Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

Proyecto Fin de Carrera

Desarrollo de aplicación del sistema web de gestión de carreteras de Iternova para plataformas móviles

Autor

Francisco Javier Aguerri Moreno

Ponente

Antonio Valdovinos Bardají

Director

Jorge Casas Cañada

Escuela de Ingeniería y Arquitectura
Septiembre 2012

Resumen

Desarrollo de aplicación del sistema web de gestión de carreteras de Iternova para plataformas móviles

El presente proyecto fin de carrera pretende dar una respuesta a la problemática planteada por las tareas de mantenimiento generadas por un red de carreteras. Para ello, se ha realizado un módulo de agenda móvil integrado dentro del Sistema de Gestión Web de Carreteras de Iternova S.L., un sistema web SCADA (Supervisory Control and Data Acquisition) para la gestión y explotación de carreteras.

La agenda de vialidad es el sistema encargado de gestionar la información relacionada con las operaciones de vialidad llevadas a cabo en la carretera. Éstas incluyen la actuación ante accidentes (tanto con afección a la calzada como sin afección), incidencias (animales muertos, objetos en la calzada, desprendimientos) o deterioros de vialidad (pavimentos, obras, señalización).

El proyecto concretamente se centra en el desarrollo de una solución para plataformas móviles Android que permita afrontar con éxito toda la problemática que presenta la gestión telemática de una carretera (gestión de todos los datos e imágenes generados). Este sistema será la herramienta de los operarios encargados del mantenimiento de la carretera para la realización de las tareas pendientes, y de los supervisores para controlar la correcta ejecución del proceso.

Las principales funcionalidades de este sistema son:

- Una agenda de tareas que agrupa y organiza todas las tareas pendientes de realizar. El operario sólo recibe las tareas asignadas al sector donde se encuentra, gracias a la inteligencia del motor de búsqueda que proporciona el sistema. La disposición del listado no es arbitrario, sino que se realiza una búsqueda ordenada, donde las tareas prioritarias aparecen primero.
- Gestión de formularios. Los formularios no son estáticos sino que se realiza una carga dinámica desde el servidor. El sistema debe garantizar que los cambios que haya en las fichas sean transparentes al usuario, por lo que la gestión de las bases de datos debe ser realizada cuidadosamente. El sistema debe asegurar el almacenamiento de la información en situaciones de ausencia de cobertura, y su envío al servidor en el momento oportuno.
- Diversas opciones de configuración, que permiten adaptar el funcionamiento de la aplicación a las necesidades del usuario y a la situación en la que nos encontremos. Es posible refrescar el listado de las tareas de manera automática de manera transparente para el usuario, así como el envío de la información pendiente cada cierto tiempo. También debe permitir realizar una adecuada limpieza de las bases de datos, y otras opciones de configuración menores.

En el desarrollo se ha tenido en cuenta el entorno de utilización: este es el de una carretera convencional, donde a menudo nos encontraremos en situaciones de ausencia de cobertura o de baja velocidad de conexión. Para solventar estos problemas, la aplicación cuenta con diferentes modos de funcionamiento para afrontar con garantías todos los posibles escenarios.

Es muy posible que, durante la jornada del operario en la carretera, no se disponga de cobertura, lo que genera un problema de gestión importante. Por un lado, el sistema debe seguir funcionando de manera autónoma sin conexión con el servidor, y por otro

la información registrada debe quedar almacenada y ser enviada cuando se detecte conexión a Internet. También debe tenerse en cuenta que en ausencia de cobertura no es posible obtener geolocalización ni recursos externos, lo que genera una problemática adicional.

Otro ejemplo de la gestión de la información en función de las condiciones se da en la transmisión de imágenes. Es posible configurar el envío de imágenes sólo en entornos de alta velocidad de conexión (redes Wi-Fi), ya que el consumo de datos en redes 3G es alto y ralentiza el manejo de la aplicación.

El sistema también cuenta con un sistema de alerta configurable que avisa al usuario (sonido y vibración del dispositivo) cuando existen nuevas tareas. Estas alertas sólo aparecerán en el caso de que las nuevas tareas se sitúen en el sector en el que trabaja el usuario.

Por último, existe también un sistema de roles que otorga diferentes privilegios (capacidad para dar de alta nuevas tareas, validación de tareas ya realizadas, etc) en función del tipo de usuario registrado.

El Sistema de Gestión Web de Carreteras de Iternova S.L. que engloba el sistema de la agenda de vialidad, es pionero en la gestión de carreteras y está implantado en algunas de las carreteras más importantes de México (Atlacomulco-Maravatío, Sinaloa) y es también la solución elegida para la gestión de las carreteras de algunas regiones de España como Aragón, Valencia y Castilla La Mancha, entre otras.

Índice de contenido

Índice de ilustraciones

Índice de tablas

Capítulo 1. INTRODUCCIÓN

En este primer capítulo se explicará la naturaleza, contexto y propósito de este proyecto fin de carrera.

1.1. Objetivos del proyecto y la solución adoptada

El objetivo de este proyecto fin de carrera es crear un sistema que ofrezca una solución telemática al problema de la gestión de incidencias en la carretera.

Dicha solución consiste en el módulo agenda de vialidad, que será el sistema encargado de gestionar todas las tareas llevadas a cabo en la carretera como respuesta a las incidencias.

El núcleo de nuestro sistema es una aplicación móvil, que se apoya en una arquitectura cliente-servidor. Es decir, tendremos múltiples usuarios (terminales) trabajando a la vez con la aplicación desarrollada que se comunican con el servidor, que actúa de elemento central.

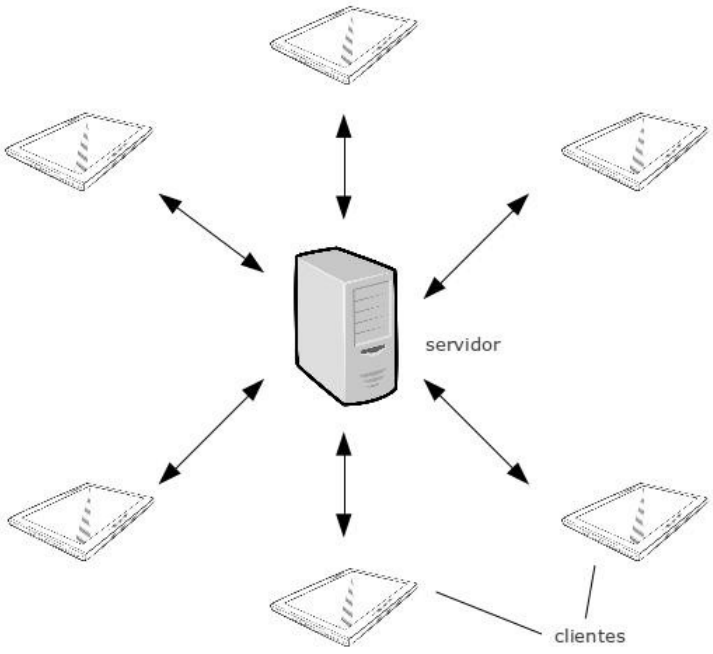

Figura 1. Diagrama de la arquitectura del sistema

Esta aplicación móvil consiste en la versión para dispositivos portables (tablet, smartphones) del módulo de agenda del Sistema de Gestión Web de Carreteras (SGWC) de Iternova. Dicho módulo de agenda es el encargado de gestionar la información relacionada con las operaciones de vialidad llevadas a cabo en la carretera. El objetivo es mejorar la organización y el servicio ofrecido a los ciudadanos.

La aplicación debe solventar correctamente dos problemas principales:

- A menudo nos encontramos en situaciones de baja capacidad de la red o directamente ausencia de la misma.

- A pesar de lo anterior, el sistema debe ser fiable y ágil dado que la naturaleza del problema a resolver (incidencias en las carreteras) no admite un gran margen de error ni largas esperas.

Debido a ello, se implementó una solución con las siguientes características:

- Compatible para diferentes modelos de terminales.

- Optimizada para minimizar los tiempos de carga, y agilizar el el uso de la aplicación por parte del usuario en entornos remotos.

- Se tienen en cuenta los problemas de límites de ancho de banda y cobertura de las redes móviles, disponiendo de los mecanismos oportunos para agilizar la velocidad de la aplicación y evitar la pérdida de información por falta de conexión con el servidor e Internet.

Respecto a la cobertura, la aplicación dispone de todos los elementos necesarios para permitir trabajar aún cuando ésta no exista (es decir, de forma autónoma), volviendo a sincronizar la información con los servidores cuando la conexión vuelva a estar disponible, de forma que se garantice la continuidad del funcionamiento.

La aplicación permite a los operarios la actualización instantánea de las tareas en el mismo momento en que proceden a su actuación, al mismo tiempo que se controla la posición exacta en la que se encuentra, posibilitando el seguimiento de la resolución de las tareas por parte de la empresa para mantener el control sobre las actuaciones realizadas.

1.2. Contexto profesional

El proyecto fin de carrera se realizó íntegramente en Iternova S.L. , una empresa de consultoría tecnológica especializada en soluciones tecnológicas innovadoras para la gestión de carreteras.

Iternova es una empresa con una continua apuesta por la innovación, que les ha permitido desarrollar productos de gran éxito e implantación a nivel nacional, como el sistema SGWC (Primer sistema web de gestión integral para la explotación de carreteras), por el cual la empresa recibió el VIII premio nacional ACEX 2012 a la seguridad en conservación de carreteras, además de otros galardones como el premio sociedad de la información Aragón 2012 a la empresa del año.

SGWC de Iternova

El presente proyecto fin de carrera, como ya hemos comentado brevemente, se enmarca en el Sistema de Gestión Web de Carreteras (SGWC) de Iternova. El SGWC se trata del primer Sistema Web SCADA (*Supervisory Control And Data Acquisition*) desarrollado para la gestión de la conservación y explotación de carreteras, e implantado a nivel nacional e internacional. El sistema plantea un importante salto cualitativo en la gestión de la explotación de carreteras, permitiendo acceder a toda la información de las mismas de una forma integrada:

- Se trata de un sistema web (permite el acceso a la información en cualquier momento y lugar), multidispositivo (facilita la integración de hardware de cualquier fabricante), basado en código libre (no requiere de licencias ni mantenimientos), modular (se adapta a las necesidades de cualquier demarcación) y multiusuario (con diferentes roles de acceso a la información).

- El sistema está basado en el concepto SaaS (*Software as a Service*), pudiendo ser adaptado a las necesidades de cada cliente.

El innovador sistema desarrollado, formado por varios módulos, integra de una forma optimizada la gestión de todos los elementos de interés de las carreteras:

- Módulo de información y gestión general de carreteras, con sus elementos más significativos inventariados y geolocalizados.

- Módulo de gestión de la vialidad en tiempo real: sistema de gestión de flotas (GPS), de cámaras de explotación, estaciones meteorológicas, aforos, paneles de información, etc.

- Módulos de comunicación, módulos de documentación (integra un completo gestor de expedientes), y funcionalidades avanzadas de gestión Web.

- Módulo de agenda avanzada (para la gestión de las actuaciones programables y no programables de vialidad), y sistema de control y seguimiento de trabajos. Aquí es donde se integra el trabajo desarrollado en este PFC.

El sistema SGWC se ha convertido en el sistema más avanzado de gestión de la Explotación y Conservación de carreteras, cuyo éxito está siendo avalado por su implantación paulatina en las principales provincias y comunidades autónomas españolas, tales como Aragón, Valencia, Murcia, Castilla La Mancha y Soria, así como en las principales carreteras de México (Atlacomulco-Maravatío, Sinaloa).

1.3. Motivación

La motivación para realizar este PFC en concreto vino por varias razones.

La primera, era mi voluntad de realizar el PFC en el ámbito de una empresa. Primeramente, creo que es una estupenda oportunidad de entrar en contacto con el mundo laboral y adquirir una primera experiencia muy valiosa. Y segundo, porque creo que los conocimientos adquiridos en una empresa son mucho más cercanos a lo que se demanda en el mercado laboral.

En segundo lugar, considero que las tecnologías estudiadas durante el desarrollo del proyecto son punteras hoy en día, y están muy presentes en el mercado laboral. Este PFC se basa en el desarrollo de una aplicación enfocada al uso en dispositivos móviles (tablets, smartphones). Estos dispositivos están ganando cuota de mercado día a día frente a medios más tradicionales como ordenadores de sobremesa en el ámbito de las TIC (Tecnologías de la Información y la Comunicación).

Es obvio que de la misma manera, asistimos a un auge de las aplicaciones móviles. Existe una amplísima gama de ellas y cada día aparecen gran cantidad de otras completamente novedosas, algunas de ellas con muchísimo éxito a nivel mundial. Por estas razones, la formación que este proyecto ofrecía en este sector es una gran oportunidad.

1.4. Elementos innovadores

Como ya se ha comentado anteriormente, el SGWC de Iternova es un sistema innovador en el ámbito de la gestión de la explotación de carreteras por tratarse de un sistema integral, flexible, y ubicuo.

El presente PFC debe amoldarse pues a las características del sistema en el que se encuentra integrado. El trabajo desarrollado es un sistema pionero en el sector de la gestión de carreteras a nivel nacional e internacional por diversos motivos:

- Al tratarse de información crucial para algo sensible como es las incidencias en las carreteras, el sistema está optimizado para funcionar de la manera más ágil posible, ofreciendo una interfaz sencilla de manejar, una gestión inteligente de la información introducida, y un sistema de alertas para que el usuario esté informado de los últimos sucesos.

- La aplicación está pensada para funcionar en entornos de baja o nula conectividad, tratando de aprovechar el escaso ancho de banda disponible.

- Sistema de permisos y roles de usuario, que permiten ejercer el control sobre los trabajos realizados y aporta otro elemento optimizador, ya que la información intercambiada depende del tipo de usuario y se ajusta a lo que éste necesita, evitando enviar datos que no serán utilizados.

1.5. Estructura de la memoria

La presente memoria se estructura en dos partes presentadas en el mismo tomo.

En esta parte principal de la memoria se tratará de presentar el trabajo desarrollado en el proyecto de forma concisa, detallando los aspectos necesarios para una comprensión adecuada de este proyecto. Esta parte principal a su vez se divide en varios capítulos:

1. Introducción. Se tratarán todos los aspectos necesarios para entender el contexto, motivación y objetivos del proyecto.

2. Trabajo previo. Se comentarán los conocimientos previos que fueron necesarios antes de comenzar a realizar el proyecto.

3. Desarrollo de la aplicación. En este capítulo se detalla el trabajo realizado y se explican con detenimiento las necesidades que demanda el mismo y las soluciones aplicadas.

4. Conclusiones. Se presenta una reflexión sobre los resultados del proyecto y las posibles líneas de continuación del mismo.

En cuanto a los anexos, contienen información no fundamental para una comprensión mínima del proyecto, pero muy útil si se pretende profundizar, sobre todo en los aspectos técnicos. La memoria del proyecto contiene los siguientes anexos:

A) Manual de usuario de la aplicación. Se trata del manual de referencia entregado a nuestros clientes para guiarles en el uso del sistema.

B) Elementos clave del código de la aplicación. Aquí se comentarán aspectos clave de la programación del sistema y el por qué de su importancia.

Capítulo 2. TRABAJO PREVIO

El trabajo previo para el desarrollo de este proyecto fin de carrera ha consistido en el aprendizaje de un conjunto de tecnologías, si bien mientras en algunas de ellas se ha alcanzado un conocimiento bastante profundo de la materia (Javascript y sus frameworks -jQuery, jQuery mobile y PhoneGap- y HTML principalmente) en otros se han adquirido unos conocimientos suficientes para realizar el trabajo pero sin obtener un manejo completo de la tecnología (SQL, PHP) y en algunos sólo se han visto los conocimientos específicos que se necesitaban aplicar (CSS). Para una mejor comprensión de este proyecto, se presenta una descripción de las tecnologías empleadas durante el desarrollo del mismo.

2.1. Android

Android es un sistema operativo inicialmente pensado para teléfonos móviles, al igual que iOS, Symbian y Blackberry OS. Lo que lo hace diferente es que está basado en Linux, un núcleo de sistema operativo libre, gratuito y multiplataforma.

El sistema permite programar aplicaciones en una variación de Java[1] llamada Dalvik. El sistema operativo proporciona todas las interfaces necesarias para desarrollar aplicaciones que accedan a las funciones del teléfono (como el GPS, las llamadas, la agenda, etc.) de una forma muy sencilla en un lenguaje de programación orientada a objetos muy conocido como es Java.

Esta sencillez, junto a la existencia de herramientas de programación gratuitas, hacen que una de las cosas más importantes de este sistema operativo sea la cantidad de aplicaciones disponibles, que extienden casi sin límites la experiencia del usuario.

Además Android incorpora de serie todas las herramientas necesarias para la creación y gestión de bases de datos SQLite, un motor de bases de datos muy popular en la actualidad por ofrecer características tan interesantes como su pequeño tamaño, no necesitar servidor, precisar poca configuración, ser transaccional y por supuesto ser de código libre.

Una de las mejores características del sistema operativo Android es que es completamente libre. Es decir, ni para programar en este sistema ni para incluirlo en un teléfono hay que pagar nada. Y esto lo hace muy popular entre fabricantes y desarrolladores, ya que los costes para lanzar un teléfono o una aplicación son muy bajos.

En febrero de 2011 se anunció la versión 3.0 de Android, llamada con nombre en clave Honeycomb, que está optimizado para tabletas en lugar de teléfonos móviles.

2.2. jQuery y jQuery mobile

jQuery es un framework[2] Javascript[3]. Cuando un desarrollador tiene que utilizar Javascript, generalmente tiene que preocuparse por hacer scripts compatibles con varios navegadores y para ello tiene que incorporar mucho código que lo único que hace es detectar el navegador del usuario, para hacer una u otra cosa dependiendo de si es Internet Explorer, Firefox, Opera, etc. Aquí jQuery es donde más nos puede ayudar, puesto que implementa una serie de clases (de programación orientada a objetos) que nos permiten programar sin preocuparnos del navegador con el que nos está visitando el usuario, ya que funcionan de exacta forma en todas las plataformas

1 Java es un lenguaje de programación orientado a objetos, independiente de la plataforma, con el que podemos realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. Está desarrollado por la compañía Sun Microsystems con gran dedicación y siempre enfocado a cubrir las necesidades tecnológicas más punteras.

2 Un framework es un producto que sirve como base para la programación avanzada de aplicaciones, que aporta una serie de funciones o códigos para realizar tareas habituales. Por decirlo de otra manera, son unas librerías de código que contienen procesos o rutinas ya listos para usar. Los programadores utilizan los frameworks para no tener que desarrollar ellos mismos las tareas más básicas, puesto que en el propio framework ya hay implementaciones que están probadas, funcionan y no se necesitan volver a programar.

3 Javascript es un lenguaje de programación orientado a objetos ampliamente utilizado en Internet. Se utiliza principalmente en su forma del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas, y en bases de datos locales al navegador, entre otros. Su uso en aplicaciones externas a la web, por ejemplo en documentos PDF, o aplicaciones de escritorio es también significativo.

más habituales.

Así pues, este framework Javascript, nos ofrece una infraestructura con la que tendremos mucha mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Por ejemplo, con jQuery obtendremos ayuda en la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de AJAX, etc.

Además, jQuery tiene licencia gratuita para uso en cualquier tipo de plataforma, personal o comercial. Para usarlo simplemente tendremos que incluir en nuestras páginas un script Javascript que contiene el código de jQuery, que podemos descargar de la propia página web del producto.

Finalmente, jQuery Mobile es un framework para desarrollo de aplicaciones y sitios web optimizados para smartphones y tablets. Agrega una capa más al jQuery tradicional y busca suplir algunas necesidades que los programadores de dispositivos móviles padecen. Antes de que aparecieran estas herramientas, los desarrolladores tenían casi que programar para cada dispositivo en concreto pero con la aparición de jQuery Mobile, nos abstraernos de la lógica específica de cada dispositivo.

jQuery mobile se apoya también en el lenguaje de marcado HTML5, la última versión del popular lenguaje de marcado HTML[4]. Esta nueva especificación añade capacidades totalmente novedosas para el lenguaje HTML pretendiendo proporcionar una plataforma con la que desarrollar aplicaciones web más parecidas a las aplicaciones de escritorio.

2.3. AJAX

El término AJAX se presentó por primera vez en el artículo ”Ajax: A New Approach to Web Applications” publicado por Jesse James Garrett en 2005. En realidad, el término AJAX viene de Asynchronous JavaScript + XML5, que se puede traducir como ”JavaScript asíncrono + XML”. El artículo define AJAX de la siguiente forma:

“AJAX no es una tecnología en sí misma. En realidad, se trata de varias tecnologías independientes que se unen de formas nuevas y sorprendentes.”

Las tecnologías que forman AJAX son:

- XHTML[5] y CSS[6], para crear una presentación basada en estándares.
- DOM[7], para la interacción y manipulación dinámica de la presentación.
- XML, XSLT[8] y JSON[9], para el intercambio y la manipulación de información.

4 HTML, (HyperText Markup Language, lenguaje de marcado de hipertexto) es un lenguaje de marcado (no es un lenguaje de programación) predominante hoy en día con el que se definen las páginas web. Básicamente se trata de un conjunto de etiquetas que sirven para dar formato al texto y otros elementos que compondrán una página web.

5 XHTML es un lenguaje similar a HTML, pero con algunas diferencias que lo hacen más robusto y aconsejable para el modelado de páginas web. Las siglas corresponden a eXtensible Hypertext Markup Language.

6 CSS es el acrónimo de Cascading Style Sheets (hojas de estilo en cascada). CSS es un lenguaje de estilo que define la presentación de los documentos HTML. Por ejemplo, CSS abarca cuestiones relativas a fuentes, colores, márgenes, líneas, y muchos otros.

7 El Modelo de Objetos del Documento (DOM, por sus siglas en inglés) es una interfaz de programación de aplicaciones (API) para documentos HTML y XML. Define la estructura lógica de los documentos y el modo en que se accede y manipula un documento.

8 XSLT (eXtensible Stylesheet Language Transformations) es un estándar de la organización W3C que presenta una forma de transformar documentos XML en otros e incluso a formatos que no son XML. Actualmente, XSLT es muy usado en la edición web, generando páginas HTML o XHTML.

9 JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos que desde hace algún tiempo ha comenzado a usarse de una forma tan popular como se usó en su tiempo el XML.

- XMLHttpRequest[10], para el intercambio asíncrono de información.
- JavaScript, para unir todas las demás tecnologías.

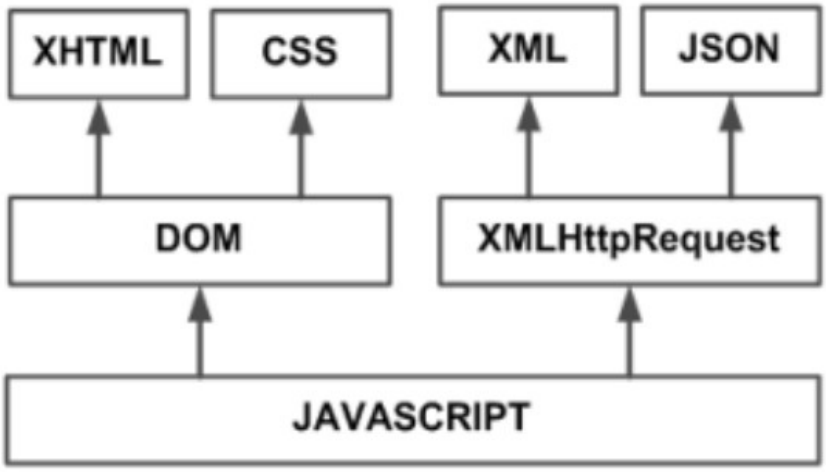

Figura 2. Tecnologías agrupadas bajo el concepto de AJAX

Desarrollar aplicaciones AJAX requiere un conocimiento avanzado de todas y cada una de las tecnologías anteriores.

En las aplicaciones web tradicionales, las acciones del usuario en la página (pinchar en un botón, seleccionar un valor de una lista, etc.) desencadenan llamadas al servidor. Una vez procesada la petición del usuario, el servidor devuelve una nueva página HTML al navegador del usuario.

Esta técnica para crear aplicaciones web funciona correctamente, pero no crea una buena sensación al usuario. Al realizar peticiones continuas al servidor, el usuario debe esperar a que se recargue la página con los cambios solicitados. Si la aplicación debe realizar peticiones continuas, su uso se convierte en algo molesto .

AJAX permite mejorar completamente la interacción del usuario con la aplicación, evitando las recargas constantes de la página, ya que el intercambio de información con el servidor se produce en un segundo plano. Las aplicaciones construidas con AJAX eliminan la recarga constante de páginas mediante la creación de un elemento intermedio entre el usuario y el servidor. La nueva capa intermedia de AJAX mejora la respuesta de la aplicación, ya que el usuario nunca se encuentra con una ventana del navegador vacía esperando la respuesta del servidor.

Desde su aparición, se han creado un gran número de aplicaciones web basadas en AJAX. Algunos ejemplos son gestores de correo electrónico como Gmail o Yahoo Mail, aplicaciones de cartografía como Google Maps y otros como Google Docs y Flickr.

10 XMLHttpRequest (XHR), es una interfaz empleada para realizar peticiones HTTP y HTTPS a servidores Web. Para los datos transferidos se usa cualquier codificación basada en texto, incluyendo: texto plano, XML, JSON, HTML y codificaciones particulares específicas.

2.4. PhoneGap

PhoneGap es un framework que nos brinda la posibilidad de crear aplicaciones que se pueden compilar para diferentes plataformas móviles (iOS, Android, Blackberry, Windows Phone, WebOS y Symbian, entre otras). La gran ventaja de este producto es que estas versiones, pueden crearse a partir de un código desarrollado en HTML 5, CSS y JavaScript.

PhoneGap cuenta con una librería JavaScript que nos da una API de funciones que nos permitirá potenciar lo que nos ofrece el navegador. Es decir, podremos tener funcionalidades más allá del estándar para acceder, por ejemplo, a alguna opciones del sistema. Dentro de lo que es PhoneGap, también encontraremos librerías nativas que funcionan como una especie de "puente" entre el código JavaScript y cada una de las plataformas nativas, esto les da a los desarrolladores un elemento extra para ir un paso más allá con sus aplicaciones y saltar limitaciones impuestas.

En conclusión, si sabemos trabajar con HTML5 + CSS + JavaScript y aprendemos a utilizar todo lo relacionado con PhoneGap, podremos crear aplicaciones que se comportarán como aplicaciones nativas en diferentes plataformas móviles. Lo que deberemos tener en cuenta es que este producto se apoya también en el navegador principal de cada una de estas plataformas. Esto en parte, nos puede dar alguna diferencia entre las plataformas y también algunas características que deberemos tener en cuenta. Por esta razón es importante leer la documentación de PhoneGap, que es muy completa, y nos permitirá saber que cosas podremos utilizar con confianza en cada una de las plataformas que elijamos compilar.

Para finalizar, vale destacar que, a pesar que existen diferentes herramientas y/o plugins para utilizarlo, Phonegap es gratuito y *Open Source*[11].

2.5. LAMP

LAMP es el acrónimo de un conjunto de tecnologías *Open Source* utilizadas para definir la infraestructura y programación de un servidor web de propósito general. LAMP viene de Linux (sistema operativo), Apache (tecnología del servidor), MySQL (gestor de las bases de datos) y PHP (lenguaje de programación). Estas cuatro tecnologías son las que conforman la arquitectura del servidor que ejercerá de elemento central de nuestro sistema.

Aunque la programación definitiva del lado del servidor no ha sido objeto de este PFC, sí que lo fue la programación del servidor en el periodo de desarrollo de la aplicación, por lo que el aprendizaje de las tecnologías anteriormente mencionadas fue significativo, especialmente en PHP y MySQL.

PHP (acrónimo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para desarrollo web (lado-servidor) y que puede ser incrustado en HTML. Lo que distingue a PHP de algo lado-cliente como Javascript, es que el código es ejecutado en el servidor, generando HTML y enviándolo al cliente. El cliente recibirá los resultados de ejecutar el script, sin ninguna posibilidad de determinar qué código ha producido el resultado recibido.

11 Código abierto, es decir, software desarrollado y distribuido libremente.

MySQL[12] es el servidor de bases de datos relacionales más popular, desarrollado y proporcionado por MySQL AB. Una base de datos relacional es una colección de datos estructurados en tablas separadas en lugar de poner todos los datos en un solo lugar. Esto agrega velocidad y flexibilidad. Las tablas son enlazadas al definir relaciones que hacen posible combinar datos de varias tablas cuando se necesita. Para agregar, indexar, y procesar los datos almacenados en una base de datos, se necesita un

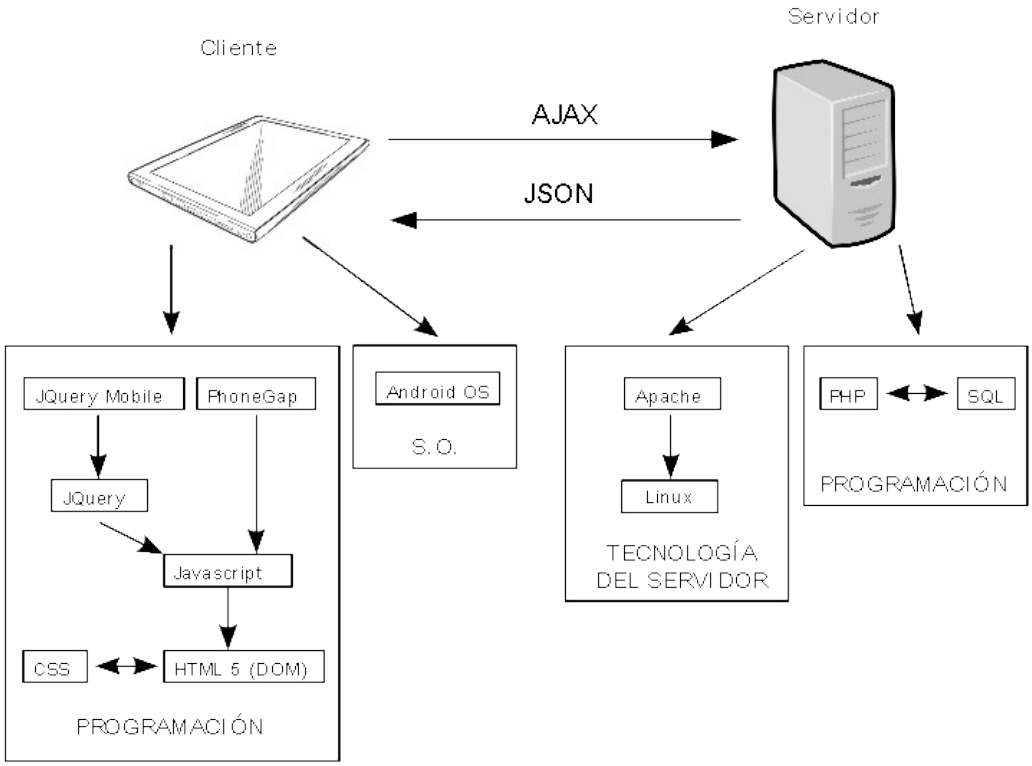

Figura 3. Diagrama de tecnologías usadas en el proyecto

sistema de administración de bases de datos, tal como MySQL.

El servidor de bases de datos MySQL es muy rápido, seguro, y fácil de usar. Además, MySQL es *Open Source*, por lo que cualquiera puede descargar el software de MySQL de Internet y usarlo, estudiarlo y cambiarlo sin pagar por ello.

12 SQL significa Structured Query Language (lenguaje de consulta estructurado). "My" proviene del nombre de la hija del creador del lenguaje, My.

Capítulo 3. DESARROLLO DE LA APLICACIÓN

En este capítulo se tratará todo lo relativo al desarrollo de la aplicación móvil y sus diferentes elementos, núcleo fundamental de este proyecto fin de carrera. Se tratará de incidir especialmente en los aspectos que doten a este proyecto de una dimensión relevante desde el punto de vista de la ingeniería, es decir, todos aquellos que sean claves en la resolución de un problema planteado por la necesidad de un servicio telemático.

3.1. Escenario. Consideraciones previas.

Como ya se ha comentado anteriormente, nuestra aplicación tendrá que trabajar en un entorno de baja conectividad, a pesar de lo cual tendrá que seguir operativa de manera fiable. Pero es que aunque se disponga de conexión hay que tener en cuenta que el sistema será utilizado en México, donde las tarifas de datos tienen un gran coste y permiten anchos de banda pequeños.

Esto nos condiciona de tal manera que hemos de tratar de aprovechar ese ancho de banda de la mejor manera posible, tratando de utilizar los momentos en los que dispongamos de red Wi-Fi para el envío de los datos más pesados y no imprescindibles (imágenes).

Además de ésto, debemos tener en cuenta una serie de condiciones de nuestro cliente. Se trata de una empresa contratada para la gestión de algunas de las carreteras más importantes de México (Atlacomulco-Maravatío, Sinaloa) y desde el ministerio de aquel país se les imponen unos plazos de actuación en ocasiones muy ajustados que deben cumplir estrictamente. Los operarios de la empresa que realizan las tareas de mantenimiento tienen en su contrato de trabajo cláusulas por las cuales pueden ser económicamente penalizados si incurren en sucesivas faltas.

Es por éstas razones por las que se ha tratado de hacer especial hincapié en la necesidad de realizar una aplicación ágil, que permita acceder a la información y manipularla en muy poco tiempo ya que los plazos son en ocasiones, de una hora, en la cual el operario debe darse cuenta de la nueva tarea, desplazarse hasta el lugar de la incidencia y rellenar los formularios adecuados.

Con respecto a la tecnología de que disponen los operarios que harán uso de este sistema, disponen del terminal Samsung Galaxy Note, que proporciona un adecuado marco para el uso de la aplicación. Actualmente el sistema operativo utilizado es la versión 2.3.5 de Android, aunque en un futuro próximo la versión será la 4.0.4.

3.2. Arquitectura y elementos de la aplicación

En este apartado veremos todas las pantallas y posibilidades que ofrece la aplicación y la lógica subyacente que hay detrás de cada una.

Además, se mencionarán en multitud de ocasiones las tablas de la base de datos del dispositivo y las variables de la caché local. Podemos encontrar un listado y descripción de toda esta información en el anexo B.

3.2.1. Registro (login) y salida (logout) en el sistema

Evidentemente, la agenda de vialidad y en general, todo el sistema SGWC es un sistema cerrado, por lo que debemos permitir el acceso sólo a ciertos usuarios. El sistema de registro permite al usuario identificarse para poder trabajar con la aplicación.

Pantalla de registro

Para realizar el registro (login), el sistema requiere de dos campos: nombre de usuario y contraseña. Estos datos son enviados al servidor con tecnología AJAX (es decir, en modo asíncrono) mediante el método POST (todas las peticiones al servidor que realiza esta aplicación utilizan el método POST), y el cliente (aplicación) recibe una respuesta en formato JSON (todas las respuestas recibidas del servidor tienen este formato) que contiene, entre otros, un campo que nos indica si el registro del usuario se ha recibido con éxito (OK) en el servidor o no (NOK). En caso afirmativo, se procede a guardar la información de registro del usuario (nombre de usuario y contraseña) en la tabla de usuarios de la base de datos del dispositivo, y avanzamos al menú principal de la aplicación.

Figura 4. Pantalla de registro

Además, cuando nos registramos con éxito en el sistema, cierta información (permisos de usuario, sector asociado a dicho usuario y otros) se guarda en la caché local del navegador (localStorage, una característica de HTML 5).

Cabe destacar que la tabla de usuarios sólo guarda un único usuario, el último que se registró. Se adoptó esta solución por seguridad, de manera que no será posible recuperar información sensible (contraseña) de otros usuarios que anteriormente hayan utilizado el terminal, siempre y cuando hayan realizado la salida (logout) correctamente.

Igualmente, comentar que la aplicación utiliza en todo momento (no sólo para el registro) conexiones mediante certificados de seguridad SSL, por lo que todos los

datos sensibles se transmiten de forma segura.

Registro automático (autologin)

En nuestro sistema, se tuvo en cuenta también que la agilidad y comodidad en la aplicación móvil son muy valorables y en algunos casos, necesarias, dado que podemos tener plazos temporales que cumplir. En este contexto, nació la idea del módulo de autorregistro (autologin). Dicho módulo verifica al abrir la aplicación que existen datos de usuario en la tabla correspondiente y realiza el registro en el servidor de manera transparente para el usuario, llevándonos al menú principal sin pasar por la pantalla de registro. Si existiese algún problema durante el autorregistro, entonces sí que pasamos por la pantalla de registro, a fin de conseguir un ingreso correcto en el sistema.

Ausencia de conexión

Se dedicará un apartado específico para explicar el comportamiento del sistema en ausencia de cobertura, dado que es uno de los problemas fundamentales a los que se enfrenta nuestro servicio, como ya hemos comentado en varias ocasiones.

En el caso del registro, distinguiremos dos situaciones: el caso de que estemos tratando de registrarnos en la pantalla correspondiente, y el caso del registro automático, dado que presentan diferencias importantes.

En el registro automático

En este caso, si disponemos de información almacenada en la tabla de usuarios, será posible acceder al sistema, manteniendo toda la configuración (permisos, etc) asociada al usuario que teníamos almacenado. Es decir, tenemos el mismo comportamiento que en el caso de un autorregistro con conexión, pero la aplicación mostrará un mensaje para avisarnos de dicha situación (ver figura 5).

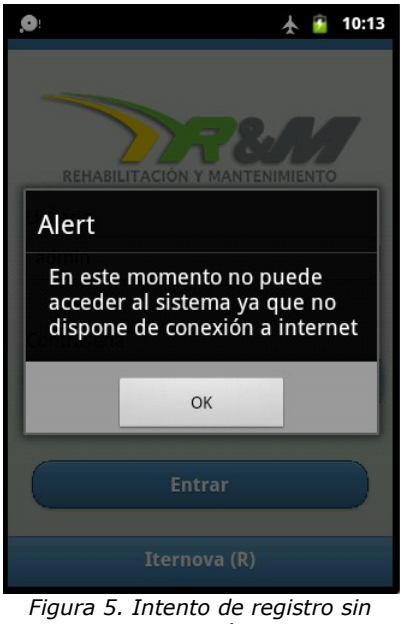

Figura 5. Intento de registro sin conexión

Esto significa que el servidor no sabe que nosotros estamos dentro del sistema. Esta situación se corrige cuando nuestro terminal entra en una zona de cobertura y realiza una petición de cualquier tipo al servidor. Podríamos pensar que dicha petición será rechazada por el servidor al provenir de un usuario que aparentemente no está registrado, sin embargo, cada vez que se realiza una petición al servidor de cualquier naturaleza hacemos primero una petición de autorregistro, de manera que el servidor siempre tiene información actualizada sobre qué usuarios están activos.

En la pantalla de registro

Si al tratar de enviar los datos de usuario al servidor no disponemos de conexión a Internet, no será posible acceder al sistema, dado que estar en dicha pantalla significa que la base de datos no contiene información sobre ningún usuario (bien porque el último cerró su sesión correctamente o porque es la primera vez que ejecutamos la aplicación).

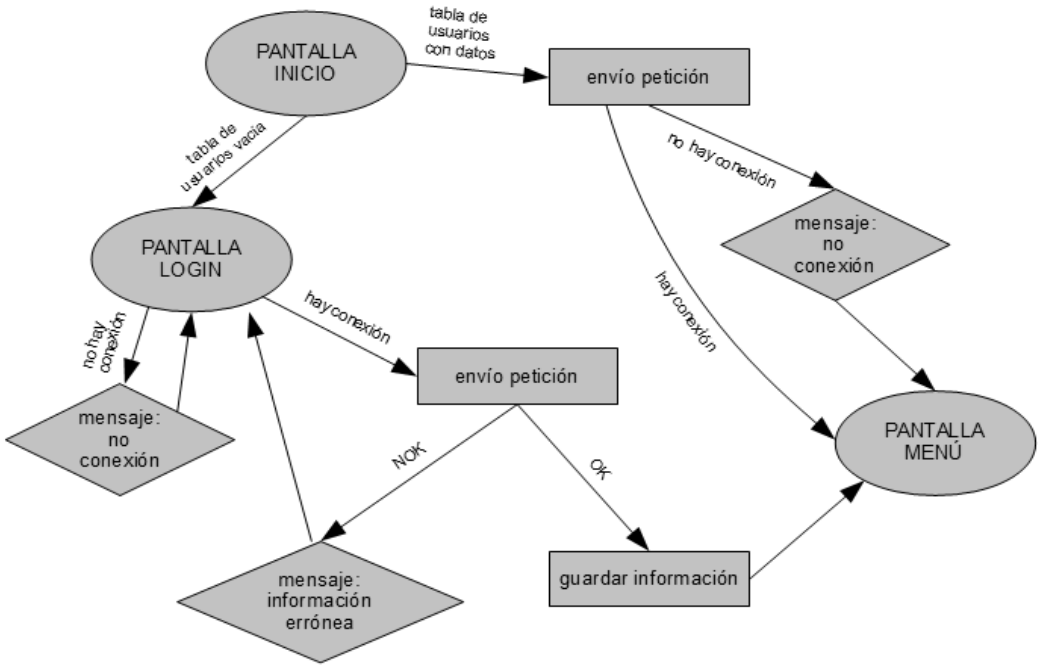

Figura 6. Proceso de registro de usuarios

Salida (logout)

Como ya hemos comentado anteriormente, si el usuario realiza un de-registro la tabla de usuarios borrará la información contenida y la aplicación nos llevará a la pantalla de registro. Además, toda la información relativa al usuario almacenada en la caché local (permisos, lista de tareas, etc) será eliminada también, para no interferir con el próximo usuario que se registre.

Figura 7. Pantalla del menú principal

3.2.2. Menú principal

Una vez registrados correctamente en el sistema, pasamos al menú principal. Como podemos apreciar en la figura 7, este módulo es un simple distribuidor de las posibilidades que ofrece nuestra aplicación:

Sí que es interesante comentar un punto relativo al

sistema de permisos de usuario y es que el botón "Dar de alta incidencia" no estará disponible para los usuarios que no tengan dicho permiso.

3.2.3. Lista de tareas

Nos encontramos ahora frente al elemento central de nuestro sistema. Se trata de un listado de elementos, cada uno de los cuales se corresponde con una "tarea" o "incidencia" que el usuario debe resolver en un plazo determinado.

Podemos observar dos botones, "actualizar lista" y "refrescar lista". El primero realiza una recarga completa de la tabla de tareas de la base de datos de la aplicación. Es decir, envía una petición (POST) al servidor y recibe de éste la lista completa en formato JSON. Como se ha comentado para el caso del autorregistro, todas las peticiones al servidor van precedidas de otra que registra al usuario. De esta manera, mantenemos el servidor con información actualizada de nuestro estado.

En la petición de la lista de tareas, se envía como parámetros la sección (grandes áreas que pueden cubrir varias carreteras) y el tramo de carretera para los cuales se desean recibir las tareas pendientes. Estos dos parámetros son configurables en el submenú de opciones (se verá más adelante).

El botón "refrescar lista" simplemente realiza una carga de las tareas almacenadas en la base de datos del terminal mostrándolas por pantalla, sin necesidad de conectarse con el servidor.

En la lista de tareas propiamente dicha, vemos cómo para cada tarea se muestra un resumen de la información de la misma con los datos más importantes, ID de la tarea, título de la incidencia, localización, plazos, etc. Podemos pulsar sobre cada una de dichas tareas, lo que nos llevará a la ficha de dicha incidencia en concreto.

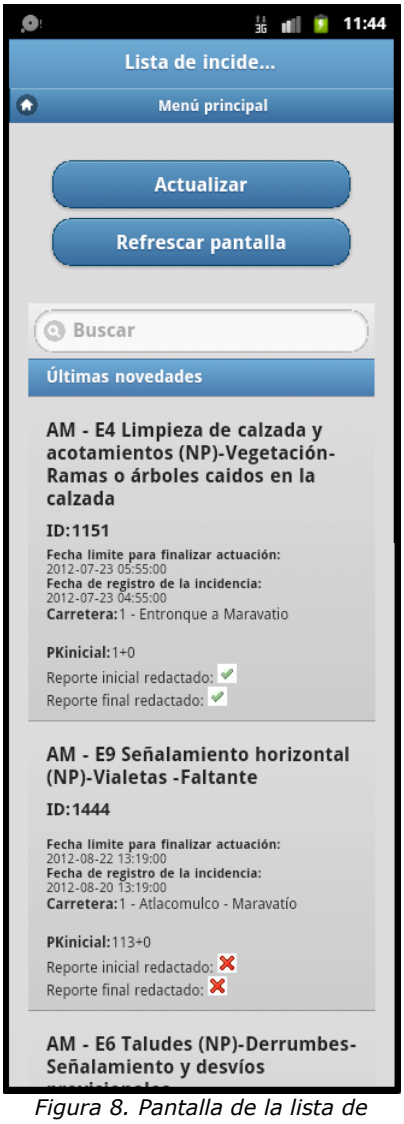

Figura 8. Pantalla de la lista de tareas

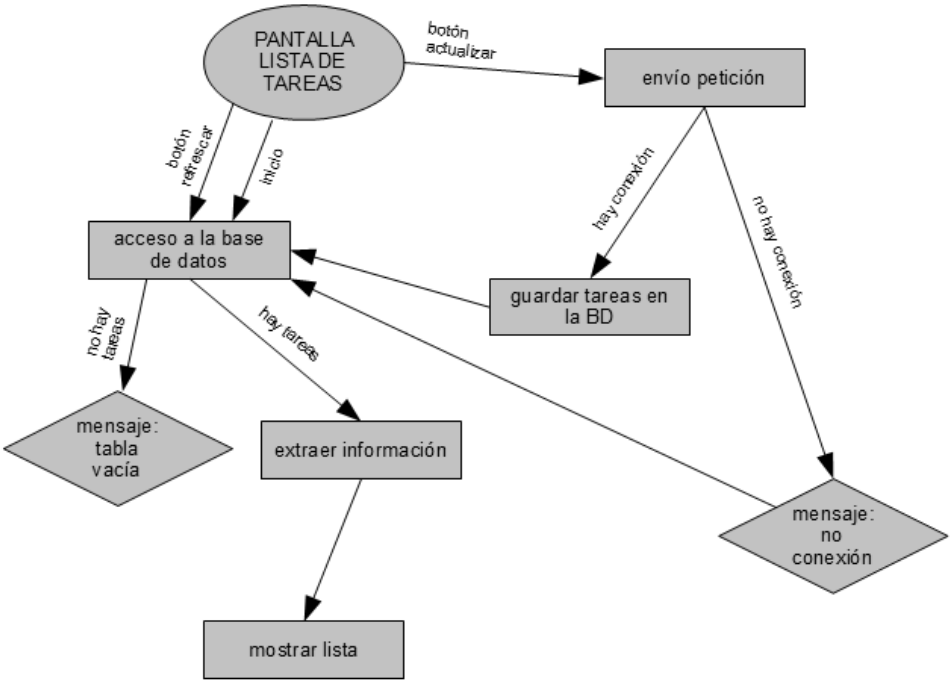

Figura 9. Diagrama de funcionamiento de la lista de tareas

Actualización automática de la lista de tareas

Como podemos comprobar, la lista de tareas es el elemento esencial de nuestro sistema. Es de vital importancia pues, que se mantenga actualizado y disponible en cualquier situación.

Para ello, se diseñó un módulo que realiza una actualización automática de la lista de tareas en la base de datos, realizando una petición al servidor para recibir los datos. Sin embargo, tenemos la constancia de que nuestra aplicación debe ser lo más ligera y eficiente posible, a la vez que funcional. Esto nos lleva a considerar los siguientes puntos:

- Nuestra aplicación funcionará en entornos de baja cobertura, problema que se agrava si tenemos en cuenta que el cliente de nuestra empresa trabajará en un país (México) donde la transmisión de datos 3G son, a menudo, cara y lenta. Es necesario pues, ahorrarnos la transmisión de datos siempre que sea posible.
- A menudo el tiempo disponible para realizar las actuaciones pertinentes es escaso (1h en algunos casos) debido a exigencias de las autoridades competentes. Si a nuestros clientes se les exige cumplir con los protocolos establecidos, nosotros haremos lo posible por proporcionar un servicio rápido y eficiente.

- Nuestra aplicación debe mantenerse simple, aprovechando el máximo código posible para hacerla fácil de mantener ante eventuales cambios en las exigencias de nuestro cliente.

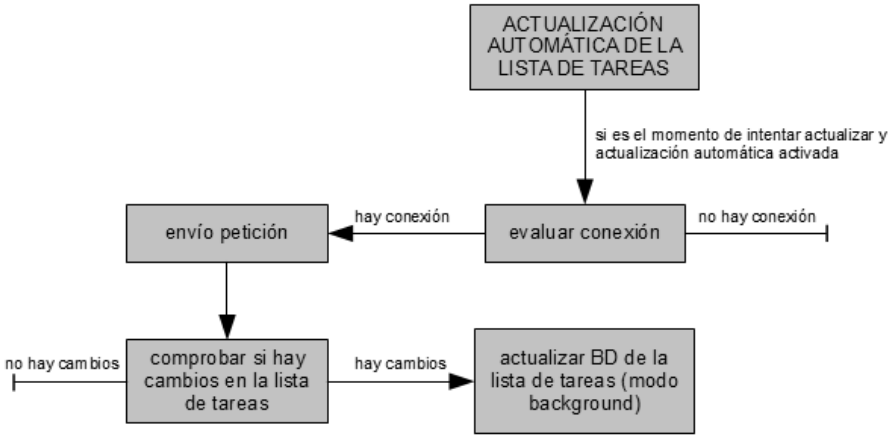

Figura 10. Gestión de un intento de actualización automática de la lista de tareas.

Ante estos retos, se realizó un módulo de actualización automática con las siguientes características:

- El núcleo de dicho módulo es la misma función de actualización de la lista de tareas que ejecutamos cuando pulsamos el botón correspondiente (ver figura 10). Simplemente, esta función tiene un parámetro que indica el modo en el que se ejecuta, pudiendo ser modo "normal" (cuando pulsamos el botón) o "background", de manera que en modo "background" no mostraremos mensajes que podrían aparecer y que no tienen sentido en un proceso que debe ser transparente al usuario, tales como errores del servidor u otros. Nuestro módulo debe gestionar correctamente estas situaciones.
- El módulo de actualización automática envía primero una petición al servidor, el cual responde con la fecha y hora de la última modificación de la lista de tareas para el sector seleccionado por el usuario. Esta fecha y hora se compara con la que el terminal tenía almacenada en su caché local.
    - Si es distinta, se actualiza con la nueva fecha y hora y se actualiza la base de datos del dispositivo tras enviar la petición correspondiente. Una vez actualizada con éxito, se mostrará por pantalla el mensaje

Figura 11. Notificación de nuevas tareas durante el uso.

"Existen nuevas tareas" acompañado de vibración y una alerta sonora, para que el operador sea consciente de los cambios.

- Si es la misma, la petición de la lista de tareas no se llega a enviar. De esta manera, ahorramos en datos transmitidos si no son necesarios.

- La actualización automática de la lista de tareas puede ser configurada en la pantalla de opciones (ver apartado 3.2.7 Opciones). Podemos desactivar totalmente este evento o bien controlar la frecuencia con la que ocurre, en un rango de entre 2 y 30 minutos. Hay que tener en cuenta que, como ya hemos visto anteriormente, los plazos de resolución de la incidencia son a menudo de menos de 1 hora (lo que incluye tiempo para llegar al lugar de la incidencia y solucionarla) por lo que se requiere actualizaciones cada pocos minutos para no perder tiempo.

3.2.4. Ficha de la tarea

Al pulsar sobre alguna de las tareas de la lista, accederemos a su ficha, que muestra todos los detalles sobre dicha incidencia. Podemos ver entre otros, un mapa de Google Maps con la geolocalización, si los informes de iniciación y finalización están completados y el resto de detalles.

En esta sección se realiza una gestión de las posibilidades de las que dispone el usuario. Estas posibilidades vendrán marcadas primeramente por los permisos de los que dispone y en segundo lugar, por el grado de compleción en el que se encuentre la tarea.

Cada una de las opciones cuya disponibilidad varía se describen a continuación:

- Iniciar/finalizar actuación. Si pulsamos uno de estos botones nos llevan a la redacción del informe o reporte de actuación correspondiente (inicial, final). La posibilidad de redactar estos informes viene dada por los permisos de usuario, por lo tanto, para algunos de ellos no aparecerán estos botones y no será posible redactar dichos informes.

  Adicionalmente, el sistema gestiona que no se produzcan situaciones no deseadas o inverosímiles. Por ejemplo, no es posible redactar el informe de finalización si no está redactado primero el informe de iniciación.

  Además, si tratamos de redactar un informe que el operario ya había enviado, el sistema debe detectarlo y no permitir su modificación ni la creación de un nuevo informe para la misma incidencia. Esto es así debido a petición expresa

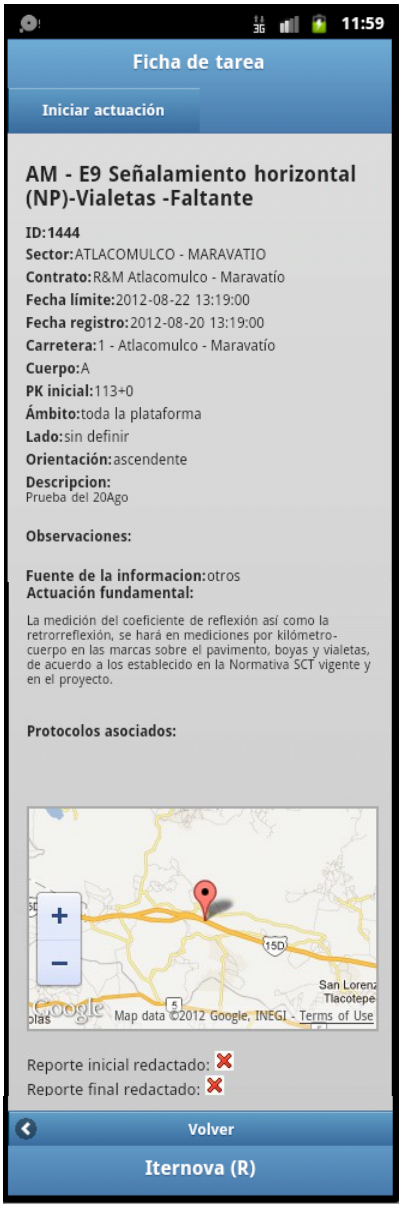

Figura 12. Pantalla de la ficha de la tarea

de nuestro cliente, quien consideraba esencial que los usuarios no pudiesen realizar acciones que diesen lugar a situaciones ambiguas ni modificar datos ya enviados previamente.

- Validar. Este botón aparecerá cuando ambos informes (iniciación y finalización) estén listos y a la vez, el usuario tenga el permiso correspondiente. La validación consiste en el visto bueno del usuario (normalmente un supervisor de los operarios) sobre el trabajo realizado (informes) para la tarea correspondiente. Es una simple petición al servidor, que recibe como respuesta si la validación ha quedado registrada en el servidor o no.

En la figura 12 vemos como no aparecen ni el botón de finalizar actuación (debería aparecer a la derecha del de iniciar actuación) debido a que el reporte inicial no está redactado (tal como marca el icono debajo del mapa). El botón validar tampoco aparece (debería aparecer encima del botón ”volver”), ya que no hemos redactado los informes, aunque en caso de que estuvieran podría no aparecer si no tuviésemos el permiso necesario.

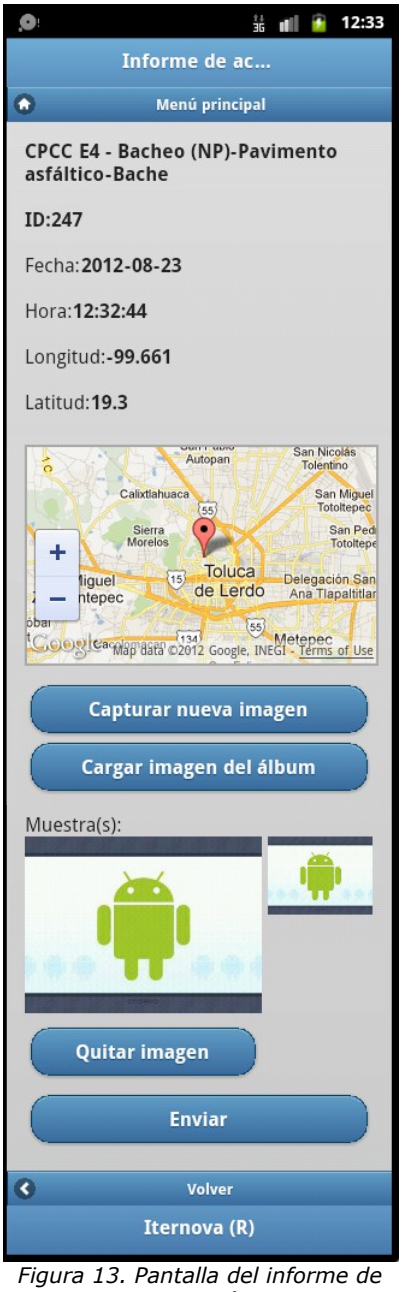

Figura 13. Pantalla del informe de actuación

3.2.5. Informes de actuación

Si pulsamos sobre alguno de los botones de iniciar o finalizar actuación llegaremos a una pantalla donde se pueden dar diferentes casos:

- El informe ya había sido redactado por el usuario actual, por lo tanto se encuentra en nuestra base de datos.
- El informe había sido redactado por otro usuario.
- El informe está pendiente de realizar.

En el primer y segundo caso simplemente se mostrará el informe completo, sin posibilidad de modificarlo ni de reenviarlo.

Nos centraremos de momento en el tercer caso. En el subapartado "Gestión de informes en el sistema" se estudiarán más a fondo las otras situaciones. Por ahora, comentaremos los campos a rellenar en el informe:

- Fecha y hora. Automáticamente se toman los datos del dispositivo.
- Geolocalización. Se toman los datos del sistema GPS, también de manera automática. Cuando la geolocalización se completa con éxito, se muestra un mapa de Google Maps con la situación del usuario.
- Posibilidad de tomar imágenes de la incidencia. Tendremos opción de, o bien tomar nuevas instantáneas con la cámara integrada en el

terminal, o bien cargar una imagen del álbum. En cualquier caso, la cantidad de imágenes incluidas en nuestro informe no podrá ser superior a 3, si bien se incluye un sencillo gestor que nos permite desincrustar alguna imagen que no convenza al usuario para sustituirla por otra.

- Solamente para el informe de finalización de actuación, se incluye un campo de observaciones en el que podemos realizar una breve descripción del trabajo realizado.

Es destacable el hecho de que algunos datos como la hora, fecha, y geolocalización se toman directamente del sistema y no es posible modificarlos por parte del usuario. Esto, como ya hemos comentado anteriormente, viene dado por la necesidad de un cumplimiento estricto de los plazos, teniendo así que acabar con cualquier posibilidad de un uso malintencionado por parte del usuario (colocar una hora anterior a la real para que parezca que la tarea se atendió más pronto, por ejemplo).

Proceso de envío de informes

Una vez que hayamos completado los campos e insertado las imágenes necesarias, el siguiente paso es pulsar sobre el botón "enviar". En ese momento, se activa una cadena de eventos que se describen a continuación:

- Se guarda la información del reporte en la tabla "reportes" de la base de datos.
- Si tenemos éxito al guardar, se extraen todas las entradas de la tabla de reportes (iniciación, finalización y validación) que estén pendientes de ser enviadas.
- Enviamos los datos de los informes al servidor.
- Si el envío tiene éxito, revisamos la respuesta del servidor, donde podemos comprobar individualmente para cada elemento enviado si se guardó con éxito en el servidor o no. En caso afirmativo, procedemos a actualizar nuestra base de datos local, marcando como "datos enviados" las entradas correspondientes.
- Se procede a enviar las imágenes pendientes de enviar (las del informe actual y las que estuvieran pendientes). En este caso, comprobamos que disponemos de una conexión Wi-Fi (ver apartado 3.2.7 Opciones). Si disponemos de ella, procedemos a enviar las imágenes en un procedimiento similar al de los datos, marcando como "imágenes enviadas" las entradas de la tabla cuyas imágenes han sido enviadas satisfactoriamente.

Durante el desarrollo del sistema, se decidió enviar de manera separada los datos y luego las imágenes. Esto es así debido las limitaciones de cobertura que presenta el contexto en el que se utilizará la aplicación. Los datos son lo realmente prioritario a la hora de resolver las tareas, por tanto no podemos depender del correcto envío de las imágenes (mucho menos fiable, y más costoso tanto en cantidad de datos como en tiempo) para poder marcar un informe como válido. Por tanto, es necesario dar la oportunidad de prescindir del envío de las imágenes si supone una pérdida de agilidad para nuestro sistema. Es ésta la razón de la separación del envío de datos e imágenes en procesos distintos.

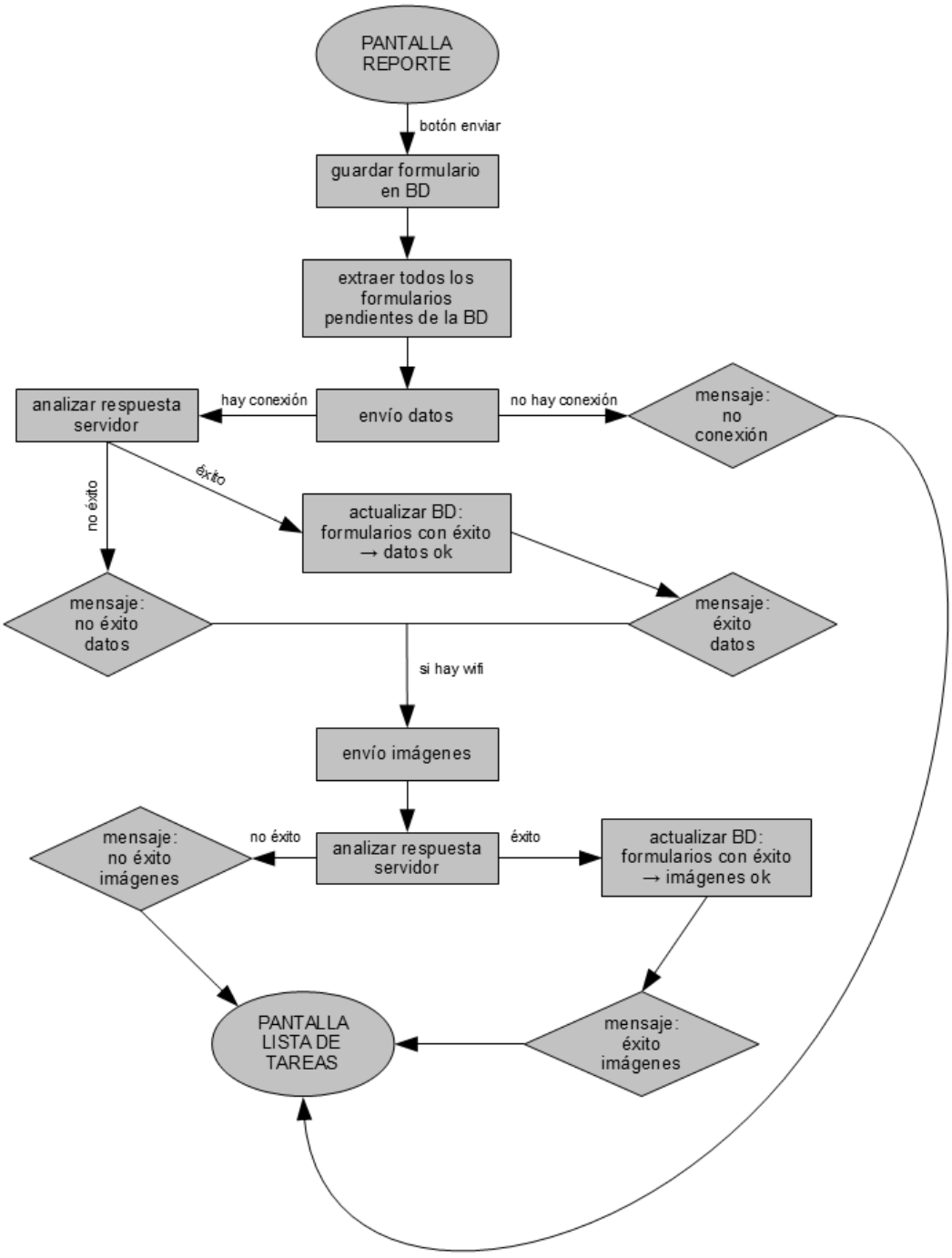

Figura 14. Proceso de envío de informes

Ausencia de conexión

Probablemente el envío de los informes de actuación sea el punto más sensible en situaciones de ausencia de conexión, por lo tanto, nuestro sistema debe ser capaz de gestionar esta situación para que no se produzcan duplicidades ni pérdidas de información.

Cuando pulsamos el botón "enviar", hemos visto que se produce una secuencia de eventos. Durante estos eventos se realizan varias peticiones al servidor que, en caso de falta de conexión, obviamente no serán posibles. Para evitar que la información que habíamos introducido en el formulario se pierda sin más, guardamos primero nuestra información en la base de datos marcándola como "no enviado". Sólo si alcanzamos el servidor y la información se guarda en éste con éxito, marcaremos nuestros datos como "enviados" para que el sistema no trate de volver a enviarlos posteriormente. Lo mismo ocurre con las imágenes.

Esto dota a nuestra herramienta independencia de la situación en la que se encuentre, de manera que el usuario puede continuar trabajando sin preocuparse de disponer de conexión o no, ya que la aplicación gestiona la situación en la que se encuentra.

Gestión de informes en el sistema

Tanto en la lista de tareas como el la ficha de cada incidencia podemos observar dos iconos que muestran si los informes de iniciación y finalización han sido completados o no (ver figuras 8 y 12). Existen dos posibilidades de que un informe haya sido rellenado. La primera es que otro operario lo haya hecho desde otro terminal. La segunda es que lo haya hecho el propio usuario. Esto nos lleva a plantearnos varios escenarios:

1. Otro operario ha enviado un informe y tenemos la base de datos actualizada con esos últimos cambios.
2. El propio usuario ha enviado un informe al servidor con éxito y ha actualizado la lista de tareas posteriormente.
3. Otro operario ha enviado un informe al servidor, pero nosotros no tenemos cobertura o bien no hemos cargado la última actualización de la lista de tareas.
4. El propio usuario ha redactado un informe, pero no ha podido enviarlo por falta de cobertura o bien lo ha enviado pero no ha actualizado la lista de tareas.

La situación en los escenarios 1 y 2 es clara y sencilla. Como la base de datos del terminal cuenta con la lista de tareas actualizada con los últimos cambios, la información sobre los informes ya redactados se encuentra en la tabla de tareas. Si tratamos de redactar el mismo informe, la aplicación nos avisará de que ya está hecho, y nos mostrará el informe sin posibilidad de modificarlo ni de reenviarlo.

Sin embargo, las otras dos situaciones (3 y 4) requieren algo más que la consulta de nuestra lista de tareas. En dicha lista aparecen como no realizados informes que puede que sí que lo estén, bien por nosotros o bien por otros usuarios. ¿Cómo gestionará nuestro sistema estas situaciones? Distinguiremos dos casos:

1. Si el informe fue redactado por el propio usuario, está almacenado en la tabla correspondiente. Rescataremos la información de dicha tabla.

2. Si el informe fue enviado por otro usuario, no tenemos manera posible de saber que ya fue redactado. Aquí entra el juego la gestión de los informes en el servidor, de tal manera que ante el envío de un informe que ya está registrado el servidor devuelve una respuesta satisfactoria como si nuestro envío fuese normal y corriente, aunque el servidor descartará redacciones posteriores del mismo informe.

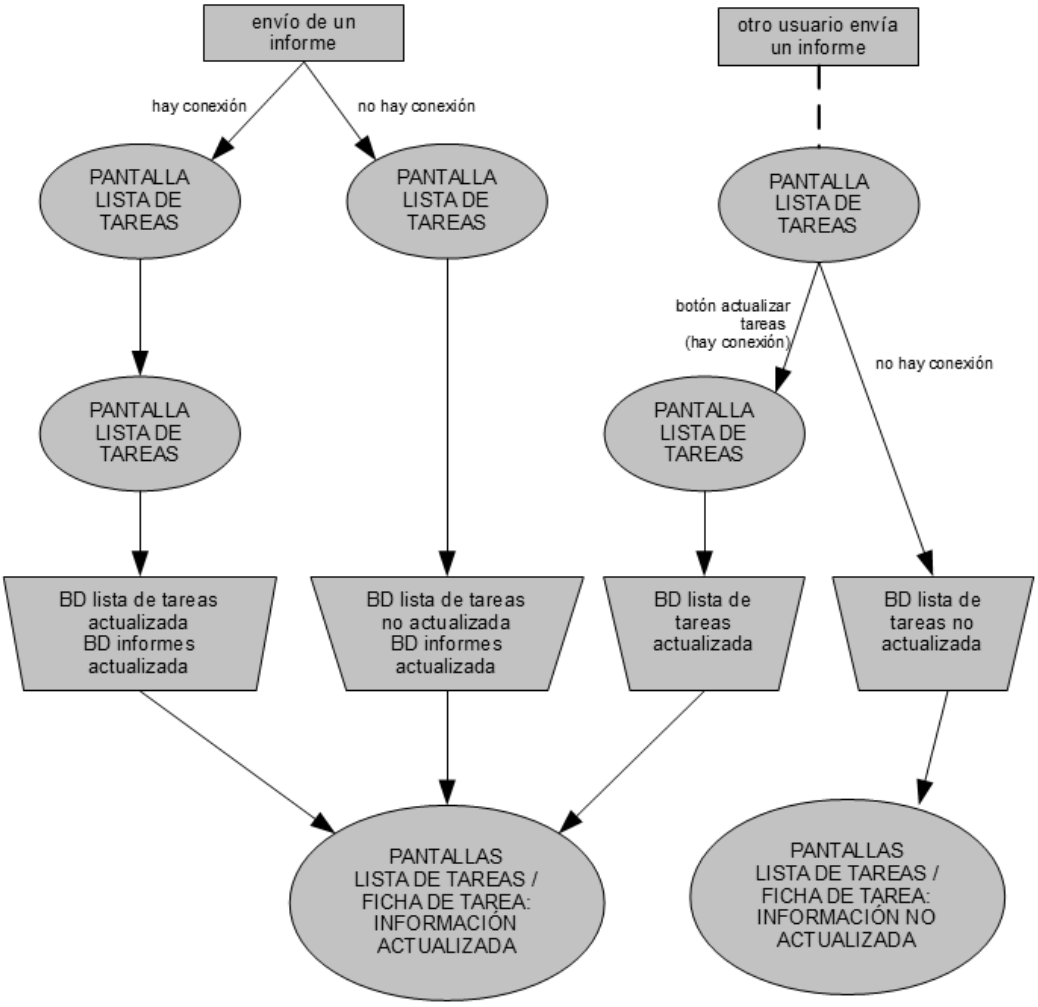

Figura 15. Gestión de informes de actuación en la BD

3.2.6. Alta de nueva tarea

Así como la lista de tareas contiene las incidencias ya creadas que deben resolverse, es posible también dar de alta nuevas tareas en el sistema. Para ello, pulsaremos sobre el botón "Dar de alta incidencia" en el menú principal. Como ya se ha comentado anteriormente, esta es una operación de cierta importancia, por lo tanto sólo algunos usuarios tendrán el permiso necesario para realizarla.

El alta de una nueva tarea es un extenso formulario donde existen numerosos selectores desplegables entre cuyas opciones el usuario debe elegir. Todos los campos obligatorios (marcados con '*') deben rellenarse, de lo contrario no se permitirá enviar el formulario.

Además, en el servidor se comprueba si ya existe la tarea previamente (mediante comprobación de varios atributos, entre los que se encuentran la carretera, plataforma, puntos kilométricos, tipos de incidencias, fechas...), por lo que no se sobrescriben datos en el mismo en caso de que ya exista. En este caso, el servidor devolverá el resultado como si no existiera la tareas previamente, por lo que la aplicación móvil podrá actualizar el registro de la base de datos según corresponda.

Lo más destacable de esta sección es que las distintas opciones entre las que podemos elegir en los selectores no son estáticas, sino que pueden variar según las necesidades del cliente. Esto es gracias a que existen una serie de tablas en la base de datos que contienen la información relativa a dichos menús desplegables. Un módulo se encarga de realizar una petición al servidor de esa información y actualizar las tablas con el nuevo contenido cuando sea necesario (ver apartado 3.2.7 Opciones).

De esta manera conseguimos una gran flexibilidad, no siendo necesaria la reinstalación de la aplicación en caso de que, por ejemplo, se añadan nuevos tramos de carreteras o se incluyan nuevos tipos de incidencias, es decir, nuestro sistema es fácilmente escalable.

De la misma forma que para los otros formularios, la fecha, hora y geolocalización son tomadas por el dispositivo, impidiendo cualquier modificación por parte del usuario.

El proceso de envío del formulario generado es básicamente el mismo que en el caso de los reportes de iniciación/finalización. Así mismo, tampoco se volverá a explicar el caso de la ausencia de conexión, dado que es también análogo al caso de los informes.

En esta pantalla, además, se ha tenido en cuenta que debía ser fácilmente rellenable y utilizable, ya que muchas de las tareas las toman con el vehículo en marcha, por lo que una tarea debería estar completada y enviada al servidor en menos de 2 minutos. Hay que volver a remarcar la importancia de la rapidez de uso y las notificaciones instantáneas, ya que para solucionar cualquier incidencia hay un plazo de tiempo bastante ajustado.

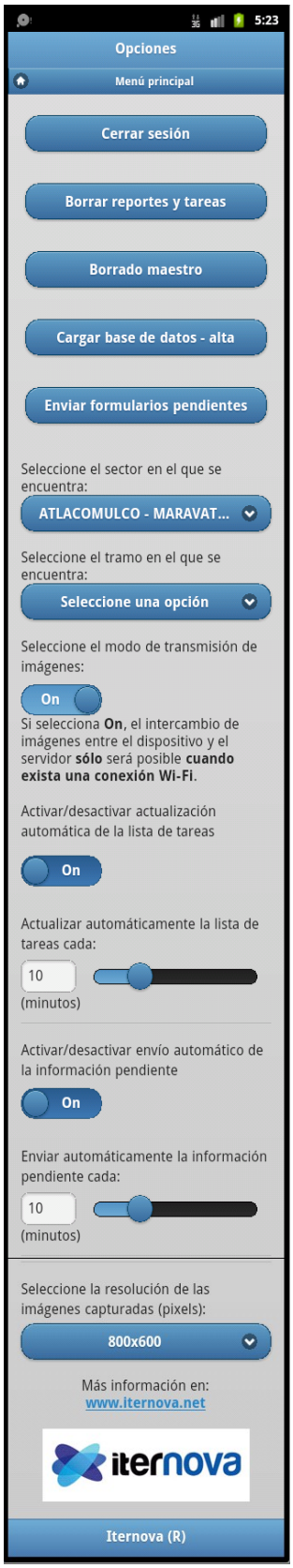

Figura 16. Pantalla de opciones

Envío automático de los informes pendientes

Así como la lista de tareas trata de actualizarse automáticamente cada cierto tiempo, el sistema incorpora también otro módulo similar para el envío automático de informes (iniciación, finalización, validación y altas de nuevas tareas). Este módulo extrae de las tablas correspondientes los informes de iniciación y finalización, y las validaciones (tabla "reportes") y las altas de incidencias (tabla "altas") que todavía no han sido registradas con éxito en el servidor.

Del mismo modo que en el caso de la lista de tareas, este módulo realiza sus operaciones en modo "background" es decir, sin interferir en las operaciones que esté realizando el usuario y sin mostrar ningún tipo de mensaje.

3.2.7. Opciones

Finalmente se describirá la tercera y última posibilidad que nos ofrece el menú principal.

Como podemos apreciar en la figura 17, en primer lugar podremos cerrar nuestra sesión como se ha comentado en el apartado "registro y salida en el sistema".

A continuación, tenemos la opción de realizar un mantenimiento de nuestra base de datos pulsando sobre "Borrar reportes y tareas", lo cual provocará que se eliminen las entradas de las tablas de reportes y altas que hayan sido enviadas con éxito al servidor y que se borre también la lista de tareas, siendo necesario que vuelva a traerse desde el servidor.

El tercer botón, como su propio nombre indica, provoca el restablecimiento de los valores por defecto de toda la configuración y el vaciamiento de todas las bases de datos. Por tanto, esta opción no debería de utilizarse habitualmente.

La siguiente opción produce la actualización de las tablas que contienen la información para dar de altas nuevas incidencias. Ya se ha hablado de ello en la sección anterior.

El último de los botones nos permite enviar todos los formularios almacenados hasta el momento (informes de iniciación y finalización, validaciones y nuevas altas). Es como el envío automático descrito en la sección inmediatamente anterior, pero con la diferencia de que al final mostrará un informe con el resultado del envío. En la situación más común, mostrará un mensaje informando de que toda la información fue enviada con éxito al servidor.

Más abajo encontramos 2 selectores que nos permiten establecer el sector y el/los tramo(s) de carretera en el que estamos trabajando. Estos dos parámetros, como ya vimos, son enviados junto con la petición de la lista de tareas, obteniendo así las incidencias asociadas a dicho sector y tramo(s).

Lo siguiente es la configuración de envío de imágenes. Por defecto, estas se enviarán sólo al encontrar una conexión rápida (red Wi-Fi), aunque podemos establecer que se utilice también la red 3G para este fin (habrá que tratar de enviar imágenes de poco tamaño).

Después tenemos la opción de activar o desactivar la actualización automática de tareas y el envío automático de formularios, junto con un control deslizante para configurar la frecuencia de estas operaciones.

Finalmente, se incluye un selector para establecer el tamaño de las imágenes capturadas por la cámara, de manera que podemos rebajar los datos enviados en redes de poca capacidad.

3.3. Plan de pruebas del sistema

A continuación se detalla el plan de pruebas seguido para comprobar el buen funcionamiento del sistema y que cumple con los requisitos que se le exigían.

Aunque durante el desarrollo de la misma se iban haciendo pequeñas pruebas para comprobar el funcionamiento de cada una de las partes desarrolladas, era necesario realizar un plan de pruebas global para verificar el comportamiento de cada una de las partes de la aplicación, del global de la misma, y de su integración con el sistema en su conjunto.

El método de comprobación del cumplimiento del plan de pruebas se reforzó con los ensayos del cliente del sistema, que contribuyó en buena medida a detectar problemas ya que contaba con el entorno real en el que el sistema ha de trabajar, además de sus propias simulaciones.

Gracias al empeño puesto en el desarrollo del sistema, a la rigurosidad del plan de pruebas al que fue sometido y la aportación de nuestro cliente ha sido posible desarrollar este sistema en su versión más robusta, fiable y eficiente.

El plan de pruebas se dividirá en varias partes, cada una enfocada en un módulo por motivos de claridad. La metodología del desarrollo de las pruebas no se basó solamente en comprobar cada módulo por separado sino que también se realizaron pruebas para ensayar aspectos que requerían la cooperación de varios de los módulos realizados (por ejemplo, el envío automático de los informes).

3.3.1. Instalación e inicio de la aplicación

REF	Proceso	Resultado esperado
A. 01	Instalación de la aplicación	Debe instalarse correctamente, teniendo en cuenta la actualización del número de versión
A. 02	Detección de nuevas versiones y actualización	Debe ser capaz de actualizarse automáticamente si existe una versión posterior en el market privado
A. 03	Configuración automática primer uso	La primera vez que se ejecuta la aplicación, debe ajustar la configuración inicial correctamente
A. 04	Login (registro) de usuario en el sistema. Datos correctos. Se dispone de conexión	Envío de los datos de usuario al servidor. Almacenamiento de información de usuario en la base de datos local
A. 05	Login con datos erróneos	Detección de pares nombre/contraseña erróneos. Solicitar nuevo par nombre/contraseña
A. 06	Login sin disponer de conexión	Imposibilidad de entrar al sistema.
A. 07	Login de usuarios con distintas combinaciones de permisos (alta, iniciar/finalizar, validar)	El usuario puede realizar acciones acordes a sus permisos. Por ejemplo, un usuario con permisos de inciar/finalizar tareas y dar de alta tareas podrá realizar dichas acciones pero no validar.
A. 08	Autologin. Hay conexión y hay almacenados datos de usuario.	Registro transparente para el usuario. Pasamos directamente al menú principal.
A. 09	Autologin. No hay conexión. Hay datos de usuario almacenados.	Mensaje de aviso. Registro en el sistema utilizando la identidad y permisos del usuario guardado en la BD.
A. 10	Autologin.No hay almacenados datos de usuario.	El sistema muestra la pantalla de login.
A. 11	Cerrar sesión (logout).	Borrado de los datos de usuario de la BD. Posteriormente, al iniciar debe mostrar la pantalla de login (y no realizar un registro automático, ya que no tiene información almacenada)

Tabla 1. Plan de pruebas para la instalación y el módulo de registro

3.3.2. Lista de tareas

REF	Proceso	Resultado esperado
B. 01	Desde el menú principal, acceder a la lista de tareas	Se accede a la pantalla y se desencadenan las acciones producidas al pulsar el botón "refrescar tareas"
B. 02	Botón "refrescar tareas"	Se muestran las 50 primeras tareas almacenadas en nuestra base de datos. Para cada tarea, la información sobre los reportes ya redactados debe ser correcta. Aviso de tabla vacía si es necesario.
B. 03	Botón "actualizar tareas". Configuración: sector seleccionado y uno o varios tramos seleccionados	Se actualiza la base de datos local con la información recibida del servidor. Sólo recibimos las tareas correspondientes a nuestro sector y tramos seleccionados, además de que el servidor tampoco nos envía las que ya están validadas. Se desencadenan las acciones producidas al pulsar el botón "refrescar tareas"
B. 04	Botón "actualizar tareas". Configuración: sector seleccionado y ningún tramo seleccionado	Igual que el anterior, excepto que el servidor nos envía todas las tareas asociadas al sector seleccionado, sin filtrar por tramos.
B. 05	Botón "actualizar tareas". Configuración: sector no seleccionado.	El servidor no nos devuelve información. Por consiguiente, la tabla de la lista de tareas estará vacía.

Tabla 2. Plan de pruebas para la lista de tareas

3.3.3. Ficha de tarea

REF	Proceso	Resultado esperado
C. 01	Desde la lista de tareas, hacer click en una tarea. Hay conexión.	Accedemos a la ficha de dicha tarea. Se muestran todos los detalles correctamente.
C. 02	Desde la lista de tareas, hacer click en una tarea. No hay conexión.	Accedemos a la ficha de dicha tarea. Se muestran todos los detalles excepto el mapa de google maps, que no puede cargarse dado que requiere conexión.
C. 03	Tenemos permiso para redactar informes. Todavía no hemos redactado ninguno.	Aparece en la parte superior de la pantalla un botón para redactar el informe de iniciación.
C. 04	Tenemos permiso para redactar informes. Hemos redactado el de iniciación pero no el de finalización.	Aparecen en la parte superior de la pantalla dos botones, uno para visualizar el reporte de iniciación y otro para redactar el de finalización.
C. 05	El propio usuario ha redactado el informe de iniciación de una tarea. Pulsamos el botón "iniciar actuación" en esa tarea.	Se visualiza el informe redactado anteriormente. No se permite realizar ningún cambio ni guardarlo de nuevo.

C. 06	Otro usuario ha redactado el informe de iniciación de una tarea. Pulsamos el botón "iniciar actuación" en esa tarea.	El sistema muestra un mensaje notificando que dicho informe ya ha sido redactado por otro usuario.
C. 07	Tenemos permiso para redactar informes. Ambos han sido ya redactados.	Aparecen en la parte superior de la pantalla dos botones para visualizar los respectivos reportes.
C. 08	El propio usuario ha redactado el informe de finalización de una tarea. Pulsamos el botón "finalizar actuación" en esa tarea.	Se visualiza el informe redactado anteriormente. No se permite realizar ningún cambio ni guardarlo de nuevo.
C. 09	Otro usuario ha redactado el informe de finalización de una tarea. Pulsamos el botón "finalizar actuación" en esa tarea.	El sistema muestra un mensaje notificando que dicho informe ya ha sido redactado por otro usuario.
C. 10	Tenemos permiso para validar tareas. Los dos informes de actuación de una tarea (iniciar/finalizar) se han rellenado.	Aparece en la parte inferior de la pantalla un botón para validar la tarea.
C. 11	Se pulsa el botón "validar tareas". La tarea no estaba validada. Hay conexión.	Se guarda el formulario en la tabla de formularios de la BD. Después se ejecuta el **proceso de envío de formularios**. Consiste en rescatar de esta tabla todos los datos de los formularios pendientes. Todos aquellos que se guardan correctamente en el servidor actualizan su estado, pasando de "datos pendientes" a "datos enviados", de manera que ya no serán enviados de nuevo posteriormente. Se realiza el mismo proceso para las imágenes pendientes de ser enviadas. Ver diagrama del **proceso de envío de formularios**.
C. 11	Se pulsa el botón "validar tareas". La tarea no estaba validada. No hay conexión.	Se guarda el formulario en la tabla de formularios de la BD, quedando como pendiente.
C. 12	Se pulsa el botón "validar tareas". La tarea ya estaba validada.	Se muestra un mensaje notificando que la tarea ya estaba validada.

Tabla 3. Plan de pruebas para la ficha de tarea

3.3.4. Informes de iniciación/finalización

REF	Proceso	Resultado esperado
D. 01	Viniendo de la pantalla de la ficha de tarea, entramos en la pantalla de redactar un informe. Hay conexión.	Se muestra correctamente el ID de la tarea. Además, aparecen la hora y fecha actuales y la geolocalización (longitud, latitud). El mapa de google maps se encuentra centrado en las coordenadas de nuestra localización.
D. 02	Viniendo de la pantalla de la ficha de tarea, entramos en la pantalla de redactar un informe. No hay conexión.	Se muestra correctamente el ID de la tarea. Además, aparecen la hora y fecha actuales. No aparece la geolocalización ni el mapa de google.

D. 03	Pulsar el botón "capturar nueva imagen".	Se abre la aplicación nativa de la cámara. Tomamos una fotografía y al cerrar la aplicación de cámara volvemos a nuestra aplicación. La imagen ahora se muestra en grande. Las imágenes adjuntadas previamente se muestran como miniatura.
D. 04	Pulsar el botón "cargar imagen del álbum".	Se abre la aplicación nativa de la galería de imágenes. Escogemos una imagen y al cerrar la aplicación de galería volvemos a nuestra aplicación. La imagen ahora se muestra en grande. Las imágenes adjuntadas previamente se muestran como miniatura.
D. 05	Pulsar el botón "quitar imagen".	La imagen mostrada en primer plano se desancla de nuestro formulario (no se elimina de memoria pero ya no está adjunta al formulario). No será enviada al servidor.
D. 06	Hacer click sobre alguna de las miniaturas de las imágenes que tengamos adjuntas.	Dicha imagen pasa a mostrarse en primer plano, y la que estaba antes en grande se muestra sólo como miniatura.
D. 07	Añadir más imágenes cuando tenemos ya tres adjuntas.	Se muestra un mensaje informando de la imposibilidad de adjuntar más de 3 imágenes por formulario.
D. 08	Pulsar el botón "enviar"	Se guarda el formulario en la tabla de formularios de la BD. Después se ejecuta el **proceso de envío de formularios**. Ver diagrama del **proceso de envío de formularios**.

Tabla 4. Plan de pruebas para la redacción de informes

3.3.5. Alta de tareas

REF	Proceso	Resultado esperado
E. 01	Entramos en la pantalla de altas de nuevas tareas. Hay conexión.	Se muestran todas las opciones y menús desplegables. Además, aparecen la hora y fecha actuales y la geolocalización (longitud, latitud). El mapa de google maps se encuentra centrado en las coordenadas de nuestra localización.
E. 02	Entramos en la pantalla de altas de nuevas tareas. No hay conexión.	Se muestran todas las opciones y menús desplegables. Además, aparecen la hora y fecha actuales. No aparece la geolocalización ni el mapa de google.
E. 03	Seleccionamos un sector en el menú desplegable correspondiente.	Se activa el menú desplegable "contrato". El campo provincia muestra el dato correspondiente. Los campos "protocolo general del sector" y "protocolo específico" muestran la información correspondiente.

E. 04	Deseleccionamos el sector que tuviéramos en el menú desplegable (elegimos la opción "seleccione una opción").	El menú desplegable "contrato" pasa a no tener ninguna opción seleccionada y se desactiva. Los campos "protocolo general del sector" y "protocolo específico" se vacían (muestran "No hay protocolo designado").
E. 05	Seleccionamos una clase en el menú desplegable correspondiente.	Se activa el menú desplegable "tipo".
E. 06	Seleccionamos un tipo en el menú desplegable correspondiente.	Se activa el menú desplegable "subtipo". El campo "actuación fundamental" muestra la información correspondiente.
E. 07	Seleccionamos un subtipo en el menú desplegable correspondiente	Los campos "plazo para finalizar actuación" y "fecha y hora para finalizar actuación" muestran la información correspondiente.
E. 08	Deseleccionamos el subtipo que tuviéramos en el menú desplegable (elegimos la opción "seleccione una opción").	Los campos "plazo para finalizar actuación" y "fecha y hora para finalizar actuación" se vacían.
E. 09	Deseleccionamos el tipo que tuviéramos en el menú desplegable (elegimos la opción "seleccione una opción").	El menú desplegable "subtipo" pasa a no tener ninguna opción seleccionada y se desactiva. El campo "actuación fundamental" se vacía.
E. 10	Deseleccionamos la clase que tuviéramos en el menú desplegable (elegimos la opción "seleccione una opción").	El menú desplegable "tipo" pasa a no tener ninguna opción seleccionada y se desactiva.
E. 11	Menú desplegable "fuente de información"	Se muestran sólo las opciones acordes al rol del usuario registrado.
E. 12	Pulsar el botón "capturar nueva imagen".	Se abre la aplicación nativa de la cámara. Tomamos una fotografía y al cerrar la aplicación de cámara volvemos a nuestra aplicación. La imagen ahora se muestra en grande. Las imágenes adjuntadas previamente se muestran como miniatura.
E. 13	Pulsar el botón "cargar imagen del álbum".	Se abre la aplicación nativa de la galería de imágenes. Escogemos una imagen y al cerrar la aplicación de galería volvemos a nuestra aplicación. La imagen ahora se muestra en grande. Las imágenes adjuntadas previamente se muestran como miniatura.
E. 14	Pulsar el botón "quitar imagen".	La imagen mostrada en primer plano se desancla de nuestro formulario (no se elimina de memoria pero ya no está adjunta al formulario). No será enviada al servidor.
E. 15	Hacer click sobre alguna de las miniaturas de las imágenes que tengamos adjuntas.	Dicha imagen pasa a mostrarse en primer plano, y la que estaba antes en grande se muestra sólo como miniatura.
E. 16	Añadir más imágenes cuando tenemos ya tres adjuntas.	Se muestra un mensaje informando de la imposibilidad de adjuntar más de 3 imágenes por formulario.
E. 17	Pulsar el botón "enviar"	Se guarda el formulario en la tabla de formularios de la BD. Después se ejecuta el **proceso de envío de formularios**. Ver diagrama del **proceso de envío de formularios**.

Tabla 5. Plan de pruebas para la realización de informes

3.3.6. Opciones

REF	Proceso	Resultado esperado
F. 01	Entramos en la pantalla de opciones.	Se muestran todos los menús desplegables y campos correctamente.
F. 02	Hacer click en el botón ″Cerrar sesión″ (logout).	Se muestra un mensaje preguntando si se está seguro de querer cerrar sesión. Borrado de los datos de usuario de la BD. Posteriormente, al iniciar debe mostrar la pantalla de login (y no realizar un registro automático, ya que no tiene información almacenada)
F. 03	Hacer click en el botón ″Borrar reportes y tareas″.	Deben de eliminarse todos aquellos formularios que ya hayan sido enviados con éxito al servidor. Además, también debe vaciarse la lista de tareas.
F. 04	Hacer click en el botón ″Borrado maestro″.	Realiza un reseteo de la aplicación. Borra todos los contenidos de la base de datos y restaura la configuración de fábrica.
F. 05	Hacer click en el botón ″Cargar base de datos – alta″.	Pide al servidor la información relativa a las altas de nuevas tareas (contratos, clases, tipos, carreteras, protocolos… etc) y las almacena en las correspondientes tablas de la BD. Al dar de alta una nueva tarea, las opciones se muestran actualizadas.
F. 06	Hacer click en el botón ″Enviar formularios pendientes″.	Se activa el **proceso de envío de formularios**. No se muestra ningún mensaje hasta el final, cuando se nos indicará si se han producido errores en alguno de los formularios pendientes.
F. 07	Seleccionar un sector en el menú desplegable correspondiente.	Cuando actualicemos la lista de tareas, el servidor nos enviará las tareas asociadas a dicho sector.
F. 08	Seleccionar uno o varios tramos en el menú desplegable correspondiente.	Cuando actualicemos la lista de tareas, el servidor nos enviará las tareas asociadas a los tramos seleccionados del sector seleccionado.
F. 09	Modo de transmisión de imágenes ON.	Las imágenes adjuntas a los formularios sólo tratarán de ser enviadas al servidor si existe una conexión rápida (wifi)
F. 10	Activar actualización automática de tareas ON.	El sistema muestra la pantalla de login.
F. 11	Seleccionar un sector en el menú desplegable correspondiente.	Cuando actualicemos la lista de tareas, el servidor nos enviará las tareas asociadas a dicho sector.

F. 12	Seleccionar uno o varios tramos en el menú desplegable correspondiente.	Cuando actualicemos la lista de tareas, el servidor nos enviará las tareas asociadas a los tramos seleccionados del sector seleccionado.
F. 13	Modo de transmisión de imágenes ON.	Las imágenes adjuntas a los formularios sólo tratarán de ser enviadas al servidor si existe una conexión rápida (wifi)
F. 14	Actualización automática de tareas ON.	Se realizará una actuación automática de la lista de tareas cada X minutos, siendo X la frecuencia seleccionada en la opción correspondiente. Ver diagrama del **proceso de actualización automática de la lista de tareas**.
F. 15	Envío automático de formularios ON.	Se realizará un envío automático de los formularios pendientes cada X minutos, siendo X la frecuencia seleccionada en la opción correspondiente. Ver diagrama del **proceso de envío de formularios**.
F. 16	Seleccionar uno o varios tramos en el menú desplegable correspondiente.	Cuando actualicemos la lista de tareas, el servidor nos enviará las tareas asociadas a los tramos seleccionados del sector seleccionado.
F. 17	Modo de transmisión de imágenes ON.	Las imágenes adjuntas a los formularios sólo tratarán de ser enviadas al servidor si existe una conexión rápida (wifi)
F. 18	Resolución de imágenes capturadas	Según la opción elegida, las imágenes capturadas por la cámara tendrán una cierta resolución.

Tabla 6. Plan de pruebas para las opciones de configuración

3.4. Elementos de apoyo a la aplicación

Aunque el elemento principal del proyecto es la aplicación móvil, hay elementos adicionales que la apoyan y junto con ésta, conforman un sistema integrado.

3.4.1. Servidor

Ya hemos comentado anteriormente que nuestro sistema se basa en una arquitectura cliente-servidor. Aunque hasta ahora nos hayamos centrado en la aplicación (lado cliente) por ser el elemento central del sistema, merece la pena también comentar algunos aspectos del servidor.

El servidor está basado en la tecnología Apache y programado en el lenguaje PHP. Así mismo, gestiona una base de datos perteneciente al sistema MySQL.

Cabe decir que aunque la programación final del servidor no ha sido realizada por el autor de este proyecto, sí lo fue la programación del servidor de pruebas durante el desarrollo de la aplicación. Esto tiene que ver con la metodología de trabajo en la empresa Iternova S.L., según la cual todos los módulos del servidor deben estar integrados en el sistema SGWC para lo cual se sigue un formato, metodología y documentación concretos.

Sin embargo, como ya se ha comentado la programación funcional del servidor sí que fue realizada por el autor del proyecto, y todas las decisiones acerca de cómo debía interaccionar con la aplicación fueron tomadas por él.

Con esto queda patente que este proyecto engloba también la parte servidor, lo que le da una dimensión más amplia que la de la programación de una aplicación.

3.4.2. Control de versiones y actualizaciones

En el momento en el que se tuvo una versión apta para comenzar a realizar las pruebas en el sistema fue necesario implementar un control de versiones que nos permitiera trazar una evolución de la aplicación y controlar qué errores habían sido corregidos.

Para agilizar el proceso de actualización a última versión, debido a que se va a instalar en muchos dispositivos móviles (por ejemplo, en la carretera de Sinaloa va a haber unos 50 residentes con tablet), se utilizó un servicio on-line de gestión de versiones muy sencillo de utilizar (un market privado).

Se ha integrado una extensión en la aplicación para permitir la auto-actualización mediante market privado, así como para facilitar la instalación, lo que permitirá en el futuro añadir nuevas funcionalidades de forma rápida.

El desarrollador puede subir a la web la última versión de su aplicación, y ésta misma aparecerá junto a un código BIDI[13] el cual permite que un usuario que disponga de un lector de estos códigos pueda descargar e instalar la aplicación.

Además, una vez descargada la primera vez, la aplicación comprueba con cierta frecuencia si existen nuevas actualizaciones a la misma en dicho servicio, de tal manera que no es necesario comprobar manualmente si existen actualizaciones sino que esto se realiza de manera automática.

Se ha usado un market privado ya que el sistema de gestión web es propio de una empresa en particular, y solo usuarios registrados en dicha plataforma van a tener acceso al sistema.

3.4.3. Ficheros de configuración y librerías

Además de la información dinámica que se carga automáticamente desde el servidor web (ya sea al registrarse por primera vez como de forma periódica), se han incluido los parámetros de configuración de la aplicación en ficheros de configuración. Esto facilita las tareas de mantenimiento y reutilización de la aplicación, ya que en estos ficheros se puede almacenar, por ejemplo, la URI del servidor web, o parámetros de personalización.

También se ha realizado un fichero a modo de librería, que contiene las funciones más comunes desarrolladas específicamente para la aplicación pero que podrían ser reutilizadas en sistemas similares.

13 Los códigos bidimensionales (abreviadamente códigos BiDi), son un sistema gráfico creado para almacenar información. Constituyen la evolución natural de los conocidos códigos de barras, que sólo pueden guardar 20 caracteres, pasando de emplear sólo una dimensión a emplear dos dimensiones: una matriz de pequeños cuadrados que pueden ser blancos o negros. De este modo, se consigue una capacidad de almacenamiento 100 veces mayor y una mejor protección frente a errores de lectura.

3.5. Problemas planteados durante el desarrollo del proyecto

Han sido numerosos los problemas que han aparecido durante el transcurso de este proyecto fin de carrera. Aunque los problemas han sido de diversa naturaleza, casi todos han tenido un denominador común: la inexperiencia en los lenguajes de programación utilizados. Y como también es lógico, estos problemas se hacían menos graves al final del desarrollo, cuando el manejo de las tecnologías había alcanzado un nivel aceptable.

Las principales dificultades encontradas son:

- Manejo de múltiples lenguajes de programación a la vez, cuyo desconocimiento era prácticamente total al comienzo del proyecto. Aunque ya se ha comentado que ésta era la raíz de muchos otros problemas posteriores, es también una dificultad en sí misma en el sentido de que la comprensión de todos los conceptos y metodologías fue un importante escollo que resolver y fue necesaria una gran cantidad de tiempo para progresar.
- Problemas de incompatibilidad y ”bugs”[14]. En ocasiones se han enfrentado problemas que, en principio, parecían no tener solución. Al cabo del tiempo, tras muchas indagaciones (en ocasiones incluso de varios días) se concluyó que eran errores de incompatibilidad entre las tecnologías y API's[15] utilizadas. De esta manera se trató de realizar la misma tarea que se pretendía de una manera diferente o actualizar los códigos a la última versión. Ejemplos de esto fue la imposibilidad de añadir parámetros al tratar de cambiar a otra URL en sistemas Android. También hubo dificultades a la hora de trabajar en el servidor de pruebas dado que existían problemas con los certificados SSL del mismo.
- Problemas en los dispositivos de pruebas. En uno de los dispositivos en el cual se realizaron pruebas de la aplicación se presentaban varios problemas de funcionamiento, que no tenían su raíz en la aplicación desarrollada sino en algún problema de dispositivo. Éste fue el caso del envío de imágenes al servidor, lo cual no era posible desde uno de los tablets que se usaron para las pruebas, lo cual causó un considerable problema al pensar que se trataba de un error en el código cuando no era así realmente.
- Errores de concepto en el desarrollo de la aplicación. Éste fue un problema derivado de la falta de perspectiva de lo que tenía que ser la aplicación, por lo que el sistema se comportaba de maneras no esperadas. Esta dificultad se mostró a la hora de probar la aplicación (hubo que hacer sustanciales cambios) y al entregar las primeras versiones a nuestro cliente. Tras una serie de correcciones por parte de éste, el autor llegó a la conclusión de que tomaría menos tiempo realizar un completo rediseño de la aplicación, aprovechando los códigos que valiesen la pena, los conocimientos adquiridos y, sobre todo, la visión global de lo que tenía que ser la aplicación. Esta decisión se reveló un éxito ya que, en menos de dos semanas la aplicación estaba lista de nuevo y esta vez, tras unas últimas correcciones, funcionaba correctamente.

14 Error del programa

15 Application Programming Interface, interfaz de programación de aplicaciones. es el conjunto de métodos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Capítulo 4. CONCLUSIONES

En este capítulo final, se realizará una valoración final del proyecto, así como las posibles líneas futuras de desarrollo a partir del trabajo realizado.

4.1. Resultados obtenidos

El principal resultado obtenido ha sido la aplicación móvil ”agenda de vialidad” para la gestión de explotación de carreteras enmarcada en el sistema SGWC de Iternova. Aunque el sistema ha sufrido modificaciones menores, el objetivo principal se ha cumplido, ya que nuestros clientes han incorporado ya el sistema en su trabajo diario y, tras una fase inicial de depuración de errores, la aplicación se encuentra funcionando en el entorno para el cual se desarrolló.

Hasta el momento, nuestros clientes nos han transmitido su satisfacción por el funcionamiento de la aplicación, y se espera dotar al sistema de nuevas capacidades en un futuro próximo.

4.2. Líneas futuras de desarrollo

La aplicación móvil “agenda de vialidad” ha sido desarrollada de manera modular, de forma que la ampliación y mantenimiento del sistema no sea una tarea costosa. Esto se ha tenido en cuenta debido a que se preveían nuevas necesidades por parte de nuestros clientes y la propia capacidad de mejora que presenta la aplicación.

Entre otras, las funcionalidades que podría incluir una revisión del sistema son:

- Sistema de búsqueda específica de tareas, incluyendo filtros personalizables por el usuario. De esta manera, se amplía la agilidad en el uso al proporcionar una capacidad de búsqueda más específica.
- Obtención de los datos de hora y fecha de la información GPS. Actualmente estos datos se toman directamente de la hora marcada por el dispositivo, con lo cual nada impide al usuario modificar la hora o la fecha del dispositivo, falseando así la información incluida en los formularios. Esto es importante si consideramos que los operarios tienen unos plazos máximos en la resolución de las tareas. Utilizando la información de la portadora GPS, los usuarios no pueden modificar los datos, aunque esto plantea la problemática de la situación en la que no se dispone de cobertura GPS.
- Menú de la aplicación. Una de las carencias de la aplicación es un menú desplegable que permitiera, por ejemplo, el *logoff* (de-registro) del usuario de la aplicación, la salida de la misma, y otras opciones de menú contextual. Esto mejoraría la interacción del usuario y añadiría rapidez en su manejo.
- Otras mejoras. También se ha pensado en realizar otras pequeñas mejoras como por ejemplo, cambios menores en el diseño y esquema de colores de la aplicación (código de colores en la lista que identifiquen las tareas más prioritarias o que distingan aquellas cuyos informes han sido redactados). También se ha pensado en la inclusión de imágenes en la ficha de descripción de las tareas (nótese que al dar de alta una tarea podemos asociar imágenes a ella pero éstas no se muestran posteriormente cuando tenemos dicha tarea en nuestra lista).

4.2.1. Desarrollo de la aplicación móvil ”agenda de inspección”

Algunas de las mejoras comentadas anteriormente han sido incluidas en otra aplicación desarrollada tras el proyecto fin de carrera, enmarcada en prácticas universitarias en la misma empresa. Esta aplicación es similar a la realizada en el proyecto fin de carrera, aunque está enfocada a la revisión de los elementos de inventariado (señales de tráfico, marcas viales, estructuras) en la carretera. Este sistema cuenta con algunas características adicionales a la ”agenda de vialidad”.

- La lista de elementos incluye una barra de búsqueda en la que el usuario puede incluir el texto a buscar en los elementos.
- Además, en la lista de tareas se ha incluido un filtro de área, mediante el cual podemos obtener una lista que contenga solamente las entradas referidas a elementos situados en un área de X kilómetros alrededor de nuestra posición (tomada mediante geolocalización), donde X es un parámetro configurable por el usuario.

4.3. Valoración personal

Considero que el proyecto fin de carrera realizado ha sido una gran experiencia a todos los niveles, tanto personal como profesional.

En primer lugar, creo que los conocimientos adquiridos me serán útiles más adelante en futuros trabajos, ya que son tecnologías actuales y demandadas hoy en día en el mercado laboral.

Además, he tenido la oportunidad de comprobar la realidad de un proyecto software en el ámbito empresarial, lo que considero muy positivo ya que me da una visión global de la realidad laboral del sector de las telecomunicaciones/informática.

Así mismo, el haberme superpuesto al problema inicial de la falta total de conocimientos, y haber sido capaz de llevar a cabo algo que al principio me parecía inabordable, me deja una gran sensación de satisfacción personal, además de todos los problemas resueltos durante el desarrollo del sistema, que no fueron pocos ni sencillos.

Finalmente, ver que el trabajo realizado está siendo utilizado en este momento con buenos resultados es muy satisfactorio al saber que resulta útil para algo que mejora la calidad de vida de la gente, como es la gestión de incidencias en la carretera.

Bibliografía

jQuery

[1] http://docs.jquery.com/Main_Page

[2] http://www.javascriptya.com.ar/jquery/

jQuery Mobile

[3] http://jquerymobile.com/

[4] http://www.ibm.com/developerworks/xml/tutorials/x-jquerymobilejsontut/x-jquerymobilejsontut-pdf.pdf

PhoneGap

[5] http://docs.phonegap.com/en/2.0.0/index.html

Android

[6] http://www.androidhive.info/

[7] http://code.google.com/p/android/issues/list

HTML

[8] http://roble.pntic.mec.es/apuente/html/paginas/resumen.htm

[9] http://html.conclase.net/tutorial/html/

JavaScript

[10] http://www.librosweb.es/javascript/

[11] http://www.javascriptya.com.ar/

[12] http://www.w3schools.com/js/default.asp

AJAX

[13] http://www.librosweb.es/ajax/

[14] http://www.ajaxya.com.ar/

SQL

[15] http://www.w3schools.com/sql/default.asp

CSS

[16] http://www.librosweb.es/css/

PHP

[17] http://www.uca.es/softwarelibre/publicaciones/apuntes_php

[18] http://www.php.net/

[19] http://www.calitae.com/manuales/manual-php.pdf

jQuery Mobile Plugin for Google Map

[20] http://www.mobiledevelopersolutions.com/home/download/demos-and-tutorials

[21] http://www.mobiledevelopersolutions.com/home/start/twominutetutorials/tmt0

[22] http://www.mobiledevelopersolutions.com/home/start/twominutetutorials/tmt4part1

Foros de consulta

[23] http://stackoverflow.com/

Anexo A. Manual de usuario de la agenda de vialidad (versión móvil)

En este primer anexo se presenta el manual de usuario elaborado con el fin de detallar el funcionamiento y las posibilidades de la aplicación pensando en el usuario final.

A.1. Introducción

La aplicación ”Agenda de vialidad” ha sido desarrollada para permitir su correcta visualización en cualquier tipo de dispositivo (ordenador, tablet o móvil), ya que cuenta con una versión móvil con funcionalidades específicas para permitir la introducción de las actividades realizadas por los operarios en el mismo lugar en el que realizan una acción. Por ejemplo, pueden dar de alta nuevas tareas en el sistema, incorporando automáticamente toda la información asociada como fotografías, localización u otros datos.

La versión optimizada para dispositivos móviles (tablets y smartphones) será el objeto de este manual de usuario, y cuenta con las siguientes características:

- Compatible para diferentes modelos de dispositivo.
- Optimizada para minimizar los tiempos de carga, y agilizar el el uso de la aplicación por parte del usuario en entornos remotos.
- Se tienen en cuenta los problemas de límites de ancho de banda y cobertura de las redes móviles, disponiendo de los mecanismos oportunos para agilizar la velocidad de la aplicación y evitar la pérdida de información por falta de cobertura.
  Respecto a la cobertura, la aplicación dispone de todos los elementos necesarios para permitir trabajar con ella aún cuando ésta no exista (de forma autónoma), volviendo a sincronizar la información con los servidores cuando la cobertura vuelva a estar disponible, de forma que se garantice la continuidad del funcionamiento sin perjuicio alguno para el usuario.

La aplicación permite a los operarios la actualización instantánea de las tareas en el mismo momento en que proceden a su actuación, al mismo tiempo que se controla la posición exacta en la que se encuentra, posibilitando el seguimiento de la resolución de las tareas por parte de la empresa para mantener el control sobre las actuaciones realizadas.

El módulo de agenda móvil es similar al que se puede observar en su versión extendida (para plataforma web), aunque está orientado a la introducción de información (y no a la consulta de informes).

A.2. Opciones del usuario

A.2.1. Acceso a la aplicación (login)

Al ejecutar la aplicación en el dispositivo móvil se muestra la pantalla de entrada en la que se deben ingresar los datos del usuario registrado (login/paswword).

Figura 17. Pantalla inicial

Figura 18. Formulario de registro de usuario

Si los datos son correctos, desde ese momento el usuario (operario) podrá acceder a las opciones que, según los permisos asignados a ese usuario, tenga disponibles:

- Ver los registros de tareas asignadas
- Registrar una actuación
- Iniciar / finalizar una actuación
- Validar tareas

Además, se guardan los datos del usuario en la base de datos del móvil, de manera que cuando cierre o minimice la aplicación no se le vuelva a solicitar los datos de usuario (sólo se le vuelve a solicitar cuando el usuario cierra la sesión desde el menú de configuración).

A.2.2. Inicialización y configuración de la aplicación

La primera vez que se ejecuta la aplicación el usuario debe hacerlo asegurándose de que dispone de conexión a Internet. Al registrarnos por primera vez aparecen los siguientes mensajes:

Figura 19. Configuración automática inicial (1)

Figura 20. Configuración automática inicial (2)

Simplemente dejamos que la aplicación haga su trabajo, y al finalizar con éxito observaremos el mensaje:

Figura 21. Configuración automática inicial (3)

Figura 22. Menú principal

Para poder comenzar a trabajar, lo primero que debemos hacer es establecer un sector y tramo. Para ello deberá acceder al apartado de "Opciones" que se muestra en el menú principal de la aplicación y seleccionar un sector del desplegable correspondiente, ya que solo recibirá tareas del servidor del sector que le corresponda. También puede seleccionar un tramo/segmento de carretera, para recibir incidencias sólo de este tramo.

Si por alguna razón la configuración automática inicial fallara, deberemos hacer click sobre "Opciones" del menú principal y seguir los siguientes pasos por orden:

Desde esta pantalla deberá llevar a cabo las siguientes acciones por orden:

1. Borrado maestro: necesario para realizar la primera instalación.
2. Cargar base de datos - alta: en este momento, la aplicación se conecta con el servidor y se descargará todas las configuraciones "dinámicas" desde el servidor, por ejemplo: clases, tipos, subtipos de tareas de la agenda de vialidad, las carreteras configuradas en el sistema, los sectores, etcétera...
3. Seleccionar sector: el usuario deberá seleccionar un sector del desplegable correspondiente, ya que solo recibirá tareas del servidor del sector que le corresponda. También puede seleccionar un tramo / segmento de carretera, para recibir incidencias solo de este tramo.

Con estos pasos ya está preparada la aplicación para ser utilizada. Sin embargo hay otras opciones de configuración que explicaremos en el siguiente apartado. Las siguientes imágenes corresponden al submenú "Opciones":

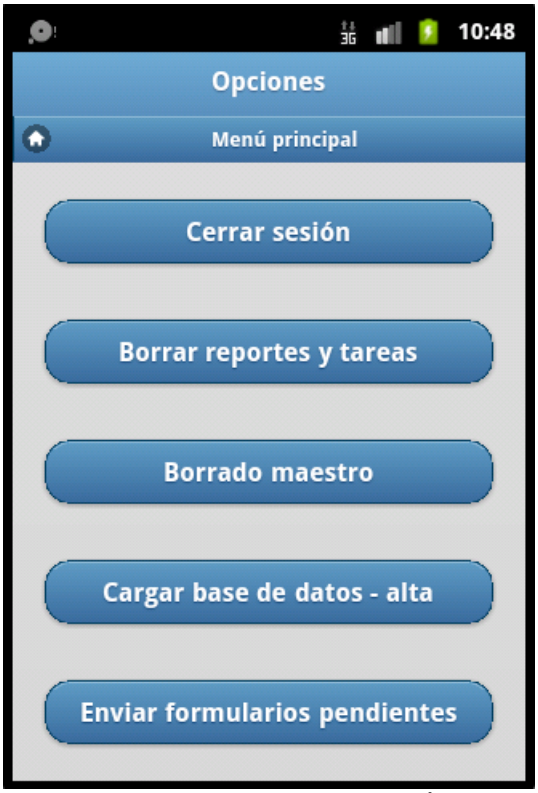

Figura 24. Opciones de configuración (1)

Figura 23. Opciones de configuración (2)

Otras opciones de configuración

Desde la pantalla de "Opciones" vamos a poder configurar algunas características de funcionamiento de la aplicación (que podremos variar siempre que deseemos).

Modo de transmisión de imágenes y tamaño

Podemos elegir cómo se va a comportar la aplicación en caso de que tenga que enviar imágenes al servidor:

A) Usar comunicación 3G para el envío de imágenes: cuando una imagen ha de ser enviada, basta con que exista conexión 3G para ser enviada en ese momento. No es muy recomendable ya que las conexiones 3G suelen tener restricciones de ancho de banda y datos totales enviados.

B) No usar comunicación 3G: la transmisión de imágenes entre el dispositivo y el servidor sólo se realizará cuando exista conexión Wi-Fi, (opción por defecto).

También podemos configurar la resolución con la que deseamos enviar las imágenes al servidor. Con la ayuda del desplegable seleccionamos la resolución.

Actualización de la lista de tareas

Se puede configurar si la lista de tareas se actualiza automáticamente o no y, en caso de que lo haga, la frecuencia con la que lo hará.

Borrar reportes y tareas

En la base de datos del dispositivo móvil se queda almacenada la información que por falta de conexión con el servidor no ha podido ser todavía enviada, pero también aquella información que ya ha sido enviada y confirmada.

Como esta información ya enviada correctamente ocupa espacio en la base de datos del dispositivo móvil, tenemos la opción de borrarla de la base de datos del dispositivo.

Para ello, en la sección de "Opciones" deberemos pulsar el botón "Borrar reportes y tareas". Debe quedar claro que esta operación sólo afecta a la información que ha llegado con éxito al servidor, por lo que no hay peligro de perder información.

Borrado maestro

Esta opción provocará una restauración completa de la configuración de la aplicación, volviendo a los valores de fábrica. No debe utilizarse a no ser que la aplicación muestre comportamientos extraños que no puedan solucionarse de otro modo.

Carga de base de datos – alta

Si en el sistema se introducen cambios en las bases de datos que describen las tareas (inclusión de nuevos tipo de incidencia, nuevos sectores o cambios similares), será necesario hacer click en este botón para poder dar de alta incidencias que contengan los nuevos cambios.

Sector y tramo

Éstos son dos menús desplegables que permiten seleccionar el sector y el/los tramo(s) en los que va a trabajar el usuario. Estos dos parámetros condicionan la lista de tareas que recibirá desde el servidor, que es específica para cada sector y tramo(s). Es obligatorio seleccionar un sector para poder recibir la lista de tareas. Sin embargo, si no se selecciona ningún tramo se recibirá la lista de tareas asociada a todos los tramos de dicho sector.

Envío automático de información pendiente

De la misma manera que podemos configurar la actualización automática de la lista de tareas, podemos configurar el envío automático de la información pendiente (aquella que permanece almacenada sin que anteriormente haya podido ser transferida con éxito). Tenemos una opción para activar el envío automático y otra para configurar la frecuencia de envío.

Envío de formularios pendientes

Además del envío automático que podemos configurar según el punto anterior, existe la posibilidad de enviar toda la información pendiente al mismo tiempo de forma manual. Para ello, solamente tendremos que hacer click en el botón ”Enviar formularios pendientes”.

Si hay conexión a Internet en ese instante, toda la información pendiente será transferida al servidor. De lo contrario, seguirá almacenada esperando el momento oportuno.

A.2.3. Acciones con las tareas

Desde el menú inicial, accedemos al botón “Lista de tareas”.

Al pulsar sobre el botón, se muestra la pantalla en la que se pueden observar, en caso de existir, las tareas asignadas al operario que todavía no han sido validadas:

Ver detalle de tarea

En el listado de la pantalla anterior, podemos pulsar sobre la tarea deseada y accedemos a la pantalla con los detalles de la tarea a realizar y desde la cual podremos llevar a cabo otras acciones.

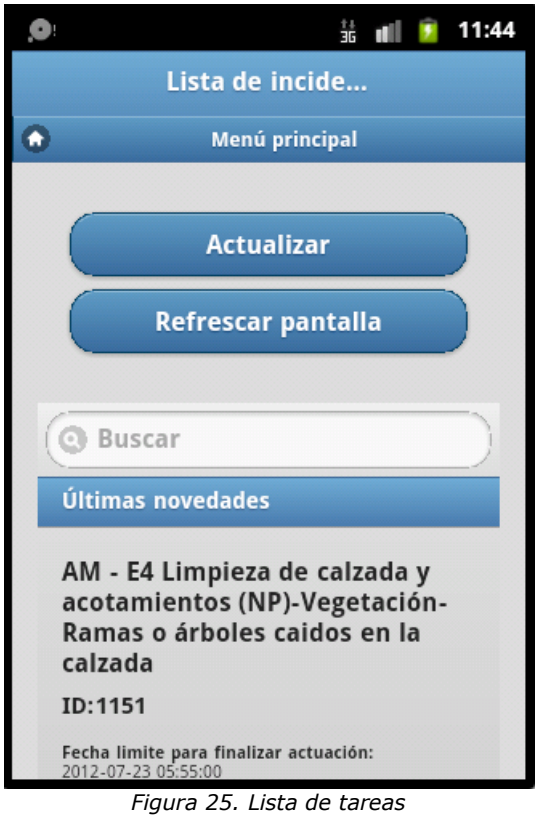

Figura 25. Lista de tareas

Figura 26. Ficha de tarea (1)

Figura 27. Ficha de tarea (2)

Iniciar actuación

Pulsando en el botón "Iniciar actuación" de la pantalla anterior, pasamos a la pantalla en la que el operario va a rellenar la información de la tarea (redactar el informe). Es importante recordar que tanto para iniciar una actuación como para finalizarla (proceso descrito a continuación) es necesario que el usuario disponga de los permisos correspondientes.

Figura 28. Informe de actuación (1)

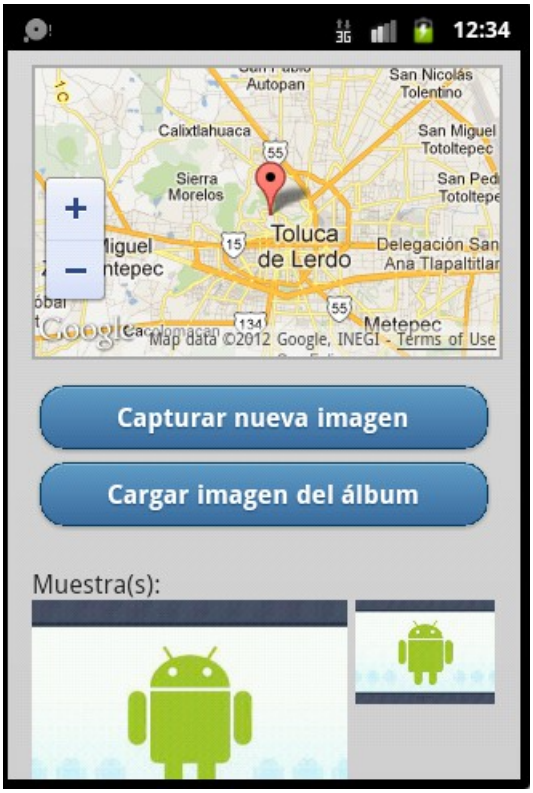

Figura 29. Informe de actuación (2)

Figura 30. Informe de actuación (3)

El operario podrá capturar una imagen en ese momento de la tarea que está realizando o añadir una imagen tomada previamente. A cada tarea se le pueden asociar hasta un máximo de 3 imágenes.

Los datos de la fecha, hora y geolocalización serán tomados automáticamente por el dispositivo. Pulsando el botón ”Enviar”, todos los informes de actuación (iniciación y finalización) pendientes, incluyendo el que acabamos de redactar, serán enviados al servidor si se dispone de conexión. Si no, el que acabamos de rellenar quedará almacenado a la espera.

Una vez realizada esta acción, haya o no conexión con el servidor, si volvemos al detalle de la tarea vemos que el marcador de reporte inicial aparece como completado.

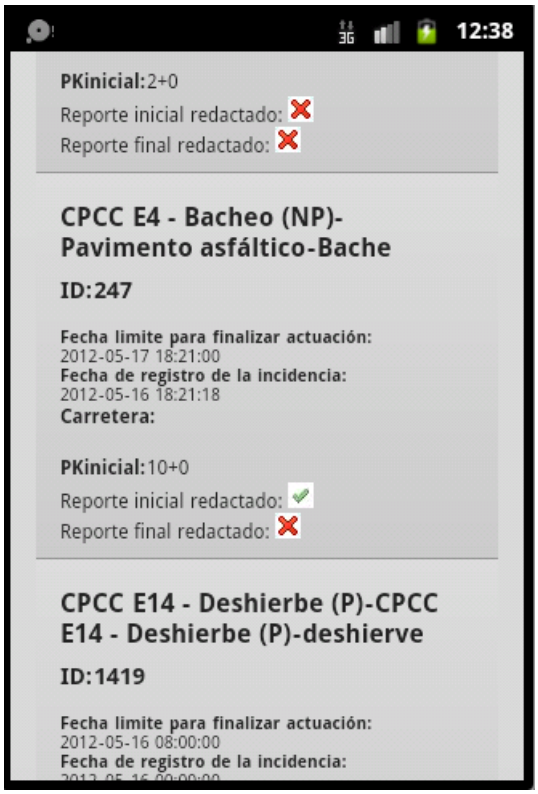

Figura 31. Lista de tareas. Informe inicial redactado

Finalizar actuación

Al igual que para iniciar la actuación, existe otro botón ”Finalizar actuación” para realizar la tarea correspondiente. Este botón sólo se mostrará si el reporte inicial ya ha sido redactado.

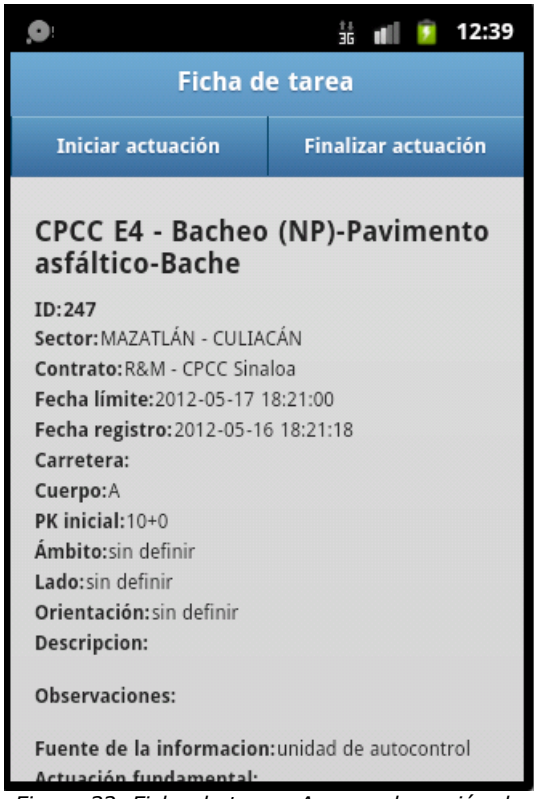

Figura 32. Ficha de tarea. Aparece la opción de finalizar

Figura 33. Informe de actuación final

Pulsándolo se muestran las mismas pantallas que se muestran en “Iniciar la actuación” para redactar (imágenes incluidas) el reporte final, aunque en este caso tendremos un campo ”observaciones” para describir el trabajo realizado.

Una vez redactado y pulsado el botón “Enviar”, se sucederá el mismo procedimiento que al iniciar la tarea. Volviendo al detalle de la tarea, se mostrará que los reportes de inicio y fin están redactados.

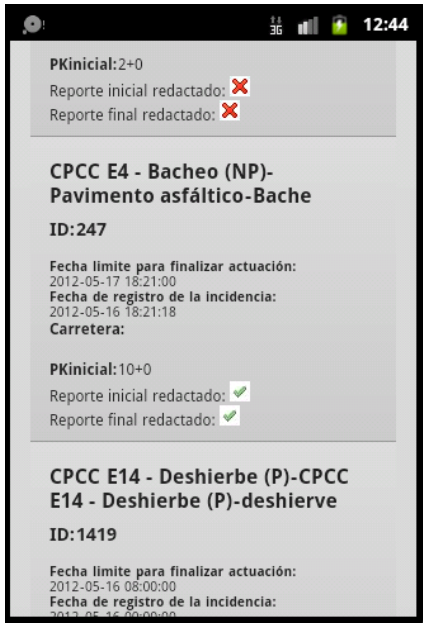

Figura 34. Lista de tareas. Ambos informes redactados

Validar una tarea

Si el usuario tiene permisos para ello, también podrá realizar la validación de una tarea. Esto supone dar el visto bueno a los trabajos realizados y hacerlo así constar en el sistema.

Para validar una tarea, deberemos acceder a la pantalla de "Ver detalle de tarea". Bajo la información de la tarea se muestra, en caso de haber rellenado los informes de iniciación y finalización, el botón "Validar". Pulsamos el botón y si disponemos de conexión, el servidor registrará esta validación. Si no, la aplicación tratará de reenviar la petición en otro momento como en el caso de los otros informes.

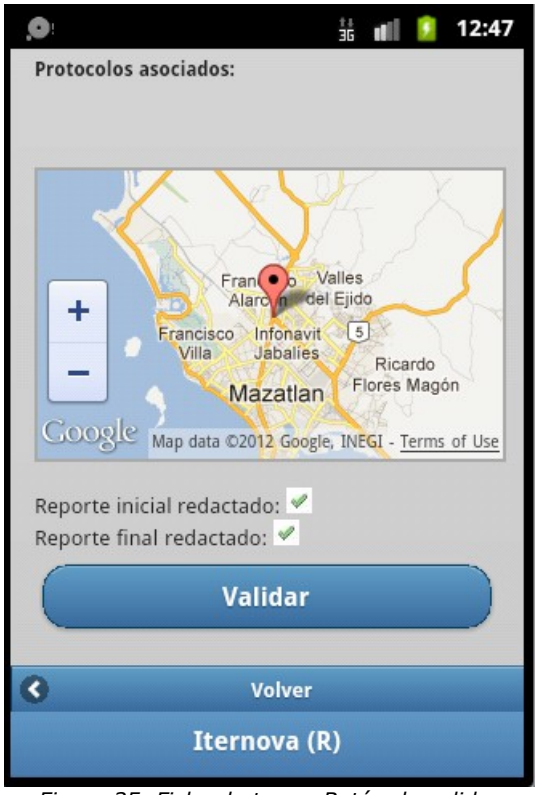

Figura 35. Ficha de tarea. Botón de validar

A.2.4. Dar de alta incidencias

Los operarios que dispongan del permiso necesario pueden dar de alta nuevas incidencias en el sistema. Para ello, desde el menú de la pantalla inicial se deberá pulsar el botón "Dar de alta inicidencia". Al pulsarlo se accede al formulario con el que ingresar los datos de la incidencia (todas las imágenes que se muestran a continuación pertenecen a una misma pantalla con el formulario de ingreso de datos de la incidencia):

Figura 36. Alta de nueva tarea (1)

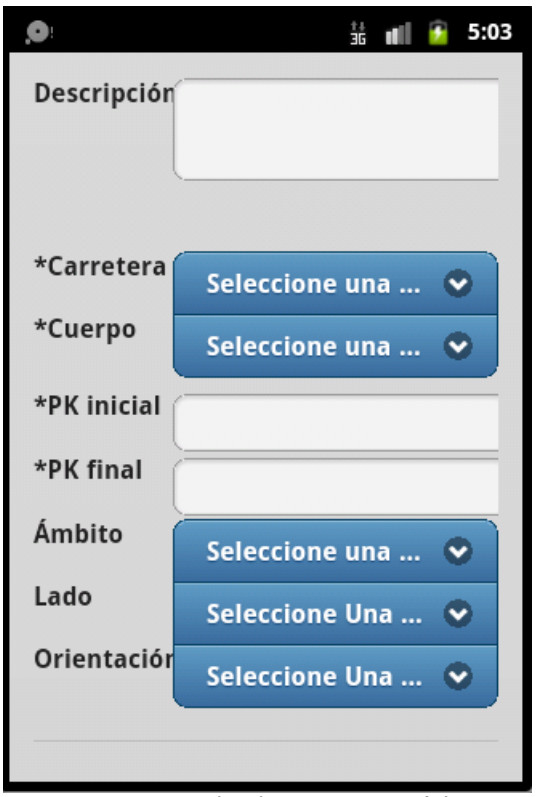

Figura 37. Alta de nueva tarea (2)

Con la ayuda del formulario irá ingresando la información:

- Primero deberá ingresar los datos completos del segmento y carretera de la que se trata con la ayuda de los desplegables y el resto de campos destinados a tal efecto.
- Podrá añadir imágenes, bien tomadas en ese momento o seleccionándolas de un repositorio (máximo 3 en cada formulario).
- Deberá seleccionar el tipo de agente que registra la incidencia
- Dependiendo del tipo de incidencia se muestra la actuación a llevar a cabo (acción fundamental y protocolo específico).
- Podrá añadir las observaciones que considere oportunas.
- Finalmente, dependiendo del tipo de incidencia y de la carta de servicios se especificarán las fechas en las que se debe finalizar la actuación.

Hay que remarcar que los campos marcados con el símbolo '*' deben ser rellenados obligatoriamente (de otro modo no podremos enviar la información).

Al pulsar el botón “Enviar” se tratará de enviar la información al servidor siguiendo el mismo procedimiento visto en los anteriores casos, es decir, gestionando adecuadamente los casos de ausencia de conexión.

Anexo B. Elementos clave del código de la aplicación

En este anexo comentaremos los puntos de la aplicación fundamentales para comprender su funcionamiento sin entrar en detalles de desarrollo demasiado técnicos. Estos elementos clave serán las bases de datos, la caché local del navegador, y algunas funciones empaquetadas para la aplicación y ampliamente utilizadas.

B.1. Base de datos

SQLite es el motor de bases de datos elegido por los desarrolladores de Android ya que ofrece características tan interesantes como su pequeño tamaño, no necesitar servidor, precisar poca configuración, ser transaccional y por supuesto ser de código libre.

Android incorpora de serie todas las herramientas necesarias para la creación y gestión de bases de datos SQLite, y entre ellas una completa API para llevar a cabo de manera sencilla todas las tareas necesarias. Sin embargo no entraremos en los detalles sino que en la presente sección se expondrá la estructura de la base de datos que se ha implementado para el desarrollo del sistema.

B.1.1. Tabla de usuarios

Primeramente se presenta la más sencilla de las tablas, aquella que recoge la información del usuario registrado. Hemos de recordar (si se necesita más información, consultar el apartado 3.2.1. Registro y salida en el sistema) que esta tabla nunca almacenará la información relativa a más de un usuario por motivos de seguridad. Por tanto, en esta tabla siempre habrá registrado uno o ningún usuario.

USUARIOS		
login	password	remote

Tabla 7. Usuarios

Como podemos observar, la información almacenada consta de tres campos:

- login: es el nombre de usuario.
- password: la contraseña del mismo.
- remote: éste es un parámetro que en el caso de la aplicación móvil siempre vale el string ”tareas”.

La información de esta tabla es consultada cada vez que se inicia la aplicación. Si existe algún usuario guardado, se envía al servidor su información automáticamente sin tener que pasar por el formulario de registro. Esto se conoce como ”guardar la sesión” o "login automático" y contribuye a la agilidad de la aplicación.

B.1.2. Tabla de reportes de actuación

TAREAS												
rep_id	ini_fin	datos_ok	comments	date	time	lat	lng	img_image1	img_image2	img_image3	hay_img	img_ok

Tabla 8. Informes de actuación

A continuación se verá la tabla donde se almacena toda la información relativa a los informes relativos a las tareas (iniciación, finalización y validación).

- rep_id: identificador de tarea. Se trata de un número asignado a cada tarea de manera exclusiva (cada tarea tiene un número distinto) para identificarla, por lo que si tenemos varios informes (iniciación, finalización y validación) sobre la misma tarea, todos ellos tendrán el mismo identificador.
- ini_fin: marcador de tipo de informe. Puede tomar valores ”ini” (si es un reporte de iniciación), ”fin” (finalización) y ”validar” (si se trata de una validación).
- datos_ok: registro de estado del envío de datos. Si este campo vale false, quiere decir que los datos del correspondiente formulario no han sido recibidos correctamente en el servidor, y por tanto el sistema tratará de reenviar este informe cuando corresponda. Si vale true, significa que los datos han llegado con éxito al servidor.
- comments: comentarios adicionales añadidos por el usuario en el formulario.
- date y time: fecha y hora en la que el informe fue redactado. Para evitar el falseamiento de esta información, son tomados automáticamente por el dispositivo y no se permite su modificación.
- lat y lng: coordenadas de geolocalización (latitud, longitud). Son también tomadas automáticamente por el dispositivo.
- img_imageX: estos tres campos almacenan la ruta y nombre (en el sistema de archivos local) de las imágenes adjuntas al reporte. Sirve para poder visualizar estas imágenes junto con el resto del informe posteriormente, además de para poder transferirlas.
- hay_img: registro de estado que indica si en el correspondiente informe se incluye, al menos, una imagen. En caso de que no haya imágenes adjuntas este campo valdrá false, de otro modo valdrá true.
- img_ok: registro de estado cuya función es similar a ”datos_ok”, pero relativo a las imágenes. Indica, por tanto, si las imágenes asociadas al reporte correspondiente han sido correctamente recibidas en el servidor (true) o si será necesario enviarlas posteriormente (false). La necesidad de dos registros de estado (uno para datos y otro para imágenes) viene impuesta por la separación en la transmisión de ambos elementos, ya que la transmisión de datos es más prioritaria y no requiere consideraciones aplicables a la transmisión de imágenes.

B.1.3. Tabla de nuevas tareas

NUEVAS TAREAS						
new_id	tarealD	descripcion	fecha_conocimiento	datos_ok	bloquelD	sectorlD
contratolD	claselD	tipolD	subtipolD	provincia	carreteralD	calzadalD
PKinicial	PKfinal	fuente_informacion	plazo_llegada	fecha_llegada	actuaciones_complementarias	observaciones
proc_gen	proc_esp	latitud	longitud	zoom	img_image1	img_image2
img_image3	userlD_alta	ambitolD	ladolD	orientacionlD	hay_img	img_ok

Tabla 9. Alta de nuevas tareas

En esta tabla quedarán registradas todas aquellas tareas que damos de alta rellenando el formulario correspondiente.

Como podemos observar, esta tabla contiene gran cantidad de campos ya que describe todos los detalles necesarios para dar de alta una nueva tarea en el servidor.

- new_id: es un identificador local de la tarea, pero no tendrá ninguna validez en el servidor.
- tareaID: es el identificador de tarea que aparecerá en la lista de tareas. Es exclusivo de cada tarea.
- descripción: detalles adicionales de la tarea.
- fecha_conocimiento: fecha tomada automáticamente al dar de alta una nueva incidencia.
- datos_ok: registro de estado cuya función es la misma que en el apartado anterior (B.1.2. Tabla de reportes de actuación).
- bloqueID: identificador de bloque. Bloque es la distinción entre las tareas programables y no programables (sólo damos de alta las no programables).
- sectorID: identificador de sector. Los sectores son las unidades en las que se divide la carretera.
- contratoID: identificador del contrato que regula la gestión de la carretera.
- claseID: identificador de la clase de la nueva incidencia.
- tipoID: identificador del tipo de la nueva incidencia.
- subtipoID: identificador del subtipo de la nueva incidencia.
- provincia: nombre de la provincia donde se originó la incidencia.
- carreteraID: identificador de la carretera donde se originó la incidencia.
- calzadaID: identificador de en qué plataforma (derecha, izquierda, mediana) se

originó la incidencia.

- PKinicial: punto kilométrico del inicio de la incidencia.
- PKfinal: punto kilométrico del final de la incidencia.
- fuente_informacion: indica quién informó de la incidencia.
- plazo_llegada: tiempo disponible (minutos) para realizar las actuaciones necesarias. Ese tiempo comienza a consumirse en cuanto se da de alta la tarea.
- fecha_llegada: fecha y hora en la que el operario deberá tener finalizada la actuación. Equivale a fecha_conocimiento + plazo_llegada.
- actuaciones_complementarias: protocolo de actuación establecido para una clase, tipo y subtipo.
- observaciones: otra información a aportar por el operario.
- proc_gen: procedimiento general de actuación del sector en el que se originó la incidencia.
- proc_esp: procedimiento específico de actuación asignado a un sector y un contrato.
- latitud y longitud: coordenadas de geolocalización de la incidencia.
- zoom: nivel de zoom con el que aparecerá el mapa de Google Maps que muestra la situación de la incidencia en la ficha de la lista de tareas.
- img_imageX: imágenes asociadas a la nueva incidencia. Guardan el nombre y la ruta de las imágenes para poder ser enviadas más tarde.
- userID_alta: identificador del usuario que da de alta la incidencia.
- ambitoID: identificador del ámbito.
- ladoID: identificador del lado.
- orientacionID: identificador de la orientación. Junto con el ámbito y el lado, conforma la georeferencia transversal.
- hay_img: registro de estado que vale true si alguno de los campos img_imageX contiene la ruta de una imagen. Vale false si todos los campos están vacíos (es decir, el formulario no tiene imágenes adjuntas).
- img_ok: registro de estado que indica si las imágenes han sido correctamente enviadas al servidor.

B.1.4. Tabla de la lista de tareas

Esta será la tabla que almacena la lista de tareas que nos llega del servidor. Es la que contiene una mayor cantidad de campos ya que es la que más información necesita almacenar. Muchos campos son idénticos a los del apartado anterior (B.1.3. Tabla de alta de nuevas tareas) y por tanto sólo se comentarán aquellos que no estuvieran presentes en dicho apartado.

- inicio_actuacion: registro de estado que vale true si la incidencia había sido iniciada cuando descargamos la lista del servidor por última vez.

- periodo_transcurrido: minutos que han pasado desde la fecha de conocimiento hasta el momento actual.
- dif_temporal: control relativo a las diferencias horarias para evitar manipulaciones por parte de los usuarios. En pruebas (sin implementar).
- comprobacion_presencia: verificación de presencia física en la realización de la tarea. En pruebas (sin implementar).
- userID_baja: identificador del usuario que finalizó la tarea.
- fin_actuacion: registro de estado que vale true si la incidencia había sido finalizada cuando descargamos la lista del servidor por última vez.
- calzada: texto que nombra la parte de la vía en la que sucedió la incidencia (derecha, izquierda, mediana...).
- validada: registro de estado que vale true si la incidencia había sido validada cuando descargamos la lista del servidor por última vez.
- pk_hito_ini: hito kilométrico inicial.
- pk_dst_hito_ini: distancia al hito kilométrico inicial.
- pk_hito_fin: hito kilométrico final.
- pk_dst_hito_fin: distancia al hito kilométrico final.
- tipo: texto que nombra el tipo de incidencia. Esta información se complementa con su identificador (tipoID).
- bloque: texto que nombra el bloque de la incidencia. Esta información se complementa con su identificador (bloqueID).
- clase: texto que nombra la clase de incidencia. Esta información se complementa con su identificador (claseID).
- contrato: texto que nombra el contrato que regula la explotación de la carretera donde sucedió la incidencia. Esta información se complementa con su identificador (contratoID).
- sector: texto que nombra el sector donde sucedió la incidencia. Esta información se complementa con su identificador (sectorID).
- permissionID: identificador de los permisos de los usuarios que pueden actuar sobre la incidencia. Generalmente se relaciona con el sector.
- carretera: texto que describe el sector donde sucedió la incidencia. Esta información se complementa con su identificador (sectorID).
- antelacion_aviso: indicador relativo a las tareas programadas. Sirve para mostrar un aviso antes de que aparezca la tarea programada.
- fecha_inicial: indicador relativo a las tareas programadas. Fecha en la que se realizó la programación de la tarea.
- modo_repeticion: indicador relativo a las tareas programadas. Frecuencia con la que se repite una tarea programada (anual, mensual...).
- img_rep_ini_status y img_rep_fin_status: registros de estado que controlan si

el reporte de iniciación y finalización, respectivamente, fueron redactados o no, bien sea porque al descargar la lista del servidor ya estaban redactados o bien porque el usuario los ha redactado sin haber podido enviar la información al servidor. Controlan los iconos que muestran el estado de los reportes (terminado o no) en la lista de tareas y en la ficha.

B.2. Caché local

Al igual que los navegadores de Internet, las aplicaciones Android permiten el almacenamiento de datos en una memoria caché local. Existen dos tipos en esta memoria, las variables de sesión y las variables locales. Aunque dichos nombres son una herencia del uso de este tipo de variables en los navegadores, siguen teniendo vigencia para las aplicaciones móviles

Mientras que las variables de sesión, como su nombre indica, están asociadas a la sesión del navegador y su contenido se volatilizará cuando cerremos dicha pestaña del navegador, las segundas permanecen almacenadas hasta que el sistema reciba la orden de borrarlas (el usuario puede programar esta acción). De esta manera, con la memoria caché local conseguimos un almacenamiento permanente muy útil que será utilizado a modo de variables globales en la aplicación.

Las variables más relevantes de este tipo se comentan a continuación:

- localStorage.pfc_sector_val: guarda el sector seleccionado por el usuario en la configuración.
- localStorage.pfc_tramos_val: guarda los tramos seleccionados por el usuario en la configuración.
- localStorage.pfc_solo_wifi: indica si las imágenes se enviarán sólo cuando esté disponible una red wifi o por el contrario se tratará de utilizar también una red 3G.
- localStorage.pfc_f_act_lista: frecuencia de actualización automática de la lista.
- localStorage.pfc_auto_lista: indica si la actualización automática de la lista está activada o no.
- localStorage.pfc_f_act_envios: frecuencia de envío automático de la información pendiente.
- localStorage.pfc_auto_envios: indica si el envío automático de la información pendiente está activado o no.
- localStorage.pfc_inicio: indica si es la primera vez que ejecutamos la aplicación tras instalarla (para realizar los ajustes iniciales).
- localStorage.pfc_tabla2_existe: indica si la lista de tareas no existe o existe pero está vacía.
- localStorage.pfc_img_width: contiene el valor, en píxels, del ancho de las imágenes tomadas por la cámara en el entorno de la aplicación.
- localStorage.pfc_img_height: contiene el valor, en píxels, del alto de las imágenes tomadas por la cámara en el entorno de la aplicación.

- localStorage.pfc_fuente_infoID: para el usuario registrado, contiene la lista de agentes que pueden dar de alta una nueva tarea.
- localStorage.pfc_userID: identificador del usuario registrado.
- localStorage.pfc_permisos: permisos del usuario registrado.
- localStorage.pfc_reporte_hecho: indica si el informe seleccionado ya había sido rellenado.
- localStorage.pfc_inicio: indica si es la primera vez que ejecutamos la aplicación.

B.3. Librería de la aplicación

Como ya se ha adelantado anteriormente, durante el desarrollo del código de la aplicación de este PFC se fueron creando una serie de funciones genéricas que podrían aprovecharse en futuros trabajos. Otras, se crearon a partir de la generalización de diversos módulos muy similares que cumplían funciones análogas (aunque con ligeras diferencias). Simplemente añadiendo parámetros de configuración a la función, obteníamos un módulo de gran potencia dada su versatilidad, de modo que el programador puede controlar el funcionamiento de esa función. Esas diversas funciones de la misma familia se sustituyeron por unas pocas que diferían ligeramente en su comportamiento en función de los parámetros al ser llamadas.

A continuación se citan algunas de esas funciones, explicando su cometido y el por qué de su relevancia.

B.3.1. Funciones genéricas para la gestión de la base de datos local

insert_Sql (table_name, columns_names, data_values, tx)

Inserta una nueva fila de datos en la tabla.

> *table_name*: string con el nombre de la tabla.
>
> *columns_names*: array que contiene los nombres de las columnas que queremos rellenar.
>
> *data_values*: array con los valores a insertar. Debe estar ordenado con respecto a ”columns_names”.
>
> *tx*: objeto ”tx” de la transacción.

create_Sql (table_name, columns_names, tx)

Crea una nueva tabla si no existía previamente.

> *table_name*: string con el nombre de la tabla.
>
> *columns_names*: array que contiene los nombres de las columnas de nuestra nueva tabla.
>
> *tx*: objeto "tx" de la transacción.

save (table_name, columns_names, saved_values, save_callback_success, save_callback_error)

Esta función intenta insertar un conjunto de datos en una tabla y si tiene éxito ejecuta la función *save_callback_success*. Si falla, llama a la función *save_callback_error* para capturar el error.

table_name: string con el nombre de la tabla donde guardaremos los datos.

columns_names: array que contiene los nombres de las columnas que queremos rellenar.

saved_values: array con los valores a insertar. Debe estar ordenado con respecto a "columns_names".

save_callback_success: función llamada si tenemos éxito al guardar la información.

save_callback_error: función llamada si algo va mal durante la ejecución de "save". Recibirá un parámetro err, que contiene las propiedades "code" y "message".

extract (table_name, columns_names, where_clause, extract_callback_success, extract_callback_error)

Esta función extrae información de una tabla, teniendo la opción de incluir una cláusula condicional. Si la consulta tiene éxito se ejecutará la función *extract_callback_success*. Si falla, llama a la función *extract_callback_error* para capturar el error.

table_name: string con el nombre de la tabla de donde extraemos los datos.

columns_names: array que contiene los nombres de las columnas de donde queremos extraer.

where_clause: en principio, es un string que indica las condiciones que deben de cumplir los datos a extraer (p. ej.: campo ID=12). Si no se desea una búsqueda condicional, "where_clause" deberá tomar el valor booleano "false".

extract_callback_success: función llamada si tenemos éxito al extraer la información. Recibirá tres parámetros:

- *extracted_info_array*: array bidimensional de índice numérico con la información extraída.
- *extracted_indexes*: array de índice numérico con los índices (nombres de las columnas).
- *extracted_info_obj*: objeto de índice numérico en su primera dimensión y asociativo en su segunda (con los índices "extracted_indexes") que contiene la información extraída.

extract_callback_error: función llamada si algún error ocurre durante la extracción. Recibirá un parámetro err, que contiene las propiedades "code" y "message".

drop (array_tables, drop_callback_success, drop_callback_error)

Esta función elimina, si existen, las tablas recibidas como parámetro.

array_tables: array de strings que contiene los nombres de las tablas a eliminar.

drop_callback_success: función de callback llamada en caso de que la operación tenga éxito.

drop_callback_error: función de callback llamada en caso de que la eliminación falle.

B.3.2. Funciones específicas

auto_login (success_callback, tipo_llamada, no_conexion_callback)

Esta función realiza un registro transparente al usuario, de manera que continuará ejecutando la función que se le pasa como parámetro (*success_callback*) si es capaz de hacer el login correctamente.

success_callback: función llamada si el registro automático tiene éxito.

tipo_llamada: modo en el que trabaja la función (inicio, activo y pasivo o *background)*

no_conexion_callback: función llamada en caso de que no exista conexión a internet.

success_save (tipo_llamada, tipo_llamada_auto_login, objetivo, envio_success_callback)

Función que se ejecuta después de haber guardado los datos del formulario con éxito. Extrae la información asociada a los formularios que no han sido enviados y realiza el envío de los mismos.

tipo_llamada: modo en el que se ejecutará la función (nuevo envío, envío general, automático).

tipo_llamada_autologin: cuando llamemos a la función de *auto_login*, modo en el que se llamará ésta.

objetivo: indica si la información a enviar se trata de informes de iniciación/finalización, de una nueva alta o de una validación.

envio_success_callback: función llamada tras realizar el envío.

comprobar_envio_imagenes (tipo_llamada, tipo_llamada_auto_login, objetivo, envio_success_callback)

En esta función se comprueba si hay que enviar imágenes, es decir, por un lado hay que verificar que la conexión y configuración permiten el envío de imágenes y por otro lado hay que comprobar si hay imágenes en nuestra BD que deban ser enviadas. Después, llama a la función encargada de enviar las imágenes.

Los parámetros de esta función cumplen el mismo papel que en la anterior (*success_save*).

enviar_imagenes (img_info_array, parametros_img, tipo_llamada, tipo_llamada_auto_login, objetivo, envio_success_callback)

Esta función realiza el envío al servidor de las imágenes y procesa su respuesta para después llamar a la función de actualización de la base de datos local.

img_info_array: contiene las rutas a las imágenes que han de enviarse.

parametros_img: objeto que contiene una serie de parámetros sobre las imágenes que han de enviarse.

El resto de parámetros cumplen el mismo papel que en las anteriores funciones.

update_data (respuesta, parametros, tipo_llamada, tipo_llamada_auto_login, objetivo, envio_success_callback)

En esta función se realiza la actualización de la base de datos local, marcando como enviado aquello que ha sido recibido con éxito en el servidor.

respuesta: es la respuesta que nos llega del servidor, con ella determinaremos que datos han sido registrados en el mismo.

parametros: objeto que contiene una serie de parámetros sobre la información enviada.

envio_success_callback: función llamada al finalizar la ejecución.

El resto de parámetros cumplen el mismo papel que en las anteriores funciones.

Modos de ejecución

Hemos visto que estas funciones incluían parámetros que eran descritos como ”modo en el que se ejecutará la función”. La razón de estos parámetros es disponer de una manera de modificar ligeramente el comportamiento de la función dependiendo del contexto en el que se ejecute.

Pongamos como ejemplo la función de envío de la información (*success_save*). Existen tres posibles escenarios en el que dicha función se ejecutará:

- El usuario ha rellenado un formulario y ha pulsado en enviar. En este caso, mientras se produce el envío se muestra en pantalla un mensaje de espera, y al finalizar se informa del éxito o fracaso del intento y se vuelve a la pantalla anterior.
- El usuario, desde el submenú de opciones, ha seleccionado enviar toda la información de golpe. En este caso, necesitamos seguir mostrando un mensaje mientras se producen los envíos. Pero sería deseable que no se mostrara un mensaje indicando éxito o fracaso por cada uno de los formularios pendientes (en caso de tener un gran número esto sería tedioso) sino mostrar solamente un mensaje al final, con un resumen de la información que ha podido ser enviada y los errores producidos. Permanecemos en la misma pantalla.
- Envío automático de la información. Esta tarea por definición se realiza de manera transparente al usuario para no interferir con la actividad que éste realizase en el momento. Por tanto, no debe mostrar ningún tipo de mensaje (ni siquiera aunque no haya conexión, en ese momento simplemente abortará el intento) ni tampoco cambiar de pantalla. Debe ser indetectable para el usuario.

B.3.3. Funciones de propósito general

asToNum (arr)

Esta función toma un array asociativo (o un objeto) y devuelve un array igual pero de

índice numérico.

purga_array (arr1,pos,arr2)

Esta función toma un array numérico y elimina sus filas cuyo elemento en la posición "pos" vale "false".

get_ind (arr)

Esta función toma un array asociativo (o un objeto) y devuelve un array numérico con los nombres de sus índices.

tx_server (tx_url, tx_type, tx_dataType, tx_data, tx_server_success_callback, tx_server_error_callback).

Esta función realiza una llamada ajax al servidor para enviar datos.

tx_url: dirección URL del servidor.

tx_type: método de envío (POST/GET).

tx_dataType: formato de respuesta (JSON, html, script...)

tx_data: datos a enviar.

tx_success_callback: función llamada en caso de que el envío tenga éxito.

tx_error_callback: función llamada en caso de que el envío falle.

tx_server_images (image_URI, image_parameters, url_server_images, tx_server_images_success_callback, tx_server_images_error_callback)

Esta función realiza una llamada AJAX al servidor para enviar imágenes.

image_URI: ruta local donde se encuentra la imagen a enviar.

image_parameters: objeto que contiene información adicional sobre la imagen.

url_server_images: dirección URL del servidor.

tx_server_images_success_callback: función llamada en caso de que el envío tenga éxito.

tx_server_images_error_callback: función llamada en caso de que el envío falle.

get_fecha_hora ()

Devuelve la fecha y la hora locales (del cliente).

geoloc (tipo_llamada,callback)

Obtiene las coordenadas de la geolocalización y las coloca en y , respectivamente.

tipo_llamada: modo en el que se ejecutará la función.

callback: función llamada en caso de conseguir establecer la geolocalización.

google_maps (lat,long,zoom)

Esta función nos inserta un mapa de google maps en .

lat: latitud del marcador que aparecerá en el centro del mapa.

long: longitud del marcador que aparecerá en el centro del mapa.

zoom: nivel de zoom que mostrará el mapa creado.