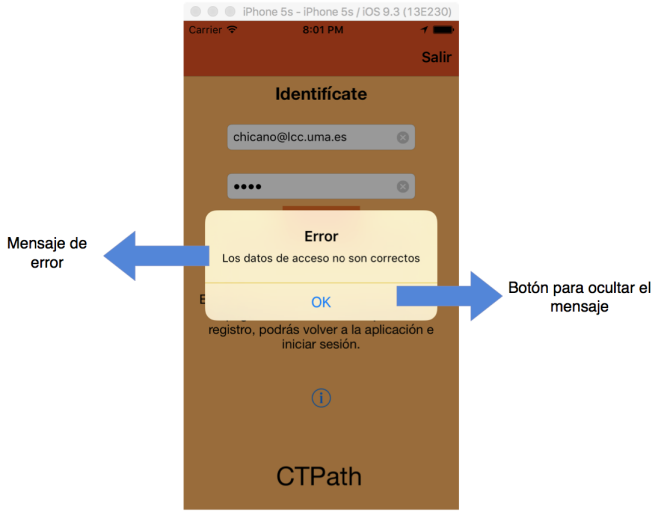

Fig. A.12: Inicio de sesión con error en credenciales

A.5 Ventana de búsqueda de calles

A continuación se explica cómo se utiliza el controlador de sugerencias de calles. Esta es su apariencia inicial:

Fig. A.13: Ventana de búsqueda de calles

A.5.1 Búsqueda de calles para punto inicial

Si deseamos insertar el punto de inicio de nuestra ruta, debemos escribir en la caja de texto “Inicio” el nombre de la calle que estemos buscando. En seguida, irán apareciendo en la tabla de sugerencias las distintas calles que más se parecen al texto insertado.

A.5.2 Búsqueda de calles para punto final

Del mismo modo, si queremos insertar el destino de nuestra ruta, debemos escribir esta calle en la caja de texto ”Final”, y obtendremos las sugerencias en la tabla.

Fig. A.14: Búsqueda de calles para punto inicial

Fig. A.15: Búsqueda de calles para punto final

Apéndice B

Documento de Especificación de Requisitos

B.1 Introducción

Los **requisitos** descritos en este documento se han obtenido a partir de varias reuniones con el director del proyecto donde se comentaba el propósito de este trabajo de fin de grado, su ámbito, la perspectiva del mismo, los requisitos funcionales y no funcionales que debía implementar, qué usuarios finales tendría la aplicación desarrollada y algunos comentarios relacionados con el proyecto.

B.1.1 Propósito

La aplicación a desarrollar pretende complementar el proyecto *CTPATH* ya que carece de aplicación *iOS*. Por tanto, este documento va dirigido al desarrollador de la aplicación, el autor de estas líneas, para conocer las especificaciones requeridas y a su vez, va dirigido al grupo de desarrollo del proyecto *CTPATH* para que el software desarrollado pueda ser comprendido correctamente.

B.1.2 Ámbito del sistema

El futuro sistema, en adelante *CTPATH-iOS* permitirá poder utilizar las rutas proporcionadas por el servidor CTPATH en dispositivos móviles con sistema operativo iOS. Las funciones que se desarrollen en esta aplicación están limitadas a la funcionalidad del propio servidor. *CTPATH-iOS* permitirá al usuario seleccionar el punto de inicio y el punto de destino, y mostrará las rutas más ecológicas que unen estos dos puntos. Estas rutas han sido previamente calculadas por el servidor CTPATH y podrán ser personalizadas si el usuario inicia sesión en la aplicación, pero no es una acción obligatoria para poder utilizar la aplicación. El registro del usuario en la aplicación no está contemplado, para poder darse de alta, deberá acudir a la página oficial del proyecto.

B.1.3 Definiciones y acrónimos

En esta sección se definen los términos y abreviaturas que serán utilizados en este documento.

Definiciones

Callback asíncrono. Un callback asíncrono es un fragmento de código *S*, que se pasa como parámetro de una función *F*. Ésta última ejecutará automáticamente el código *S* cuando finalice de ejecutar su propio código.

Servicio REST. Servicio web que define recursos identificados con una URI y se usan los métodos propios del protocolo HTTP (normalmente, GET, POST, PUT y DELETE) para especificar la operación a realizar con dichos recursos (leer, insertar, modificar y eliminar).

API REST. Conjunto de funciones que siguen la filosofía REST las cuales se ofrecen al desarrollador de una aplicación para que las pueda utilizar.

Abreviaturas y acrónimos

A continuación se muestra una tabla con las abreviaturas y acrónimos que vamos a utilizar.

API	*Application Programming Interface*
HTTP	*Hypertext Transfer Protocol*
REST	*Representational State Transfer*
TFG	Trabajo de Fin de Grado
JSON	*JavaScript Object Notation*

B.1.4 Referencias

- Especificación de Requisitos según el estándar de IEEE 830,
  `https://www.fdi.ucm.es/profesor/gmendez/docs/is0809/ieee830.pdf`

- Página web del proyecto CTPATH,
  `http://maxct.lcc.uma.es/index.php`

B.1.5 Visión general del documento

Este documento contiene dos capítulos, además del capítulo de introducción. El primer capítulo trata de comentar de manera general el proyecto, mientras que el capítulo segundo se adentra en la especificación de los requisitos funcionales y no funcionales de la aplicación.

B.2 Descripción general

B.2.1 Perspectiva del producto

CTPATH-iOS es un complemento de un proyecto mayor denominado *CTPATH*, el cual posee una aplicación web, una aplicación *Android* y un servidor web, el cual sirve las peticiones solicitadas por estas aplicaciones. Las funcionalidades implementadas en *CTPATH-iOS* deben ser las mismas al resto de aplicaciones al tratarse del mismo proyecto, y a su vez, estas funcionalidades deben ser implementadas a través de peticiones *HTTP* a un servicio *REST*, de tal manera, que la aplicación muestra los datos que el servidor ha calculado, esto implica que la aplicación no realiza cálculos, sino que procesa la respuesta que obtiene del servidor.

B.2.2 Funciones del producto

Las funciones de este producto son: ofrecer al usuario una forma de seleccionar el comienzo y el destino de la ruta, mostrándole las distintas posibilidades que puede elegir, visualizar la emisión de cada una de estas rutas, y una vez elegida, dirigir al usuario por las distintas calles hasta llegar al destino indicado, por otro lado, se debe ofrecer la posibilidad de iniciar sesión en la aplicación para que las rutas devueltas por el servidor CTPATH sean devueltas conforme al perfil de conducción del usuario con la sesión iniciada.

B.2.3 Características de los usuarios

Al ser una aplicación que muestra al usuario las rutas más ecológicas de acuerdo, entre otros factores, al vehículo que utiliza, los usuarios finales de la aplicación serán las personas que dispongan de un vehículo cuyo combustible sea un derivado del petróleo (gasolina,gasóleo...). Además, en las primeras versiones de la aplicación, sólo es posible calcular rutas en la ciudad de Málaga, por tanto, estos usuarios serán ciudadanos de Málaga, turistas, y/o visitantes de la ciudad.

B.2.4 Restricciones

Ya que la aplicación debe conectarse a los servicios *REST* que ofrece el proyecto *CTPATH*, esta comunicación se realizará a través de Internet, y por tanto, puede producirse un bloqueo del flujo normal de la aplicación debido a la espera de la respuesta del servidor, provocando que la interfaz de usuario también sufra un bloqueo. Este bloqueo está prohibido en aplicaciones *iOS*, por tanto para evitar este problema, todas las peticiones al servidor se deberán realizar en un hilo diferente al principal, utilizando los denominados *callbacks* asíncronos.

B.2.5 Suposiciones y Dependencias

Existen dos lenguajes de programación para desarrollar aplicaciones en *iOS*, uno es *Objective-C* y otro es *Swift*. La aplicación se ha programado en *Objective-C* ya que la versión existente de *Swift* en estos momentos no es definitiva y en cambio, *Objective-C* es un lenguaje con mucho más tiempo en el mercado, y por tanto, es más estable. Si se hubiese realizado en la versión actual de *Swift*, una versión futura con grandes cambios podría ocasionar la refactorización del código implementado. No obstante, parece ser que Apple busca hacer de *Swift* el lenguaje principal, por tanto en un futuro a largo plazo se podría dar el caso de tener que cambiar a *Swift*.

B.3 Requisitos específicos

Comentaremos los distintos requisitos funcionales, no funcionales, y hablaremos sobre requisitos futuros para definir algunas mejoras que se podrían realizar en el sistema actual.

B.3.1 Requisitos funcionales

Vamos a comentar los requisitos funcionales que debe cumplir. Atendiendo a las reuniones con el director del proyecto, se pueden dividir en tres grupos: cálculo de rutas ecológicas, información al usuario, seguimiento del perfil del conductor.

Cálculo de rutas ecológicas

- **RF01 - Introducir datos de la ruta**. La aplicación debe permitir al usuario introducir el punto inicial y el punto final de la ruta.
  - **RF01.1 - Seleccionar en el mapa**. El usuario podrá seleccionar un punto presionando la posición en el mapa hasta que aparezca una marca. Si no hay punto de inicio, se colocará una marca de inicio. Si hay punto de inicio, se colocará una marca de destino.
  - **RF01.2 - Buscador de calles**. Existirá un buscador con dos entradas de texto, una para el punto inicial y otra para el punto final. Dependiendo de donde se introduzca el texto, se colocará marca inicial o final.
- **RF02 - Modificar datos de la ruta**. La aplicación debe permitir al usuario modificar el punto inicial o el punto final en un momento dado.
  - **RF02.1 - Selección de marca**. El usuario podrá seleccionar una marca y desplazarla a una nueva posición.
  - **RF02.2 - Buscador de calles**. El usuario insertará una calle en la entrada de texto oportuna y la marca respectiva se trasladará a la nueva posición.

  - **RF02.3 - Seleccionar en el mapa**. El usuario presionará una nueva posición en el mapa hasta que aparezca en la nueva posición marcada.

- **RF03 - Calcular ruta**. Cuando el punto inicial y el punto final se especifiquen la aplicación deberá conectarse al servicio web automáticamente y obtener las rutas calculadas en formato *JSON*.

- **RF04 - Manejo de errores**. Cuando se produzca un error en el cálculo de rutas, se mostrará un mensaje al usuario informándole del problema ocurrido.

  - **RF04.1 - Error en el servidor**. Informar al usuario en caso de que el servicio no esté disponible.
  - **RF04.2 - Error en el cálculo de la ruta**. Informar al usuario si el servicio no puede calcular una ruta entre los puntos insertados por el usuario.
  - **RF04.3 - Error de conexión**. Informar al usuario en caso de que no tenga conexión a Internet.

Información al usuario

- **RF05 - Mostrar rutas calculadas**. Una vez responda el servidor con toda la información relativa a las rutas, se debe mostrar al usuario de distintas formas.

  - **RF05.1 - Trazado de las rutas**. Se pintará en el mapa el tradazo de las rutas devueltas cada una con un color, siendo la más ecológica pintada de color verde y la menos ecológica pintada de color negro. El resto será un degradado de estos dos tonos.
  - **RF05.2 - Datos de las rutas**. Habrá una tabla con el nombre que identifica a cada una de la rutas, la duración de la ruta y el color que identifica a esta ruta, siguiendo la misma metodología de colores explicada en el punto anterior. Cada fila de la tabla se podrá seleccionar.

- **RF06 - Mostrar detalles de la ruta**. Al seleccionar una ruta aparecerá una vista con toda la información relativa a la ruta seleccionada, duración, fecha y emisión de gases nocivos. Existirá un botón en esta vista que abra el itinerario de la ruta

  - **RF06.1 - Itinerario de la ruta**. La aplicación debe ofrecer al usuario la posibilidad de ver en formato textual los pasos que debe seguir para completar la ruta correctamente.
  - **RF06.2 - Ocultar información**. Debe existir un botón para volver a la interfaz donde se muestra únicamente el mapa, para que el usuario pueda insertar una ruta nueva.

Seguimiento del perfil del conductor

- **RF07 - Autenticación**. El usuario podrá iniciar sesión en la aplicación con su correo electrónico y contraseña. Siempre puede iniciar sesión con otro usuario para cambiar el perfil de conducción después de haber iniciado sesión anteriormente .
  - **RF07.1 - Autenticación con éxito**. El servidor web devolverá un token de autenticación que deberá ser almacanedao y utilizado en las siguientes peticiones al servidor.
  - **RF07.2 - Error en el servidor**. Se informará al usuario en caso de que el servicio de autenticación no estuviera disponible.
  - **RF07.3 - Error de conexión**. Se informará al usuario en caso de que no tenga conexión a Internet.
- **RF08 - Peticiones al servidor**. En caso de no haber iniciado sesión, se podrá utilizar la aplicación sin impedimentos. Si se inicia sesión previamente, se añadirá a una cabecera AuthenticationToken a la petición *HTTP* con el token devuelto por el servidor.

B.3.2 Requisitos no funcionales

- **RNF01 - Lenguaje**. El lenguaje utilizado debe ser propio de *Apple*: *Objective-C* o *Swift*.
- **RNF02 - Usabilidad**. La aplicación debe poder visualizarse correctamente en todos los dispositivos móviles y tabletas de *Apple*.

B.3.3 Requisitos futuros

Por razones de tiempo existen algunos requisitos que no se han especificado para esta versión de la aplicación pero que pueden implmentarse para mejorar el funcionamiento del sistema actual. Estos requisitos son los siguientes:

- **RFU01 - Direcciones por voz**. *En relación al requisito funcional RF06.1*. Actualmente se muestra el itinerario de la ruta en formato textual, una mejora considerable del sistema es poder guiar al usuario en su ruta mediante comandos de voz, y así permitirle usar la aplicación desde el vehículo en movimiento.
- **RFU02 - Rutas pasando por un punto intermedio**. Aunque esta función no esté implementada por el servidor web, podemos afirmar que si primero calculamos la ruta más ecológica desde el inicio al punto intermedio y después calculamos la ruta más ecológica desde este mismo, hasta el destino, tendríamos la ruta más ecológica para ir a nuestro destino pasando por un punto intermedio. Por tanto, esto supondría un aumento de funcionalidad del sistema.

B.4 Casos de uso

Una vez definidos los requisitos funcionales y no funcionales, vamos a elaborar los diferentes casos de uso de nuestra aplicación. Con ellos se pretende dar una visión más detallada de las funciones a implementar. A continuación se muestra un diagrama de caso de uso que comprende los requisitos funcionales **RF01, RF02** y **RF07**. Posteriormente expondremos otro diagrama de caso de uso comentando otras funcionalidades del sistema.

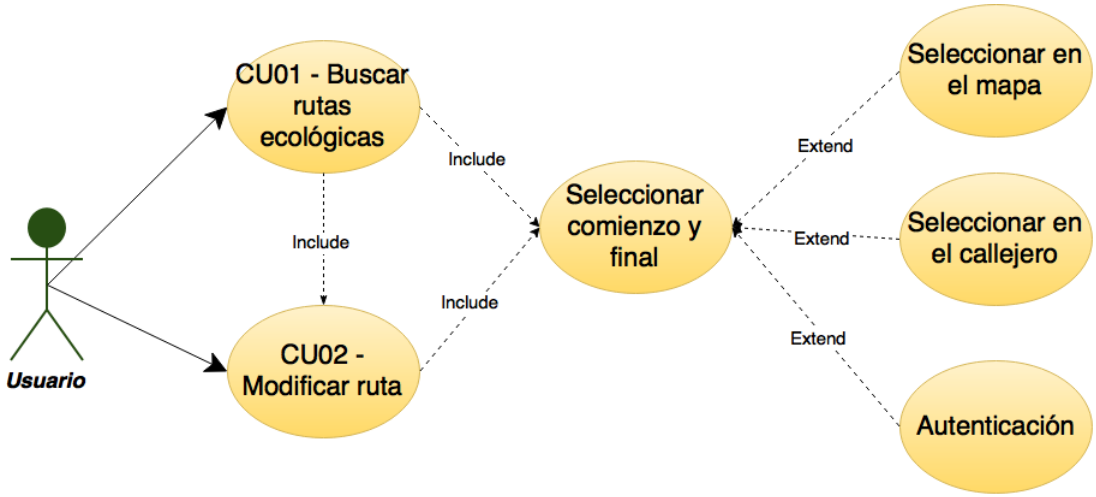

Fig. B.1: Diagrama de casos de uso conteniendo CU01 y CU02

En la figura 2.1 se muestra un diagrama de casos de uso donde están representados el caso de uso “Buscar rutas ecológicas” (CU01),“Modificar ruta” (CU02). En cualquiera de las dos situaciones, buscar o modificar ruta, el usuario deberá seleccionar el comienzo y el final de la ruta, a través de un mapa o desde un callejero, si lo desea podrá autenticarse. Vamos a explicar con más detalle cada uno de estos casos de uso.

Caso de uso CU01 - Buscar rutas ecológicas

Descripción: El usuario podrá buscar rutas ecológicas para llegar al destino deseado.

Precondición: Se deben seleccionar los puntos de inicio y fin de la ruta.

Post-condición: Se mostrará en el mapa las distintas rutas calculadas.

Escenario principal:

1. El usuario selecciona desde el mapa o el callejero el punto de inicio.
2. El usuario selecciona desde el mapa o el callejero el punto final.
3. El sistema realiza una petición al servidor con los datos insertados.
4. Los datos devueltos se muestran en el mapa.

Escenarios alternativos:

4. No conecta con el servidor por un problema:
    a) El sistema muestra un mensaje con el error.
    b) Vuelta al paso 1.

---

2. El callejero no encuentra la calle:
    a) El usuario cancela la búsqueda por callejero.
    b) Vuelta al paso 1.

Casos de prueba:

- **CP01 - Buscar rutas ecológicas**. El usuario selecciona un punto de inicio desde el mapa o el callejero, realiza la misma acción para el punto final de la ruta. El sistema se conecta con el servidor enviando los parámetros de la ruta, se recibe una respuesta y se pinta en el mapa.
- **CP02 - Error en la conexión**. Desconectar el acceso Internet, seleccionar el punto inicial en el mapa o en el callejero, realizar la misma acción para el punto final de la ruta. El sistema muestra un error avisando del error ocurrido.

Caso de uso CU02 - Modificar ruta

Descripción: El usuario podrá modificar el punto de inicio y el punto final seleccionados previamente.

Precondición: Los puntos inicial y final deben haber sido seleccionados previamente.

Post-condición: Se actualizarán los puntos inicial y final a los nuevos valores introducidos por el usuario.

Escenario principal:

1. El usuario modifica el punto inicial, arrastrando la marca existente en el mapa, o seleccionando un punto desde el callejero.
2. El usuario modifica el punto final, arrastrando la marca existente en el mapa, seleccionando un punto desde el callejero, o manteniendo pulsado el dedo en nuevo lugar en el mapa.
3. El sistema realiza una petición al servidor con los datos insertados.
4. Los datos devueltos se muestran en el mapa.

Escenarios alternativos:

2. El usuario cancela el cambio de posición del punto inicial:
    a) La marca se deja en la misma posición.

Casos de prueba:

- **CP03 - Modificar ruta**. El usuario modifica la posición del punto de inicio arrastrando la marca en el mapa o desde el callejero, realiza la misma acción para el punto final de la ruta. El sistema se conecta con el servidor enviando los parámetros de la ruta, se recibe una respuesta y se pinta en el mapa.
- **CP04 - Cancelar modificación**. El usuario elecciona una marca, cuando esté disponible para arrastrarla, el usuario la suelta. El sistema no realiza ninguna acción puesto que la posición es la misma.

Para completar todos los casos de uso del sistema mostramos el siguiente diagrama donde aparece el caso de uso *CU01* y las relaciones de éste con otros.

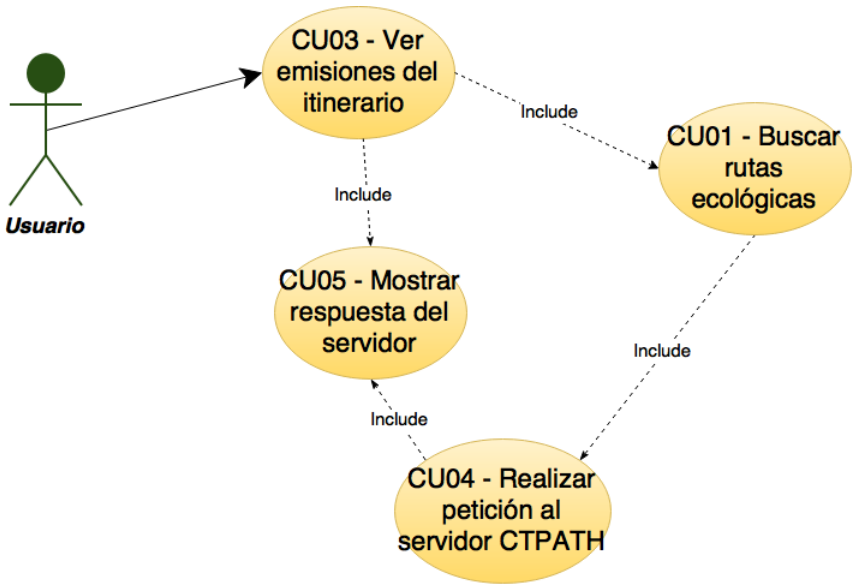

Fig. B.2: Diagrama de casos de uso incluyendo CU01, CU03, CU04, CU05

En este diagrama se pretende explicar que para que el usuario pueda realizar el caso de uso CU03 - *Ver emisiones del itinerario* debe primero buscar el itinerario implicado. Ya hemos visto en los casos de uso anteriores cómo se ejecuta éste. La novedad del diagrama anterior es la introducción de nuevos caso de usos relacionados con el caso de uso CU03. Tras buscar la ruta el usuario, la aplicación debe realizar la petición al servidor CTPATH y mostrar la respuesta recibida para que el usuario pueda ver las emisiones calculadas. Vamos a explicar con detalle cada uno de los casos de uso implicados en el diagrama anterior.

Caso de uso CU03 - Ver emisiones del itinerario

Descripción: El usuario podrá comprobar las emisiones de gases nocivos que emite la ruta elegida.

Precondición: Se ha realizado una búsqueda de itinerarios.

Post-condición: Aparece una vista con los detalles de una ruta incluyendo sus emisiones.

Escenario principal:

1. El usuario selecciona una ruta de la tabla mostrada.

2. Se abre una pestaña con los datos de la ruta pudiendose ver las emisiones de la ruta.

Casos de prueba:

- **CP05 - Ver emisiones del itinerario**. El usuario realiza una búsqueda de rutas, se obtiene una lista con las rutas calculadas. El usuario selecciona una de ellas. Tras esto, aparece una ventana con los detalles de la ruta y sus emisiones.

Caso de uso CU04 - Realizar petición al servidor CTPATH

Descripción: La aplicación deberá conectarse al servicio que realiza los cálculos de ruta.

Precondición: El usuario solicita una búsqueda de rutas.

Post-condición: Se obtienen las rutas calculadas por el servidor

---

Escenario principal:

1. Se obtienen los parámetros de la búsqueda del usuario:
   - Coordenadas de los puntos inicial y final (parámetros fromPlace y toPlace)
   - Fecha y hora actual (parámetros time y date)
   - Tipo de vehículo (En esta versión sólo es para coche. Parámetro mode)
2. Se crea la URL para realizar la petición, por ejemplo: `https://mallba3.lcc.uma.es/otp/routers/default/plan?fromPlace=36.712,-4.485&toPlace=36.707,-4.434&time=9:18pm&date=06-27-2016&mode=CAR`
3. Se ejecuta la petición.
4. Se obtiene la respuesta del servidor.

Escenarios alternativos:

4. Se obtiene un error en la petición del servidor:
   a) Se muestra un mensaje de error.
   b) Vuelta al paso 1

Casos de prueba:

- **CP06 - Realizar petición**. La aplicación obtiene los parámetros de la ruta, construye la URL de la petición y la realiza. El servidor devuelve los datos de las rutas calculadas.
- **CP07 - Error en la petición**. La aplicación obtiene los parámetros de la ruta, construye la URL. Sin conexión a Internet se realiza la petición. El sistema muestra un mensaje de error.

Caso de uso CU05 - Mostrar respuesta del servidor

Descripción: La aplicación deberá mostrar la información devuelta por el servidor.

Precondición: El servidor ha respondido con los datos de las rutas.

Post-condición: Se mostrará la información de cada una de las rutas.

Escenario principal:

1. Se obtienen los datos de las rutas calculadas por el servidor:
2. Se recoge la información de cada ruta en una estructura de datos:
3. Cuando se seleccione una ruta, se accederá a sus datos y se mostrarán en una ventana.

Casos de prueba:

- **CP08 - Mostrar datos**. La aplicación obtiene la información hallada por el servidor, la procesa y la muestra cuando el usuario selecciona una ruta.