TFC - iCMED

Consulta cuadro médico de compañía aseguradora de salud para dispositivos móviles basados en iOS

José Mariano García Luengo

Trabajo Fin de Carrera
Ingeniería Técnica de Informática de Sistemas

Consultor: Roberto Ramírez Vique

MEMORIA - Junio 2013

A mi familia:

A mi madre y hermanos. Mamá, Silvia y Óscar, os necesito.

A mi mujer, que me animó continuamente y ha padecido todas mis horas de trabajo. Lola, te quiero.

A mis niñas, mi motivo principal para continuar adelante, aunque fuese a cambio del tiempo que no he podido estar con ellas. Raquel y Mónica, mis pequeños grandes tesoros.

A mis abuelos, que aunque ya no están conmigo, los quiero con locura.

A mi PADRE, que en paz descanse, allí donde estés ¡Por fin lo conseguí!

Tabla de contenido

Índice de ilustraciones

Índice de tablas

1 Introducción

1.1 Contexto del proyecto y situación actual

El presente proyecto se encuentra enmarcado en las necesidades de información a los clientes de una compañía de gestión de seguros de salud que a partir de ahora denominaremos GESESA.

Esta compañía tiene un teléfono de atención urgente para horas fuera de oficina que ofrece información profesional médica a sus asegurados.

Una de las consultas que se realiza con mayor frecuencia a este teléfono de urgencias es sobre el cuadro médico que la compañía ofrece a sus asegurados.

Hasta ahora, en GESESA se viene dando respuesta al servicio de urgencias médicas a través de extracciones periódicas de la base de datos de médicos almacenadas en un ordenador portátil que lleva la persona que atenderá este servicio. A esta base de datos se accede mediante un aplicativo muy sencillo y simple desarrollado en MS Access.

Con el proyecto que nos ocupa se pretenden evitar las vulnerabilidades y deficiencias del actual procedimiento.

La principal vulnerabilidad reside en que, aunque el portátil está securizado con accesos controlados a él y el disco duro cifrado, es factible que los datos puedan ser accedidos de forma masiva por personal no autorizado ya que los mismos están en el disco duro del portátil.

Por otro lado, el servicio puede ser deficiente en algún caso ya que los datos no están actualizados en el momento de las consultas, lo cual redundaría en una mala imagen de los clientes hacia la compañía aseguradora.

1.2 Objetivos

1.2.1 Objetivos generales

Los objetivos generales del proyecto son:

- Afianzar los conocimientos adquiridos anteriormente.
- Obtener destreza en la planificación de proyectos de desarrollo de aplicaciones para dispositivos móviles.
- Obtener experiencia y ampliar conocimientos en el desarrollo de aplicaciones iOS

1.2.2 Objetivos del proyecto

El objetivo principal del proyecto es que la consulta de datos de cuadro médico se realice en tiempo real y no en diferido como hasta ahora se viene haciendo.

Adicionalmente, se consigue proteger el acceso a los datos corporativos ya que estos estarán ubicados en los servidores de base de datos de GESESA.

También se van a permitir funcionalidades de geolocalización que hará más fácil dirigir al asegurado hasta el centro médico de su interés.

Por último, se deja abierta la posibilidad de implementación de nuevos servicios, tanto para asegurados, como para los médicos que atienden las urgencias de dichos asegurados.

1.2.3 Objetivos como asignatura

El objetivo del Trabajo de Fin de Carrera (TFC) es el de consolidar los conocimientos adquiridos a lo largo de los estudios de Ingeniería Técnica de Informática de Sistemas.

1.3 Funcionalidades

Las funcionalidades incluidas a desarrollar en este proyecto son:

1.3.1 Servidor

- Desarrollo php acceso a base de datos: esta funcionalidad contiene el acceso a la base de datos desde el servidor Web.
- Desarrollo de procedimientos almacenados para el acceso a cuadro médico de forma dinámica en función de los parámetros de entrada (población, provincia y/o especialidad médica). Se intentará desarrollar un único procedimiento almacenado que dependiendo de la opción elegida y los parámetros que se facilitan, ejecute un apartado u otro del procedimiento almacenado.
- Implementar servicios Web.

1.3.2 Cliente

- Entrada a la aplicación con usuario y contraseña.
- Búsqueda de médico, hospital o centro médico por especialidad médica, por provincia y/o por población.
- Acceso al servicio Web.
- Geoposicionamiento del asegurado.
- Geoposicionamiento del medico, hospital o centro médico.
- Como llegar a... Esta opción nos permitirá trazar una ruta desde la posición actual hasta el médico elegido.
- Adicionalmente se ha incluido la búsqueda de asegurados.

1.4 Usuarios de la aplicación

Se pretende el uso de la aplicación principalmente por los médicos y por los propios asegurados.

No es necesaria la figura del usuario administrador.

1.5 Planificación del proyecto

Debemos conocer las tareas a realizar para así poder realizar una correcta planificación del proyecto y adaptar las mismas a las fechas propuestas en el plan docente del trabajo de fin de carrera.

Acto seguido deberemos distribuir la carga de trabajo entre las distintas fases planificadas.

1.5.1 Plazos

La fecha de finalización del proyecto y su entrega es el 10 de junio de 2013.

1.5.2 Fases del plan docente

Las fases del plan docente son las siguientes:

- Plan de trabajo y análisis preliminar de requisitos.
- Análisis de requisitos, diseño conceptual y técnico.
- Implementación.

1.5.3 Adaptación de las fases al plan docente

El desglose de las tareas en función de las entregas propuestas en el plan docente es como sigue:

- PEC 1: Plan de trabajo y análisis preliminar:
  - Selección del proyecto.
  - Preparación del proyecto.
  - Definición del proyecto.
  - Planificación del proyecto.
- PEC 2: Análisis de requisitos y diseño técnico:
  - Requisitos funcionales.
  - Requisitos no funcionales.
  - Arquitectura.
  - Clases.
  - Procedimientos almacenados.
  - Estudio alternativas tecnológicas.
- PEC 3: Implementación:
  - Desarrollo servicio Web.
  - Desarrollo aplicación Web.
  - Desarrollo procedimientos almacenados.
  - Desarrollo aplicación dispositivo móvil.
- Entrega final:
  - Elaboración memoria final.
  - Presentación proyecto.

1.5.4 Ciclo de vida

Dado que los objetivos del proyecto están claramente definidos y podemos definir perfectamente las distintas fases del mismo, utilizaremos un ciclo de vida clásico o en cascada para la estimación del volumen de trabajo y la planificación de los mismos.

La planificación temporal del proyecto se realizará con Microsoft Project.

1.5.5 Hitos

En el siguiente cuadro se muestra una temporalización inicial de todas las tareas que conforman el proyecto:

Análisis previo y planificación	**13 días**	**27/02/2013**	**11/03/2013**
Selección del proyecto	2 días	27/02/2013	28/02/2013
Preparación del proyecto	2 días	01/03/2013	02/03/2013
Definición del proyecto	4 días	03/03/2013	06/03/2013
Planificación del proyecto	1 día	07/03/2013	07/03/2013
Creación del documento	4 días	08/03/2013	11/03/2013
Entrega PEC1 (11/03/2013)		11/03/2013	
Análisis de requisitos	**5 días**	**12/03/2013**	**16/03/2013**
Especificación requisitos funcionales	4 días	12/03/2013	15/03/2013
Especificación requisitos no funcionales	1 día	16/03/2013	16/03/2013
Diseño	**23 días**	**17/03/2013**	**08/04/2013**
Arquitectura	2 días	17/03/2013	18/03/2013
Clases	2 días	19/03/2013	20/03/2013
Procedimientos almacenados	2 días	21/03/2013	22/03/2013
Estudio alternativas tecnológicas	12 días	23/03/2013	03/04/2013
Preparación documento	5 días	04/04/2013	08/04/2013
Entrega PEC2 (08/04/2013)		08/04/2013	
Implementación y pruebas	**42 días**	**09/04/2013**	**20/05/2013**
Desarrollo servicio Web	7 días	09/04/2013	15/04/2013
Desarrollo aplicación Web	6 días	16/04/2013	21/04/2013
Desarrollo procedimientos almacenados	6 días	22/04/2013	27/04/2013
Desarrollo aplicación dispositivo móvil	20 días	28/04/2013	17/05/2013
Preparación entregable y guía de uso	3 días	18/05/2013	20/05/2013
Entrega PEC3 (20/05/2013)		20/05/2013	
Finalización	**20 días**	**21/05/2013**	**10/06/2013**
Redacción memoria final	10 días	21/05/2013	30/05/2013
Creación presentación virtual	10 días	31/05/2013	10/06/2013
Entrega final (10/06/2013)		10/06/2013	

Tabla 1: Hitos del proyecto

El diagrama de Gant que representa la planificación del proyecto es el siguiente:

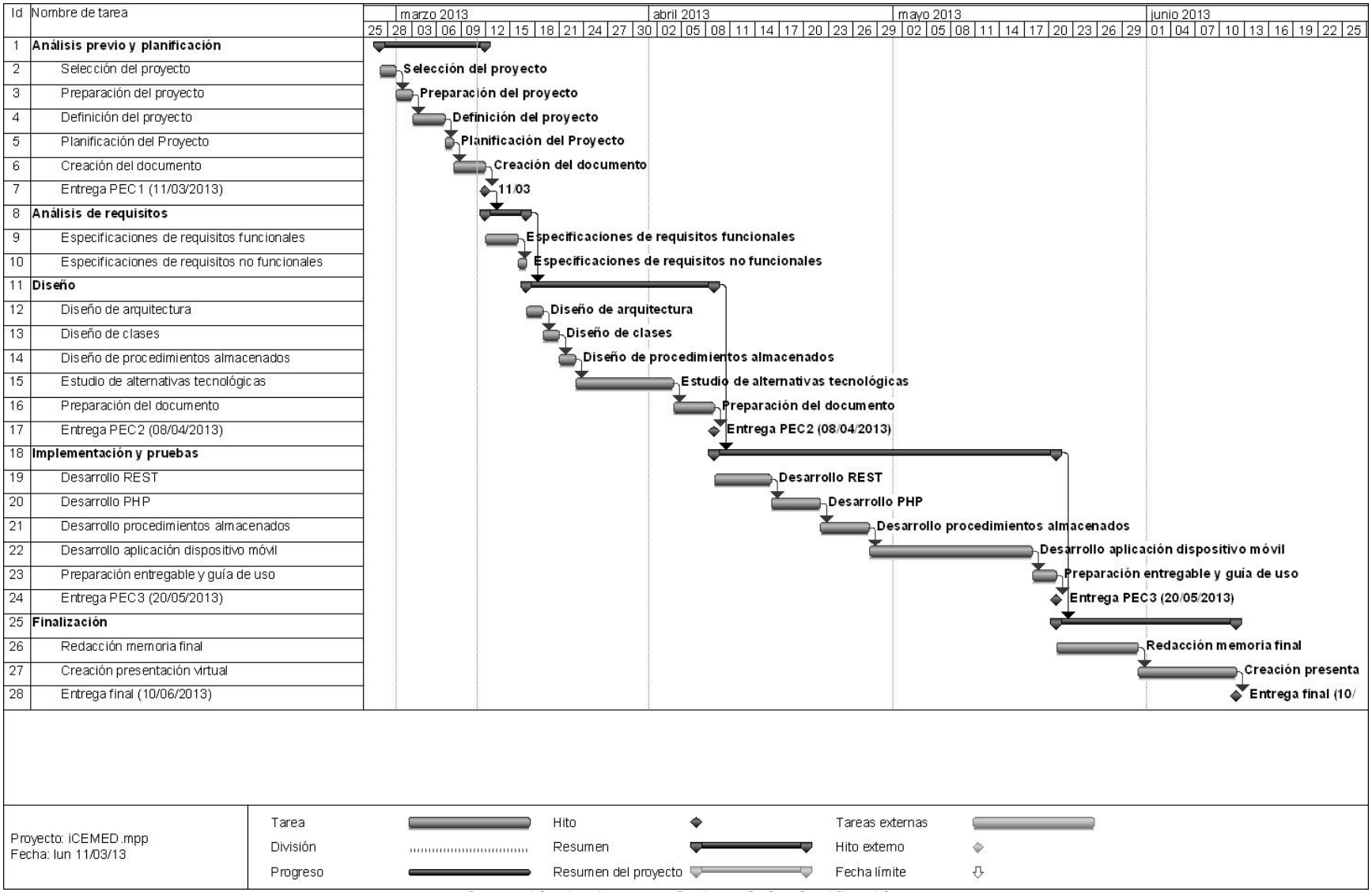


Ilustración 1: Diagrama de Gant de la planificación

1.6 Riesgos del proyecto

Los riesgos a los que está sujeto el presente proyecto son de diferentes tipos y se especifican a continuación:

1.6.1 Riesgos por trabajo

A fecha actual este riesgo es casi inexistente dada mi situación de desempleo. Si se deben tener en cuenta la preparación y asistencia a entrevistas de trabajo, pero esto no debe afectar al proyecto.

Si la situación cambiase y comenzase a trabajar, habría que tener en cuenta las posibles cargas de trabajo iniciales, desplazamiento, sesiones de formación, etc. Llegado el caso (esperemos que así sea) se deberán realizar ajustes al calendario y tener una mayor dedicación los fines de semana.

1.6.2 Riesgos por enfermedad

Dependiendo del motivo de enfermedad y si esta es leve o grave habría que realizar ajustes en el caso de enfermedad con una duración superior a 3 días.

De todos modos y dada la situación de desempleo, el tiempo disponible para el proyecto es suficiente para afrontar cualquier enfermedad que se pudiese dar durante el desarrollo del proyecto.

1.6.3 Riesgos técnicos

Es posible que surjan problemas de índole técnica en el transcurso del proyecto. Para minimizar estos, se toman las siguientes medidas:

- Ya durante el desarrollo de la PEC1 se han instalado y configurado parte del hardware y software necesario para el proyecto.
- Se dispone de otro ordenador de similares características que podría sustituir al ordenador de trabajo en caso de que este fallase.
- Se realizan copias de seguridad de la información del TFC en disco duro externo para minimizar pérdidas de información en caso de fallo del disco duro del ordenador que se está utilizando para el proyecto.

1.6.4 Riesgos introducidos por cursar otras asignaturas

Paralelamente al TFC, también se está cursando la asignatura de 'Seguridad en Redes de Computadores' (SRC). En caso de solapamiento de trabajos y/o entregas de esta asignatura con el TFC se le dará prioridad al TFC, e incluso se planteará el abandono de la asignatura SRC.

2 Análisis funcional

2.1 Diagramas de casos de uso

En este proyecto sólo se considerarán dos tipos de actores (coordinador médico y asegurado). El asegurado tendrá acceso a la consulta de cuadro médico, mientras que el coordinador médico podrá acceder también a esta consulta; el coordinador médico también tendrá acceso a los datos de las pólizas de asegurado para poder prestarle el servicio para el cual este le esté llamando.

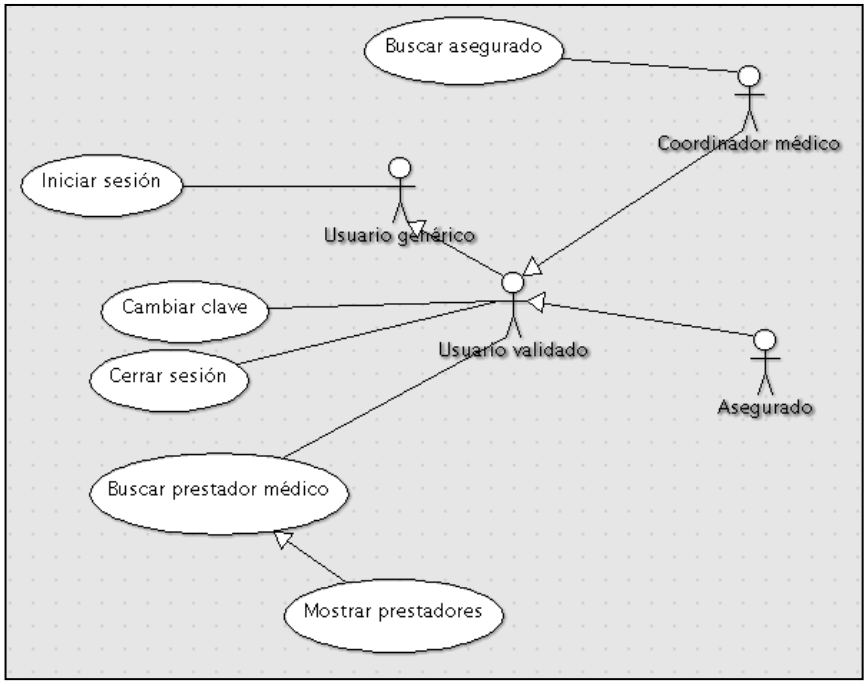


Ilustración 2: diagrama de casos de uso

Podemos ver que el sistema tiene 4 actores distintos:

- Usuario Genérico: este usuario deberá identificarse en el sistema para poder acceder al resto de funcionalidades
- Usuario validado: es una especialización del usuario genérico y que tiene acceso a las funcionalidades de la aplicación cambiar clave, cerrar sesión y buscar prestador médico.
- Asegurado: se trata de una especialización del usuario validado que tendrá acceso a las funcionalidades de este y únicamente accederá a sus datos de asegurado y no a los de los demás asegurados.
- Coordinador Médico: es otra especialización del usuario validado y además de acceder a las funcionalidades de dicho actor, tendrá acceso a buscar asegurados.

No es necesaria la figura de un usuario administrador porque toda la gestión de usuarios se realiza automáticamente en la aplicación corporativa cuando un cliente contrata una póliza de salud.

En el siguiente cuadro podemos ver resumidos los diferentes casos de uso:

Descripción del caso de uso	Actores
Iniciar sesión en el sistema	Usuario genérico
Cerrar sesión actual	Usuario validado Coordinador médico Asegurado
Cambiar la clave del usuario activo	
Buscar prestador médico	
Mostrar prestadores médicos	
Mostrar posición geográfica consulta del prestador médico	
Buscar asegurado	Coordinador médico

Tabla 2: Resumen de los casos de uso

2.2 Descripción de los casos de uso

2.2.1 Caso de uso Iniciar sesión

Nombre	Iniciar sesión
Resumen	En este caso de uso se contempla la identificación de un usuario en el sistema
Actores	Usuario genérico
Precondición	El usuario no tiene ninguna sesión activa
Postcondición	El usuario, o bien ha iniciado la sesión, o bien no ha iniciado la sesión
Flujo Normal	1. Este caso de uso inicia cuando el usuario accede por primera vez al sistema o cuando se finaliza una sesión. 2. El sistema pide al actor su nombre de usuario y su clave. 3. El usuario introduce los datos solicitados y los valida. 4. El sistema comprueba si los datos introducidos son correctos. 5. Si la información introducida es correcta la sesión queda iniciada. 6. Se muestra la pantalla principal de la aplicación. 7. Finaliza el caso de uso.
Flujos alternativos	- Si los datos de identificación facilitados en 3 son incorrectos se informa al usuario del error y se vuelve a 2. - El usuario puede terminar en cualquier momento la aplicación.
Inclusiones	Ninguna
Extensiones	Ninguna

Tabla 3: Caso de uso Iniciar sesión

2.2.2 Caso de uso Cerrar sesión

Nombre	Cerrar sesión
Resumen	En este caso de uso se contempla el cierre de la sesión de un usuario previamente validado en el sistema
Actores	Usuario validado
Precondición	El usuario tiene una sesión activa
Postcondición	El usuario no tiene ninguna sesión activa
Flujo Normal	1. Este caso de uso inicia cuando el usuario pulsa en cualquier momento la opción de cierre de sesión. 2. El sistema muestra un aviso de confirmación para el cierre de la sesión. 3. El usuario confirma el cierre de sesión. 4. El sistema finaliza la sesión con el usuario. 5. Se ejecuta el caso de uso de Iniciar sesión y finaliza este caso de uso.
Flujos alternativos	- El actor en el paso 3 decide no confirmar el cierre de sesión y se finaliza el caso de uso.
Inclusiones	Ninguna
Extensiones	Ninguna

Tabla 4: Caso de uso Cerrar sesión

2.2.3 Caso de uso Cambiar clave

Nombre	Cambiar clave
Resumen	En este caso de uso se contempla el cambio de clave de un usuario con sesión activa.
Actores	Usuario validado
Precondición	El usuario tiene una sesión activa
Postcondición	El usuario, o bien ha cambiado su clave de acceso, o bien ha cancelado el proceso
Flujo Normal	1. Se inicia cuando el actor pulsa la opción cambio de clave. 2. El sistema muestra una pantalla donde se solicita la clave actual, la nueva clave y la confirmación de la nueva clave. 3. El usuario introduce los datos requeridos y los valida. 4. El sistema cambia la clave del usuario y acaba el caso de uso.
Flujos alternativos	- Si la clave actual no coincide con la clave del usuario se informa del error y se vuelve al paso 2. - Si la nueva clave y su confirmación no coinciden se informa del error y se vuelve al paso 2. - Si la nueva clave no tiene al menos una longitud de 8 caracteres se informa del error y se vuelve al paso 2. - El usuario puede cancelar el proceso de cambio de clave en cualquier momento, y finalizaría este caso de uso.
Inclusiones	Ninguna
Extensiones	Ninguna

Tabla 5: Caso de uso Cambiar clave

2.2.4 Caso de uso Buscar prestador médico

Nombre	Buscar prestador médico
Resumen	En este caso de uso se contempla la búsqueda de prestadores médicos en el catalogo de la compañía aseguradora
Actores	Usuario validado
Precondición	El usuario tiene una sesión activa
Postcondición	Ninguna
Flujo Normal	1. Este caso de uso inicia cuando el usuario elige la opción de buscar servicio médico de la pantalla con las diferentes opciones de la aplicación. 2. El sistema muestra los posibles criterios de búsqueda. 3. El usuario introduce los datos requeridos. 4. El sistema muestra un listado con los prestadores médicos que cumplen los criterios establecidos. 5. Finaliza el caso de uso.
Flujos alternativos	- El actor puede mostrar todos los datos de un prestador médico concreto, posicionando su domicilio de prestación en un mapa al pulsar sobre él en la lista (caso de uso Mostrar prestadores).
Inclusiones	Ninguna
Extensiones	Mostrar prestadores

Tabla 6: Caso de uso Buscar prestador medico

2.2.5 Caso de uso Mostrar prestadores

Nombre	Mostrar prestadores
Resumen	En este caso de uso se muestra una ficha con los datos del prestador médico y la posición en un mapa de su domicilio.
Actores	Usuario validado
Precondición	El usuario tiene una sesión activa
Postcondición	Ninguna
Flujo Normal	1. Este caso de uso inicia cuando el usuario pulsa sobre uno de los resultados devueltos por el caso de uso Buscar prestador médico. 2. El sistema muestra una ficha con los datos del prestador médico y un mapa donde sitúa el domicilio de prestación. 3. Finaliza el caso de uso.
Flujos alternativos	- El actor puede interactuar con el mapa realizando zoom y viendo su posición actual en el mismo.
Inclusiones	Ninguna
Extensiones	Ninguna

Tabla 7: Caso de uso Mostrar prestadores

2.2.6 Caso de uso Buscar asegurado

Nombre	Buscar asegurado
Resumen	En este caso de uso se realiza la búsqueda de un asegurado en la base de datos de clientes de la compañía
Actores	Coordinador médico
Precondición	El usuario tiene una sesión activa
Postcondición	Ninguna
Flujo Normal	1. Este caso de uso inicia cuando el actor elige la opción de buscar asegurado de la pantalla con las diferentes opciones de la aplicación. 2. El sistema muestra los posibles criterios de búsqueda. 3. El usuario introduce los datos requeridos. 4. Si el sistema encuentra el asegurado muestra los datos básicos del mismo. 5. Finaliza el caso de uso.
Flujos alternativos	- En caso de no encontrarse el asegurado el sistema informa de ello y finaliza el caso de uso.
Inclusiones	Ninguna
Extensiones	Ninguna

Tabla 8: Caso de uso Buscar asegurado

2.3 Diagrama del modelo conceptual

El modelo conceptual contiene las entidades de datos necesarias para el correcto funcionamiento de la aplicación.

Estas entidades principales se encuentran reflejadas en el siguiente diagrama:

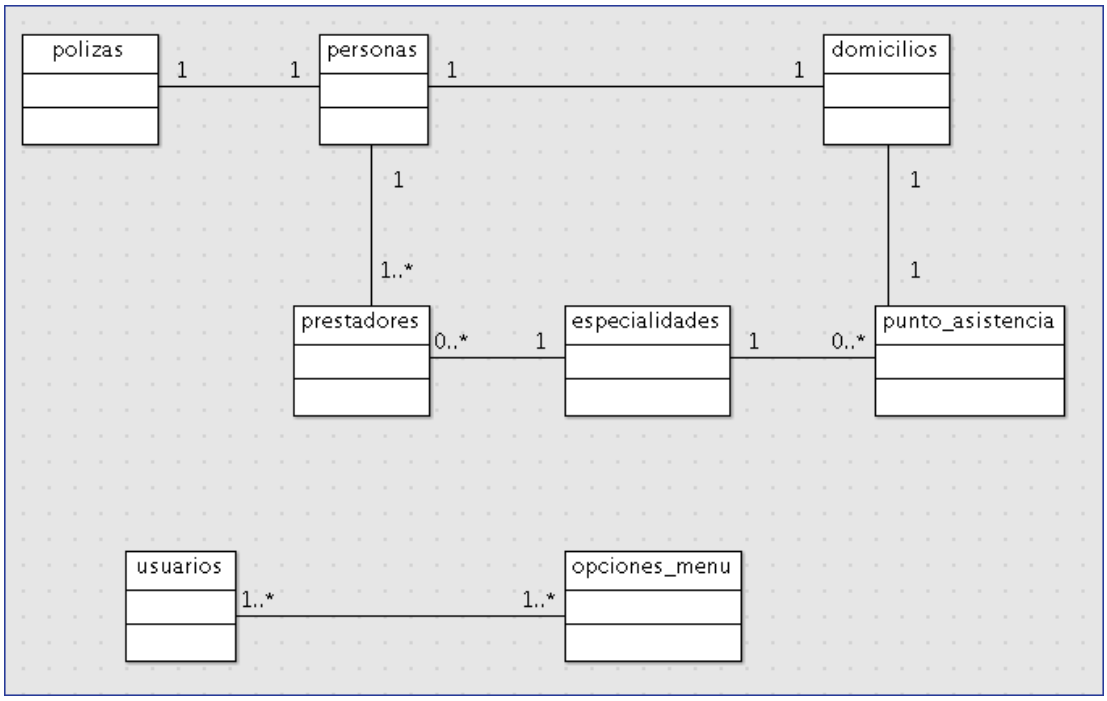


Ilustración 3: Diagrama del modelo conceptual

3 Diseño

3.1 Arquitectura de Software y de Hardware

El esquema propuesto para el diseño y explotación de este proyecto se puede ver a continuación:

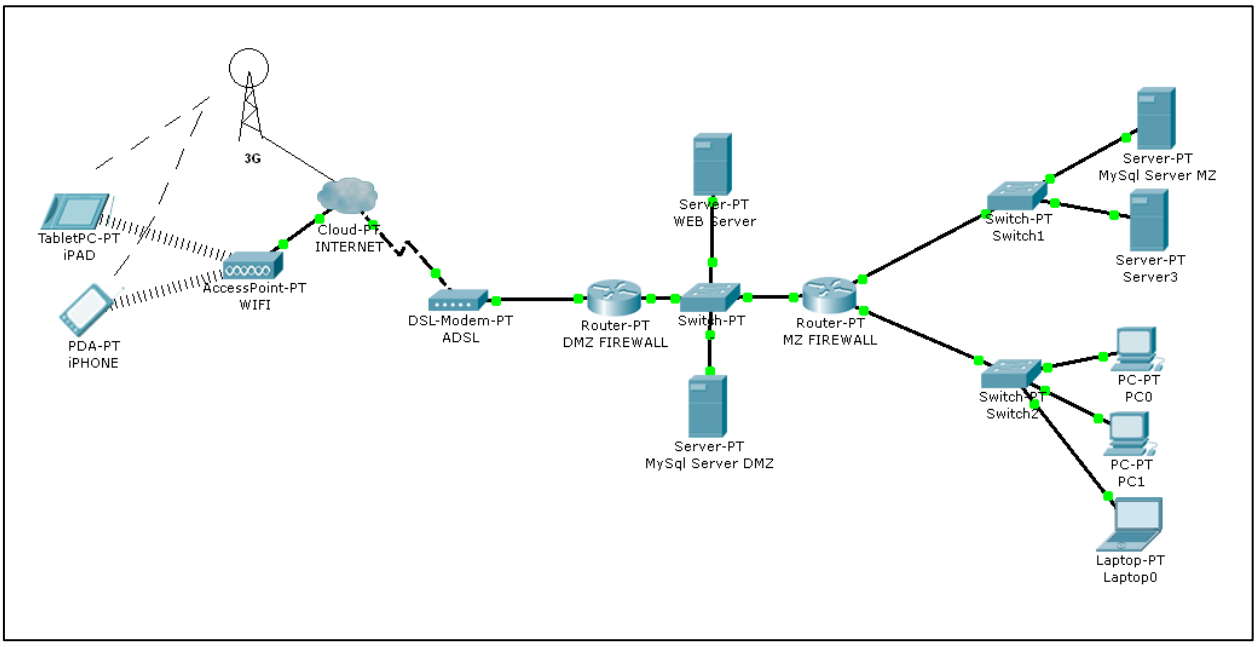
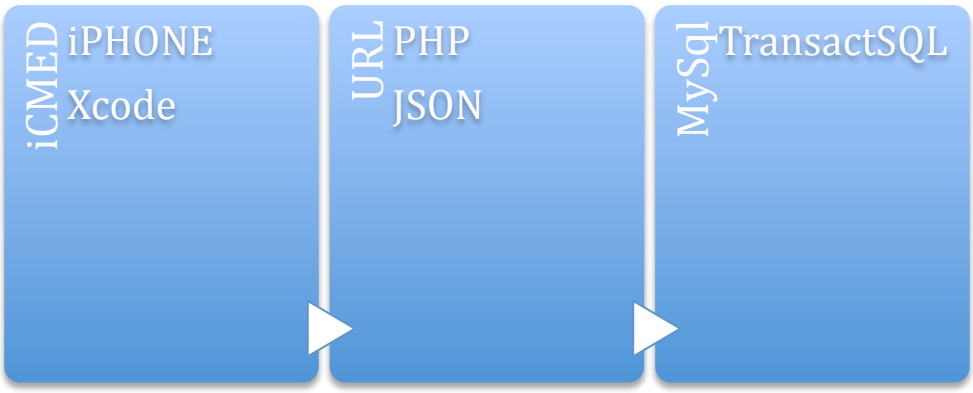


Ilustración 4: Diagrama del esquema de red

En el apartado servidor disponemos de un servidor de base de datos MySQL 5.5.30 que es accedido por PHP v5.3.15 instalados en un servidor Web Apache 2.2.22.

Se utiliza un servicio Web (JSON - JavaScript Object Notation) que sirve los datos al dispositivo móvil con iCMED para evitar el acceso directo a la base de datos por parte de la aplicación.

En cuanto a la parte cliente necesitaremos un iPhone con la versión 6.x del sistema operativo iOS.



Para el desarrollo se ha dispuesto todo en un entorno Mac utilizando el servidor Web Apache que por defecto trae el sistema operativo Lion, activando PHP; adicionalmente se ha instalado MySQL como sistema de gestión de base de datos.

El desarrollo se ha realizado con Xcode 4.6.2 y la pruebas de la aplicación sobre el dispositivo se han realizado con el simulador iOS 6.0.

En cuanto a la disposición segura de los servidores se puede apreciar en el siguiente esquema como toda la parte corporativa o MZ queda detrás de un cortafuegos y los servidores que deben ser accedidos por la aplicación iCMED se encuentran en la DMZ protegidos por otro cortafuegos, de manera que en caso de que algún equipo de la DMZ fuese accedido indebidamente, el resto de equipos estarían, en principio protegidos.

3.1.1 Decisiones tecnológicas

Se ha elegido la plataforma tecnológica basada en MAMP (Mac, Apache, MySQL y PHP) ya que se dispone de ella, y es fácilmente transportable a Linux o incluso Windows, ya que existe para estos sistemas operativos.

Como servicio Web se utiliza JSON por sencillez de uso y porque Xcode ya lo lleva implementado en sus librerías; además, PHP también tiene funciones para generar datos JSON.

En cuanto a la base de datos se pretende que sea una réplica en tiempo real de la base de datos corporativa únicamente formada por las tablas necesarias para el correcto funcionamiento de la aplicación. Esta estructura de base de datos se especifica más adelante en este documento.

3.1.2 Usabilidad

Los controles utilizados en la aplicación están específicamente diseñados para el uso sencillo de la misma en el iPhone.

Se ha evitado en la medida de lo posible el tener que teclear datos a través del teclado en pantalla del dispositivo móvil, pero, por ejemplo, en la entrada a la aplicación si se tienen que teclear el usuario y la contraseña.

En cuanto al usuario, la aplicación recordará el último usuario validado correctamente para evitar tener que teclearlo de nuevo la siguiente ocasión, en el caso de que sea el mismo, y de esta manera sólo tendrá que teclear la clave.

Aunque no se prevé un uso intensivo de la aplicación, ni por parte de los asegurados (que la usarán puntualmente), ni por parte de los coordinadores médicos de la compañía aseguradora que estén de guardia (ya que el nivel de llamadas fuera de horario de oficina es realmente bajo), se contempla el ahorro de batería realizando un geoposicionamiento por cambio significativo que utiliza triangulación en base a datos de las antenas de telefonía y no utiliza el GPS del dispositivo.

En cuanto al acceso a datos, también se ha tenido en cuenta economizar los accesos a Internet y la cantidad de datos descargados, dado que a fecha actual las tarifas de acceso a datos de los dispositivos son limitadas en función de la cantidad de información descargada. Para ello, en el caso concreto de búsqueda de ruta entre la posición actual y el domicilio del servicio médico sólo se realiza una vez para un domicilio de prestación el acceso a la descarga de datos de la ruta en coche y el acceso a la descarga de datos de la ruta a pie.

3.1.3 Dispositivos que soportan la aplicación

En este proyecto se desarrolla la aplicación de forma específica para iPhone con iOS 6.1 o superior.

Aunque la misma ya sería funcional para iPad, se podría realizar una adaptación muy sencilla para dar soporte a este dispositivo, de forma que se aproveche todo el potencial que el mismo ofrece, sobretodo en cuanto al tamaño de la pantalla.

Asimismo, también se podría contemplar realizar el desarrollo para Android que es otro de los sistemas más utilizados en la actualidad.

3.2 Diseño de la base de datos

Por cuestiones de seguridad, la base de datos es réplica de la base de datos corporativa, pero solo contiene datos necesarios para la aplicación móvil iCMED.

El modelo de base de datos necesario para la consulta del cuadro médico, la validación de usuarios y la consulta de pólizas, se muestra en la siguiente ilustración:

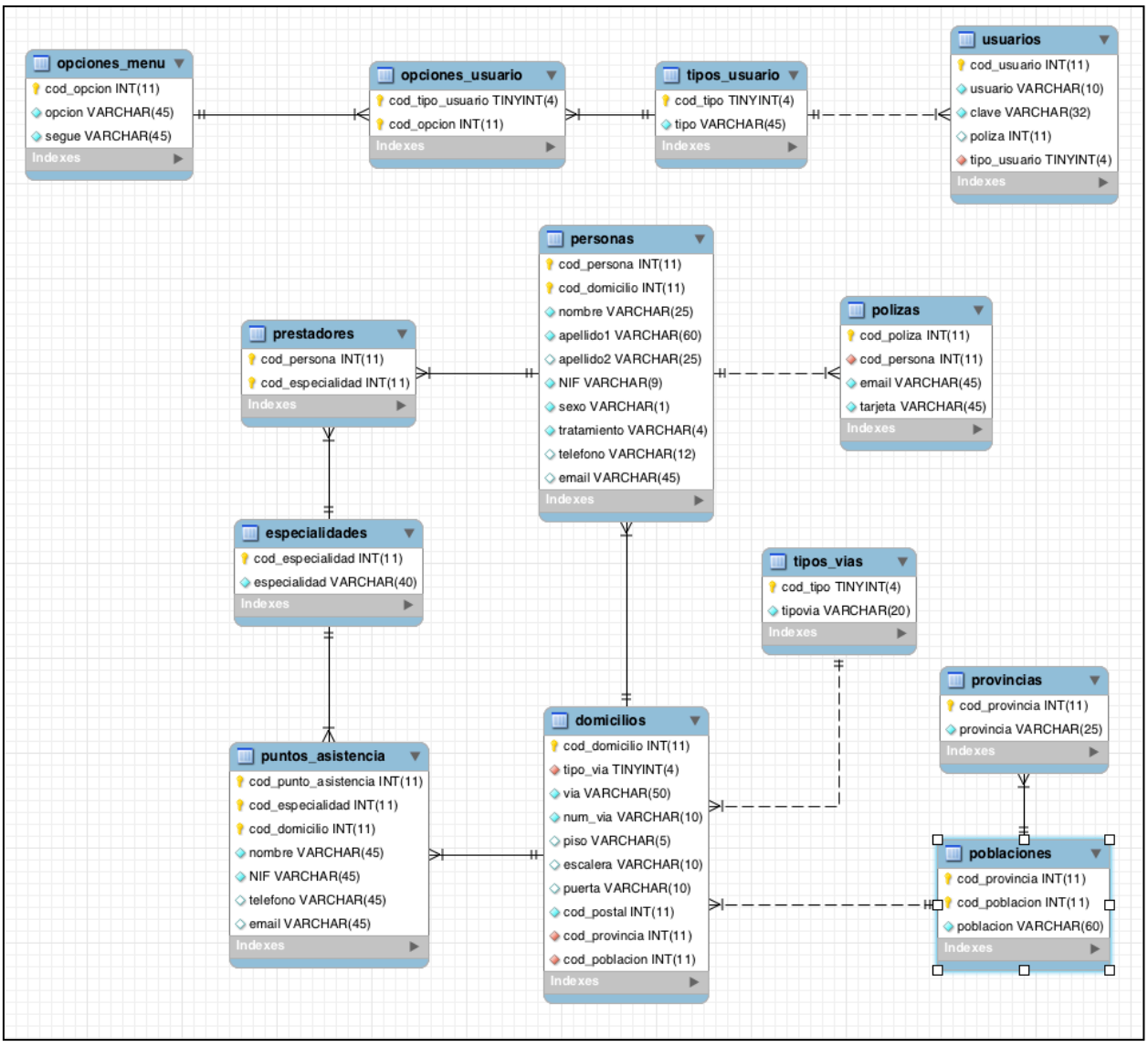


Ilustración 5: Modelo de datos

3.2.1 Tabla personas

Esta tabla contiene los datos de las personas ya sean asegurados (tabla pólizas) o médicos (tabla prestadores).

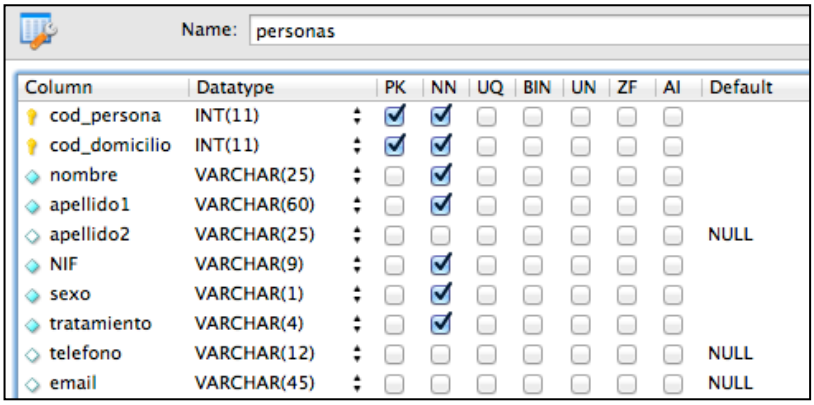


Ilustración 6: Tabla personas

Podemos observar que el campo apellido2 no es obligatorio ya que algunas personas extranjeras no usan segundo apellido.

Se encuentra relacionada con la tabla domicilios que tiene los datos del domicilio, tanto de los prestadores médicos y hospitales y clínicas, como asegurados.

3.2.2 Tabla prestadores

Esta tabla contiene la relación de los médicos que pertenecen al cuadro médico de la compañía y por tanto pueden recibir asegurados de la misma.

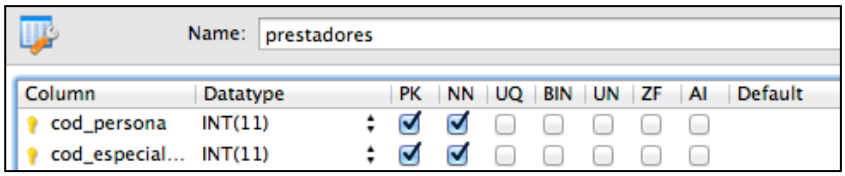


Ilustración 7: Tabla prestadores

Se encuentra relacionada con la tabla personas donde se tienen los datos básicos del prestador médico y la tabla de especialidades, donde se tienen las especialidades médicas.

3.2.3 Tabla domicilios

Esta tabla contiene los domicilios de las personas, tanto si son asegurados o médicos.

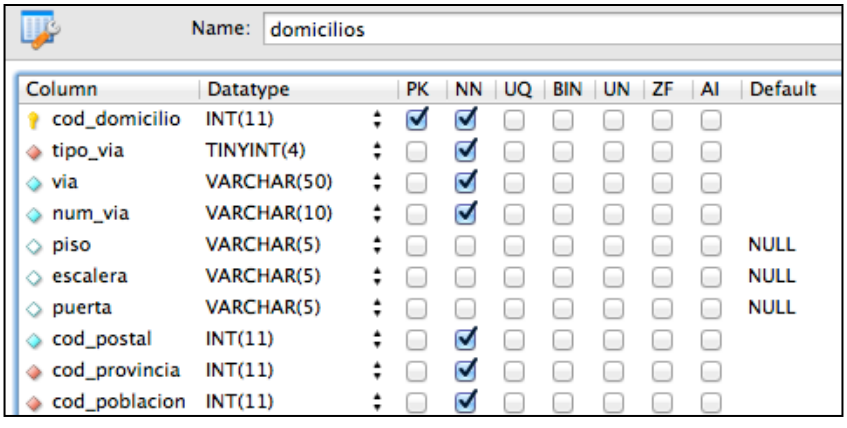


Ilustración 8: Tabla domicilios

Se encuentra relacionada con la tabla personas y con la tabla puntos_asistencia por el campo cod_domicilio.

3.2.4 Tabla especialidades

Esta tabla es donde tenemos almacenadas las especialidades médicas.

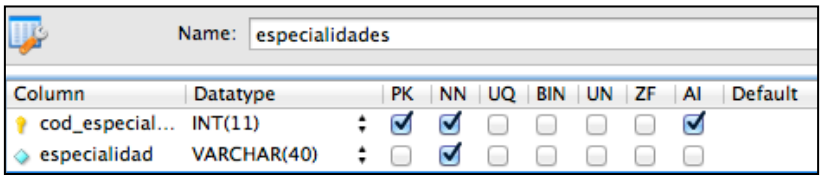


Ilustración 9: Tabla especialidades

Está relacionada con la tabla prestadores y puntos_asistencia a través del campo cod_especialidad.

3.2.5 Tabla provincias

En esta tabla tenemos todas las provincias españolas.

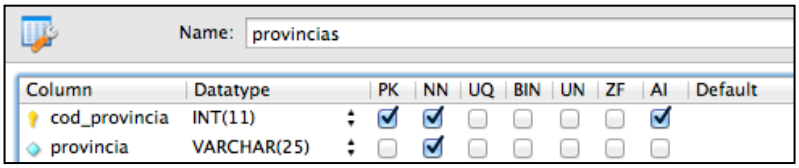


Ilustración 10: Tabla provincias

Se encuentra relacionada con la tabla poblaciones por el campo cod_provincia.

3.2.6 Tabla poblaciones

En esta tabla se almacenan las poblaciones del territorio español.

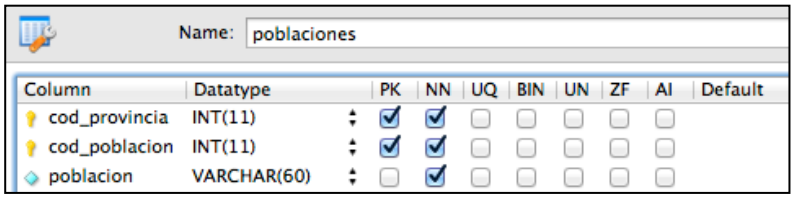


Ilustración 11: Tabla poblaciones

Esta relacionada con la tabla provincias por el campo cod_provincia; además está relacionada con la tabla domicilios por los campos cod_provincia y cod_poblacion.

3.2.7 Tabla tipos_vias

En esta tabla se encuentran referenciados los tipos de vías (paseo, calle, avenida, etc.) para los domicilios.

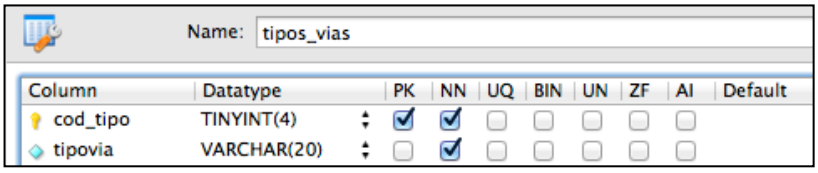


Ilustración 12: Tabla tipos_vias

Está relacionada con la tabla domicilios por el campo cod_tipo y el campo tipo_via de la tabla domicilios.

3.2.8 Tabla usuarios

En esta tabla se encuentran los datos de acceso a la aplicación de los usuarios de la aplicación iCMED; estos usuarios pueden ser los propios asegurados o los coordinadores médicos de la compañía GESESA que dan servicio a los clientes de la misma.

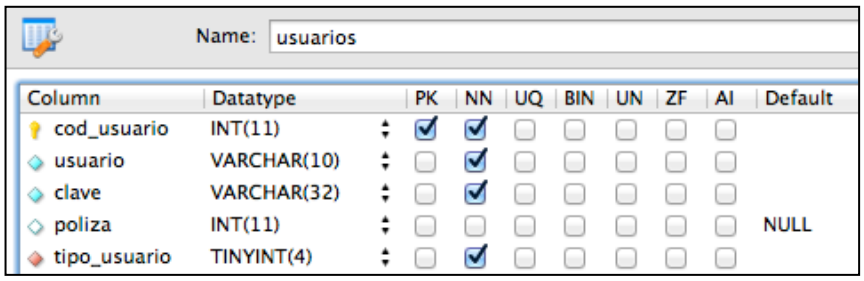


Ilustración 13: Tabla usuarios

Está relacionada con la tabla tipos_usuarios por el campo cod_tipo.

3.2.9 Tabla tipos_usuarios

En esta tabla se almacenan los tipos de usuarios con el objetivo de conocer a que funcionalidades tiene acceso un usuario concreto.

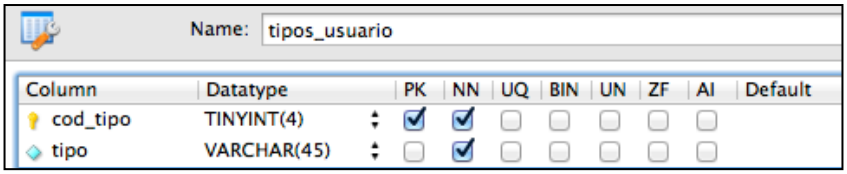


Ilustración 14: Tabla tipos_usuario

Se encuentra relacionada con la tabla de usuarios por el campo cod_tipo.

3.2.10 Tabla opciones_usuario

Esta tabla relaciona los tipos de usuario existentes en el sistema con las opciones de la aplicación iCMED.

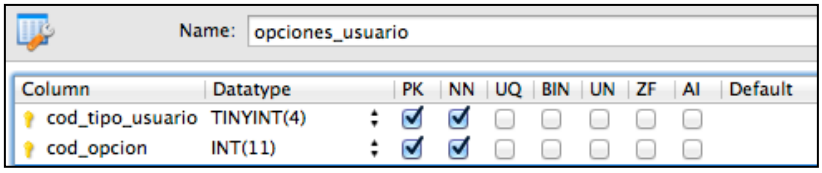


Ilustración 15: Tabla opciones_usuario

Está relacionada con tipos_usuario con el campo cod_tipo_usuario y con la tabla opciones_menu por el campo cod_opcion.

3.2.11 Tabla opciones_menu

Esta tabla contiene las diferentes funcionalidades desarrolladas en la aplicación iCMED; en un futuro se podrán incluir aquí todas las funcionalidades que se vayan desarrollando para dar servicio a los asegurados de GESESA.

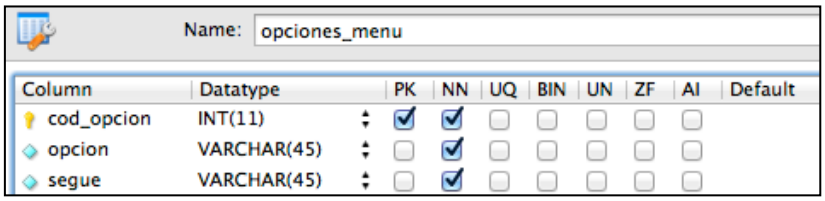


Ilustración 16: Tabla opciones_menu

El campo segue (leído como seg way) contiene el nombre del controlador de la aplicación que contiene la funcionalidad desarrollada en esta opción.

Esta tabla está relacionada con la tabla opciones_usuario por el campo cod_opcion.

3.2.12 Tabla polizas

En esta tabla almacenamos las pólizas que los clientes de la compañía aseguradora tiene contratados.

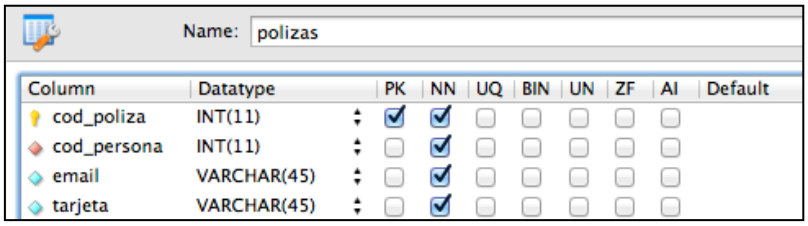


Ilustración 17: Tabla pólizas

Esta relacionada con la tabla personas por el campo cod_persona y con la tabla domicilios por el campo cod_domicilio.

3.2.13 Tabla puntos_asistencia

En esta tabla almacenamos los hospitales, clínicas y centros médicos que están incluidos en el cuadro médicos de la compañía aseguradora.

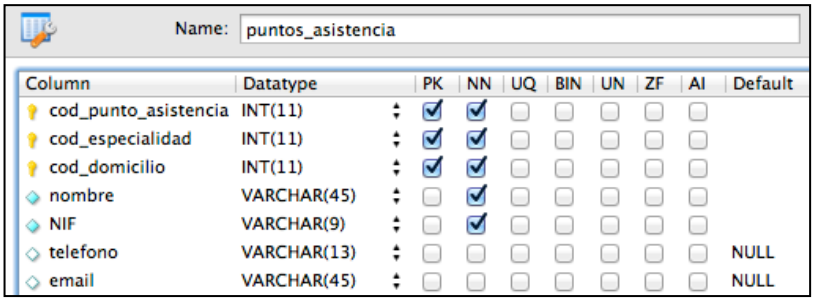


Ilustración 18: Tabla puntos_asistencia

Está relacionada con la tabla especialidades por el campo cod_especialidad. También se encuentra relacionada con la tabla domicilios por el campo cod_domicilio.

3.3 Diseño de las consultas

La aplicación necesita acceder a la base de datos en diferentes ocasiones, para ello se realizan consultas a la misma.

Se ha desarrollado un procedimiento almacenado único que dependiendo del primer parámetro que se le pase a dicho procedimiento devolverá datos de una u otra consulta.

Con esto evitamos tener las consultas en PHP, ya que este solamente llamará a un procedimiento almacenado y solo se varían los parámetros que se le pasan a dicho procedimiento almacenado (hasta 6 distintos: opc, var1, var2, var3, var4 y var5). Además, las consultas se ejecutan mucho más rápido desde un procedimiento almacenado que si se realizan directamente desde código PHP.

A continuación se explican las consultas utilizadas por la aplicación iCMED y que se ejecutarán en el procedimiento almacenado (sp_iCMED) dependiendo de la variable opc.

3.3.1 Provincias con cuadro médico

Esta consulta se realiza para averiguar las provincias donde la compañía aseguradora tiene cuadro médico.

Se utiliza una UNION de 2 SELECT para obtener las provincias donde hay prestadores (con la primera SELECT) y las provincias donde hay puntos de asistencia (con la segunda SELECT).

La consulta desarrollada es como sigue:

```
(SELECT
   provincias.cod_provincia, provincias.provincia
FROM
   provincias, prestadores, personas, domicilios
WHERE
   provincias.cod_provincia = domicilios.cod_provincia AND
   domicilios.cod_domicilio = personas.cod_domicilio AND
   personas.cod_persona = prestadores.cod_persona
GROUP BY
   provincias.cod_provincia, provincias.provincia
)

UNION DISTINCT

(SELECT
   provincias.cod_provincia, provincias.provincia
FROM
   provincias, puntos_asistencia, domicilios
WHERE
   provincias.cod_provincia = domicilios.cod_provincia AND
   domicilios.cod_domicilio = puntos_asistencia.cod_domicilio
GROUP BY
   provincias.cod_provincia, provincias.provincia
)

ORDER BY
   cod_provincia, provincia;
```

3.3.2 Poblaciones con cuadro médico de una provincia elegida

Esta consulta se realiza para averiguar aquellas poblaciones donde la compañía tiene servicio médico en referencia a una provincia previamente elegida por el usuario de la aplicación.

Al igual que en la consulta anterior también se utiliza un UNION de 2 SELECT, pero en este caso se utiliza uno de los parámetros (var1) del procedimiento almacenado.

La consulta desarrollada es la siguiente:

```
(SELECT
   poblaciones.cod_poblacion, poblaciones.poblacion
FROM
   poblaciones, provincias, prestadores, personas, domicilios
WHERE
   poblaciones.cod_provincia = provincias.cod_provincia AND
   domicilios.cod_provincia = provincias.cod_provincia AND
   domicilios.cod_poblacion = poblaciones.cod_poblacion AND
   personas.cod_persona = prestadores.cod_persona AND
   personas.cod_domicilio = domicilios.cod_domicilio AND
   provincias.cod_provincia = var1
GROUP BY
   poblaciones.cod_poblacion, poblaciones.poblacion
)

UNION DISTINCT

(SELECT
   poblaciones.cod_poblacion, poblaciones.poblacion
FROM
   poblaciones, provincias, puntos_asistencia, domicilios
WHERE
   poblaciones.cod_provincia = provincias.cod_provincia AND
   domicilios.cod_provincia = provincias.cod_provincia AND
   domicilios.cod_poblacion = poblaciones.cod_poblacion AND
   puntos_asistencia.cod_domicilio = domicilios.cod_domicilio AND
   provincias.cod_provincia = var1
GROUP BY
   poblaciones.cod_poblacion, poblaciones.poblacion
)

ORDER BY
   poblacion;
```

3.3.3 Especialidades con servicio de una poblacion elegida

Esta consulta se realiza para averiguar que especialidades médicas existen en la población elegida previamente por el usuario de la aplicación.

también es una UNION de 2 SELECT y se utilizan los parámetros var1 y var2 pasados al procedimiento almacenado que indican la provincia y población elegidas.

La consulta desarrollada es la siguiente:

```
(SELECT
   especialidades.cod_especialidad, especialidad
FROM
   especialidades, prestadores, personas,
   domicilios, provincias, poblaciones
WHERE
   especialidades.cod_especialidad = prestadores.cod_especialidad AND
   domicilios.cod_domicilio = personas.cod_domicilio AND
   personas.cod_persona = prestadores.cod_persona AND
   domicilios.cod_provincia = provincias.cod_provincia AND
   domicilios.cod_poblacion = poblaciones.cod_poblacion AND
   provincias.cod_provincia = var1 AND
   poblaciones.cod_poblacion = var2 AND
   provincias.cod_provincia = poblaciones.cod_provincia
```

```
GROUP BY
   especialidades.cod_especialidad, especialidad
)

UNION DISTINCT

(SELECT
   especialidades.cod_especialidad, especialidad
FROM
   especialidades, puntos_asistencia,
   domicilios, provincias, poblaciones
WHERE
   especialidades.cod_especialidad = puntos_asistencia.cod_especialidad AND
   domicilios.cod_domicilio = puntos_asistencia.cod_domicilio AND
   domicilios.cod_provincia = provincias.cod_provincia AND
   domicilios.cod_poblacion = poblaciones.cod_poblacion AND
   provincias.cod_provincia = var1 AND
   poblaciones.cod_poblacion = var2 AND
   provincias.cod_provincia = poblaciones.cod_provincia
GROUP BY
   especialidades.cod_especialidad, especialidad
)

ORDER BY
   cod_especialidad, especialidad;
```

3.3.4 Asegurados

Esta consulta se realiza preparando la sentencia SQL a ejecutar en función de los datos facilitados por el usuario de la aplicación para la búsqueda del asegurado.

Estos datos del asegurado pueden ser el nombre, el apellido primero, el apellido segundo, el NIF y/o el número de tarjeta de asegurado en la compañía, y se pasan al procedimiento almacenado en var1, var2, var3, var4 y var5 respectivamente.

El código desarrollado para esta funcionalidad es el siguiente:

```
SET @S = '
   SELECT
          nombre, apellido1, apellido2, NIF, cod_poliza, tarjeta,
          CONCAT(tipovia, '' '', via, '', '', num_via) AS domicilio,
          piso, escalera, puerta, cod_postal,
          provincias.provincia, poblaciones.poblacion
   FROM
          polizas, personas, domicilios,
          tipos_vias, provincias, poblaciones
   WHERE
          polizas.cod_persona = personas.cod_persona AND
          personas.cod_domicilio = domicilios.cod_domicilio AND
          domicilios.tipo_via = tipos_vias.cod_tipo AND
          domicilios.cod_provincia = provincias.cod_provincia AND
          domicilios.cod_provincia = poblaciones.cod_provincia AND
          domicilios.cod_poblacion = poblaciones.cod_poblacion';

IF var1 <> '' THEN
   SET @S = CONCAT (@S, ' AND personas.nombre = \'', var1, '\'');
END IF;

IF var2 <> '' THEN
```

```
   SET @S = CONCAT (@S, ' AND personas.apellido1 = \'', var2, '\'');
END IF;

IF var3 <> '' THEN
   SET @S = CONCAT (@S, ' AND personas.apellido2 = \'', var3, '\'');
END IF;

IF var4 <> '' THEN
   SET @S = CONCAT (@S, ' AND personas.NIF = \'', var4, '\'');
END IF;

IF var5 <> '' THEN
   SET @S = CONCAT (@S, ' AND polizas.tarjeta = \'', var5, '\'');
END IF;

PREPARE SSQL FROM @S;

EXECUTE SSQL;

DEALLOCATE PREPARE SSQL;
```

3.3.5 Prestadores médicos y puntos de asistencia

Esta consulta se realiza para averiguar qué servicios médicos (doctores y puntos de asistencia) existen en una localidad para una especialidad médica concreta.

Como en ocasiones anteriores el la UNION de dos consultas de selección:

```
(SELECT
   CONCAT (personas.tratamiento, ' ', personas.nombre, ' ',
   personas.apellido1, ' ', personas.apellido2) AS nombre_completo,
   tipos_vias.tipovia, domicilios.via, domicilios.num_via,
   domicilios.cod_postal, poblaciones.poblacion, provincias.provincia,
   personas.telefono, especialidades.especialidad,
   CONCAT (tipos_vias.tipovia,  ' ', domicilios.via, ', ',
   domicilios.num_via,   '   ',   poblaciones.poblacion,   '   ESPAÑA')   AS
domicilio_gps
FROM
   especialidades, prestadores, personas,
   domicilios, provincias, poblaciones, tipos_vias
WHERE
   especialidades.cod_especialidad = prestadores.cod_especialidad AND
   domicilios.cod_domicilio = personas.cod_domicilio AND
   personas.cod_persona = prestadores.cod_persona AND
   domicilios.cod_provincia = provincias.cod_provincia AND
   domicilios.cod_poblacion = poblaciones.cod_poblacion AND
   provincias.cod_provincia = poblaciones.cod_provincia AND
   domicilios.tipo_via = tipos_vias.cod_tipo AND
   provincias.cod_provincia = var1 AND
   poblaciones.cod_poblacion = var2 AND
   especialidades.cod_especialidad = var3
)

UNION DISTINCT

(SELECT
   puntos_asistencia.nombre AS nombre_completo,
   tipos_vias.tipovia, domicilios.via, domicilios.num_via,
   domicilios.cod_postal, poblaciones.poblacion, provincias.provincia,
```

```
   puntos_asistencia.telefono, especialidades.especialidad,
   CONCAT (tipos_vias.tipovia,  ' ', domicilios.via, ', ',
      domicilios.num_via,  '  ',  poblaciones.poblacion,  '  ESPAÑA')  AS
domicilio_gps
FROM
   especialidades, puntos_asistencia,
   domicilios, provincias, poblaciones, tipos_vias
WHERE
   especialidades.cod_especialidad = puntos_asistencia.cod_especialidad AND
   domicilios.cod_domicilio = puntos_asistencia.cod_domicilio AND
   domicilios.cod_provincia = provincias.cod_provincia AND
   domicilios.cod_poblacion = poblaciones.cod_poblacion AND
   provincias.cod_provincia = poblaciones.cod_provincia AND
   domicilios.tipo_via = tipos_vias.cod_tipo AND
   provincias.cod_provincia = var1 AND
   poblaciones.cod_poblacion = var2 AND
   especialidades.cod_especialidad = var3
)

ORDER BY
   nombre_completo;
```

3.4 Prototipo de aplicación

3.4.1 Pantalla general del simulador iOS

En esta pantalla podemos observar el icono de la aplicación iCMED antes de su ejecución:

Ilustración 19: Vista del icono de la aplicación

3.4.2 Pantalla de inicio

La aplicación comienza con una pantalla de presentación de la misma, donde se observa el nombre de la aplicación, su imagen asociada y en la parte baja información sobre el TFC y su autor. También podemos observar un botón ('Pulse para continuar') para poder pasar a la siguiente pantalla de la aplicación:

Ilustración 20: Pantalla inicio de sesión

3.4.3 Pantalla de funcionalidades de la aplicación

Esta pantalla muestra las diferentes funcionalidades de la aplicación, que variarán dependiendo del tipo de usuario que la esté utilizando.

Ilustración 21: Pantalla de Funcionalidades de la aplicación

3.4.4 Pantalla de búsqueda de prestadores médicos

Si pulsamos en el mencionado botón, pasaremos a la siguiente pantalla donde disponemos de las opciones de búsqueda.

Ilustración 22: Pantalla de búsqueda de servicios médicos

Se ha decidido realizar esta búsqueda eligiendo primero la provincia donde necesitamos el servicio médico, para seguidamente mostrarse las poblaciones de la provincia seleccionada en las que existe cuadro médico; una vez elegida la población se podrá elegir entre las especialidades médicas existentes en dicha población.

Hay que señalar que en cualquier momento y en cualquier pantalla, se puede abandonar la sesión actual pulsando en el botón superior derecho (marcado con una X); también se puede volver a la pantalla anterior pulsando el botón superior izquierdo (marcado con una flecha hacia la izquierda) desde ciertas pantallas.

3.4.5 Pantalla de resultados del cuadro médico

Una vez se elige la especialidad entre las disponibles en la población elegida, se puede pulsar el botón 'Buscar Servicio Médico' y así se pasa a la siguiente pantalla donde se reflejarán los servicios médicos disponibles para la provincia, población y especialidad seleccionadas.

Ilustración 23: Pantalla de resultados de la búsqueda de prestadores médicos.

En este caso vemos que hemos seleccionado la especialidad de 'ALERGOLOGÍA' en Madrid capital y la consulta a la base de datos nos devuelve 3 registros correspondientes a 2 alergólogos de Madrid y un hospital con servicio de alergología.

3.4.6 Pantalla de geoposicionamiento del prestador médico

Si seleccionamos uno de los médicos en la pantalla anterior de resultados, el mismo nos aparecerá localizado en un mapa y nos guiará desde nuestra posición actual, tal y como podemos observar en la siguiente ilustración:

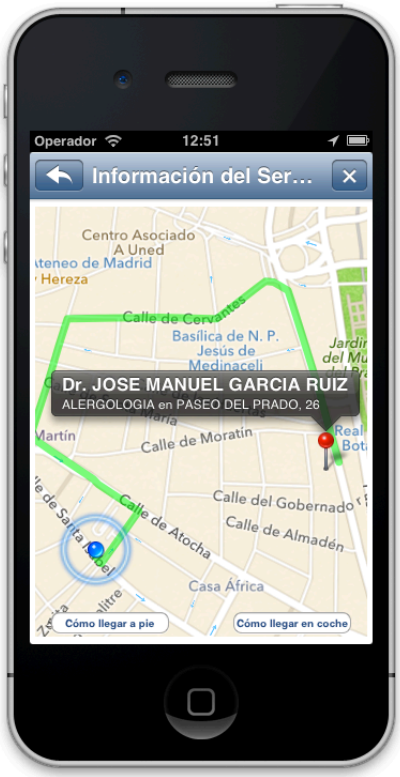


Ilustración 24: Pantalla de geolocalización de prestador médico

Podemos observar que la ruta se puede calcular de dos formas distintas, dependiendo de si queremos ir andando o queremos ir en coche.

También podemos desplegar la información del punto de asistencia o médico elegido.

3.4.7 Pantalla de cambio de contraseña de usuario

Si seleccionamos la opción de cambio de contraseña en la pantalla inicial de opciones de la aplicación, nos aparece la siguiente pantalla donde se nos permitirá cambiar la clave de nuestro usuario:

Ilustración 25: Pantalla de cambio de contraseña de usuario

3.4.8 Pantalla de búsqueda de asegurados

Si seleccionamos la búsqueda de asegurados para ver si la persona que nos llama tiene contratada una póliza con la compañía y así poder darle servicio, nos aparece la pantalla de búsqueda de asegurados:

Ilustración 26: Pantalla de búsqueda de asegurados

3.4.9 Flujo de pantallas (Storyboard)

El flujo de pantallas de forma global se puede ver en la siguiente figura:

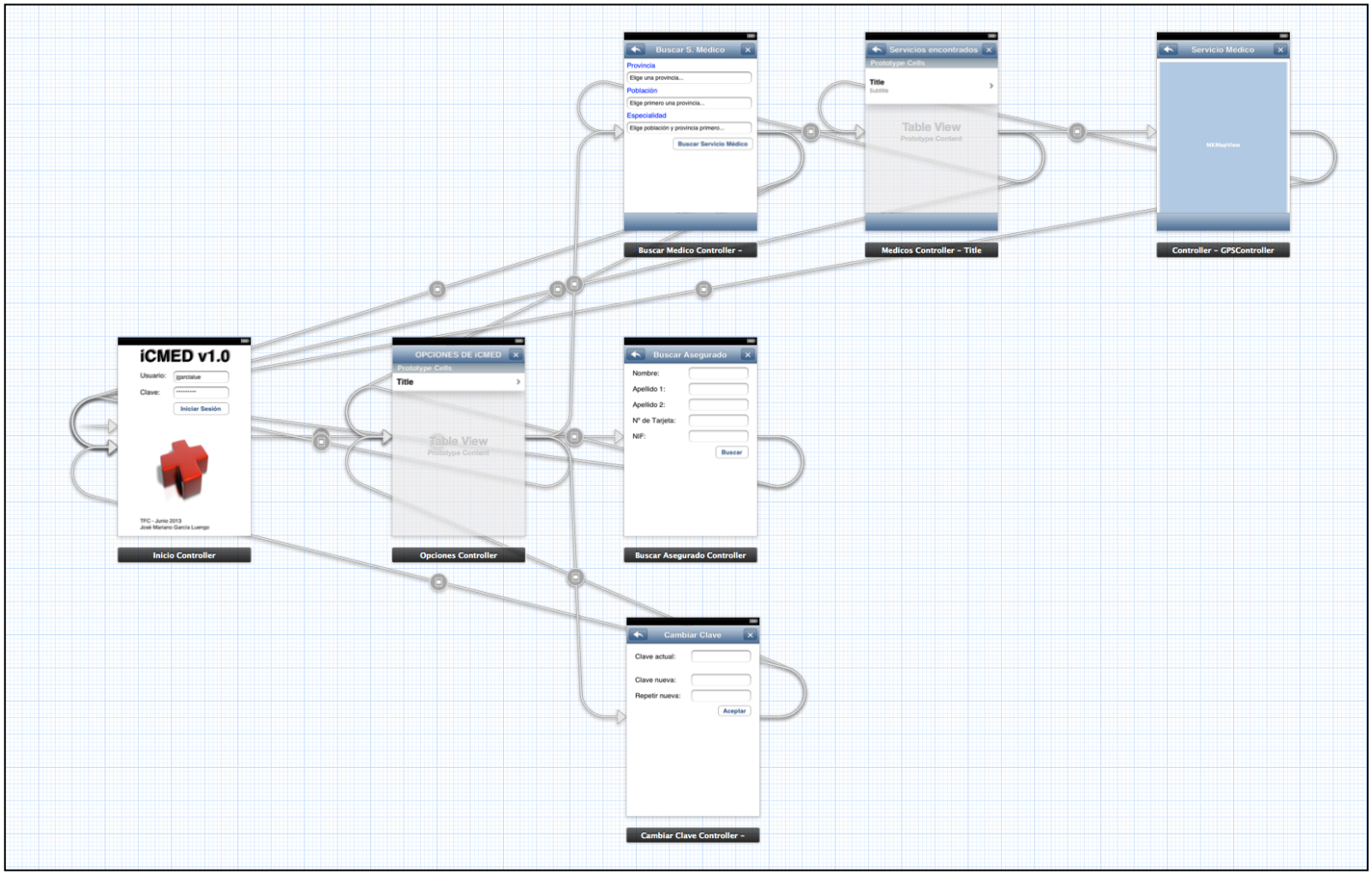


Ilustración 27: Flujo de pantallas de la aplicación (Storyboard)

A continuación podemos ver detallado diferentes partes del storyboard que permiten apreciar mejor el flujo de pantallas.

En la siguiente captura podemos ver e flujo de la parte de búsqueda en el cuadro médico:

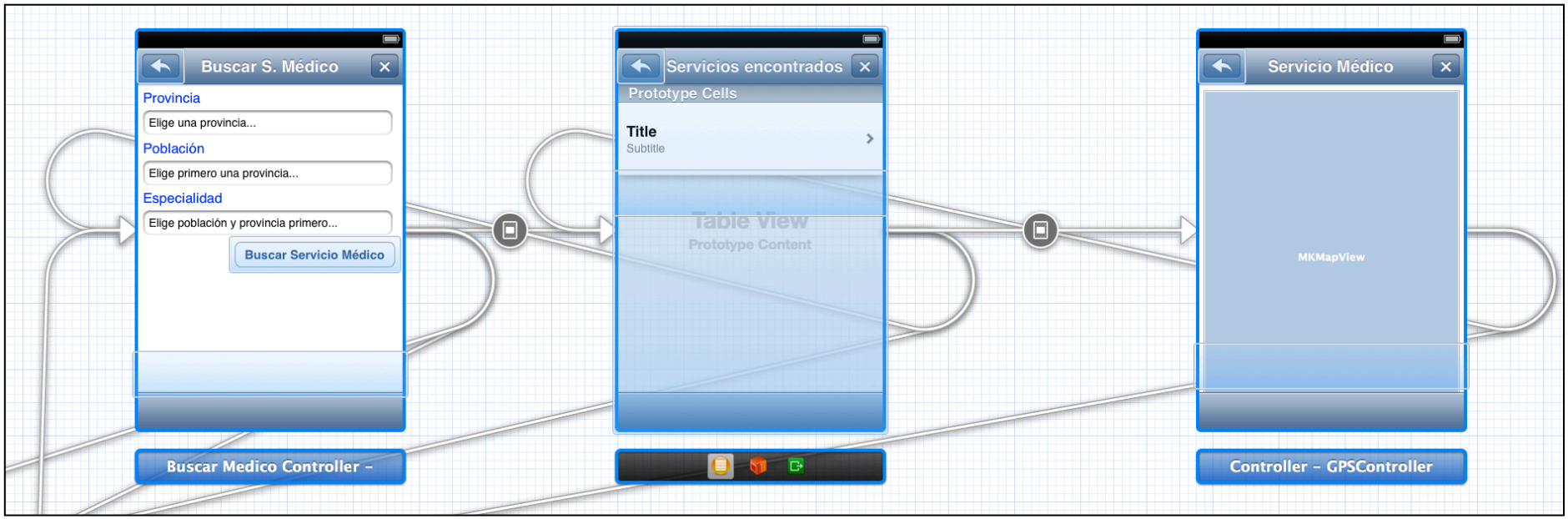


Ilustración 28: Flujo de pantallas de búsqueda en el cuadro médico

En la siguiente captura se puede observar detallado desde la pantalla de inicio de sesión, pasando por la ventana de opciones de la aplicación, para terminar con la pantalla de búsqueda de asegurado:

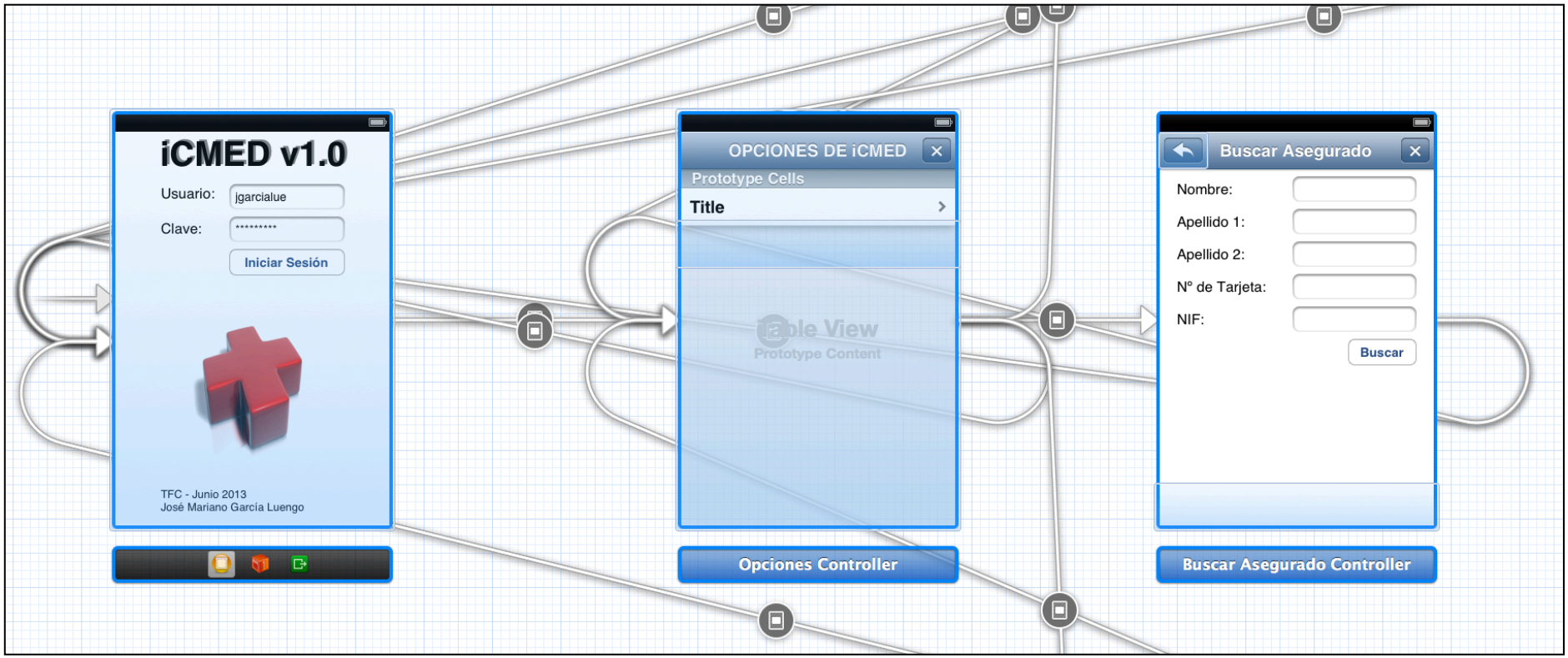


Ilustración 29: Flujo pantalla inicio de sesión, opciones de aplicación y búsqueda de asegurado

A continuación se puede observar el flujo de la aplicación con las tres funcionalidades de la aplicación ubicadas a la derecha en la captura de pantalla:

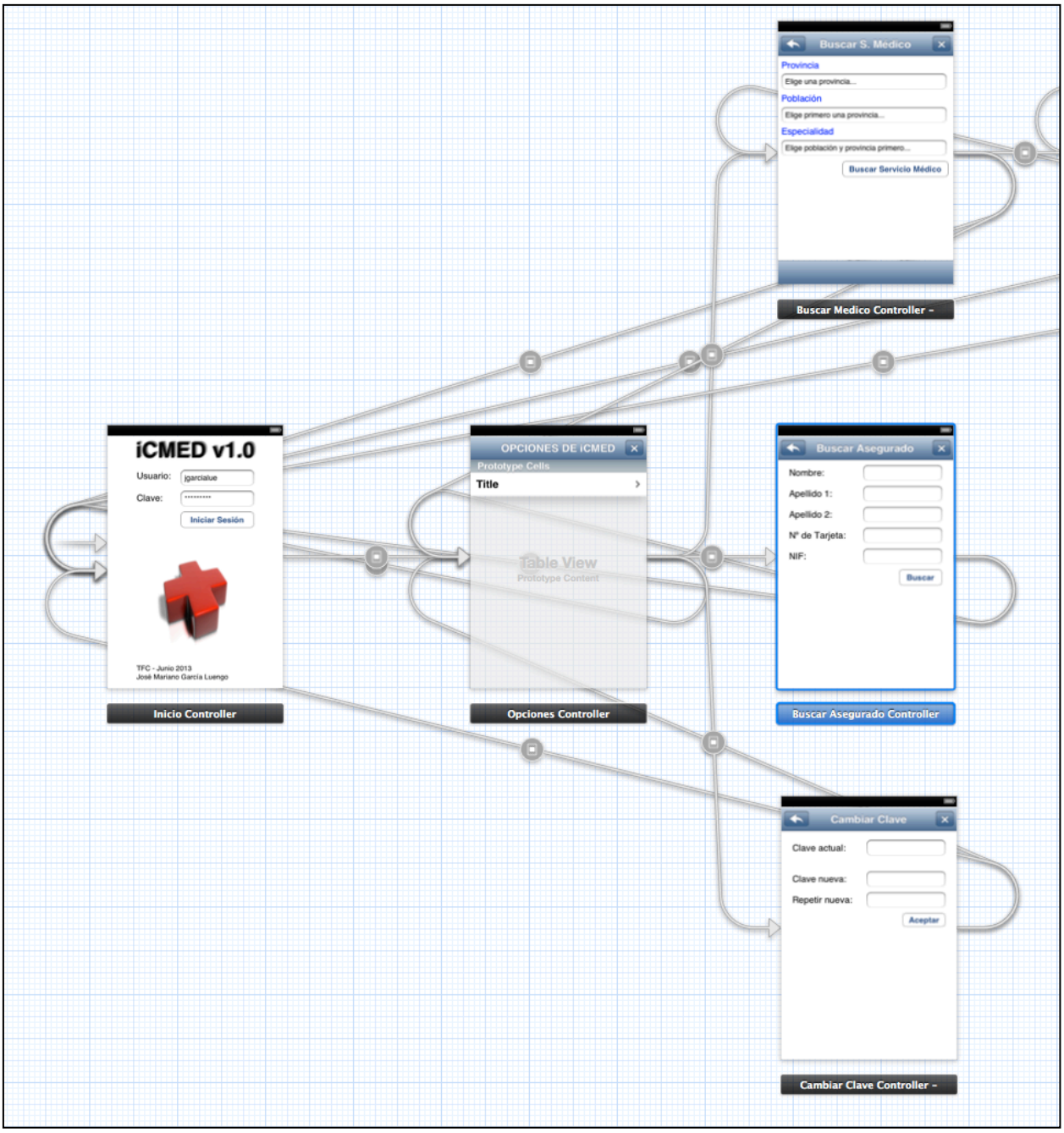


Ilustración 30: Flujo de pantallas de opciones de aplicación

3.5 Implementación

En este apartado se comentarán los aspectos más relevantes del código desarrollado para la aplicación iCMED. Incluye, tanto el código php y procedimiento almacenado, como descripción de parte del código de las diferentes clases de la aplicación en Xcode.

3.5.1 Servicios Web

Se ha desarrollado código php que accede a un único procedimiento almacenado. Este código php devuelve los datos parseados con JSON.

3.5.1.1 Módulos PHP

Se han desarrollado 2 módulos PHP que dan servicio Web y devuelven datos necesarios para la aplicación. Estos módulos son accedidos desde la aplicación Xcode.

El primer módulo (global.php) se usa para definir las variables de conexión al servidor MySQL y se requiere en el segundo módulo PHP. El fichero es muy sencillo pero nos permite acceder de forma sencilla y centralizada a estos parámetros en la configuración inicial o en caso de que haya que cambiarlos en un futuro. Su contenido es como sigue:

```
<?php
   // Variables para conectar con el servidor MySQL
   $user        =      "iCMED_usr";
   $pwd         =      "iCMED";
   $server      =      "127.0.0.1";
   $db          =      "iCMED_db";

   function safe($con, $cad)
   {
          $cad = mysqli_real_escape_string($con, $cad);
          $cad = str_replace (array("< ",">","[","]","*","^"), "" , $cad);

          return $cad;
   }
?>
```

Podemos observar la función safe que permite modificar las entradas que llegan al servicio Web eliminando las cadenas de escape Html para evitar inyecciones de código sql.

En el caso del segundo PHP (consulta_Mysql.php) tenemos las llamadas a las diferentes opciones de la aplicación que requieren devolución de datos desde la base de datos, o incluso una actualización de los mismos, como sucede en la opción de cambio de contraseña de usuario.

El contenido es el siguiente:

```
<?php
   //Variables de conexión
   require 'global.php';

   // Las acciones posibles son:
   //  usu      -> devuelve los datos de usuario (código, clave y tipo)
   //  getpwd   -> devuelve la clave de un usuario.
   //  chgpwd   -> cambia la clave de un usuario.
   //  opc      -> devuelve las opciones de la aplicación a las que tiene
   //              acceso el usuario dependiendo del tipo al que pertenezca.
   //  provi    -> devuelve todas las provincias donde hay cuadro médico.
```

```
//  pobla    -> devuelve todas las poblaciones en las que hay cuadro
//           médico de una provincia dada.
//  espec    -> devuelve las especialidades del cuadro médico de una
//           población.
//  asegurados -> busca el asegurado en función de los parámetros
//           facilitados.
//  medicos -> devuelve los prestadores y puntos de asistencia de una
//           provincia-población-especialidad dada.
//

$conex = mysqli_connect($server, $user, $pwd, $db) or
      die("No se puede conectar al servidor MySQL");

// Cambiar el conjunto de caracteres a utf8 */
if (!mysqli_set_charset($conex, "utf8")) {
            printf("Error cargando el conjunto de caracteres utf8:
                  %s\n", mysqli_error($conex));
}

$result = array();

$accion = safe($conex, $_GET['accion']);

switch ($accion) {
      case "usu":
            $usuario = safe($conex, $_GET['usuario']);
            $SQL = "CALL sp_iCMED (20, '".$usuario."', '', '', '',
                  '')";
            break;
      case "getpwd":
            $cod_usuario = safe($conex, $_GET['usuario']);
            $SQL = "CALL sp_iCMED (22, ".$cod_usuario.", '', '', '',
                  '')";
            break;
      case "chgpwd":
            $cod_usuario = safe($conex, $_GET['usuario']);
            $clave = safe($conex, $_GET['clave']);
            $SQL = "CALL sp_iCMED (23, ".$cod_usuario.", '".$clave."',
                  '', '', '')";
            break;
      case "opc":
            $tipo_usuario = safe($conex, $_GET['tipo_usuario']);
            $SQL = "CALL sp_iCMED (30, ".$tipo_usuario.", '', '', '',
                  '')";
            break;
      case "provi":
            $SQL = "CALL sp_iCMED (1, '', '', '', '', '')";
            break;
      case "pobla":
            $provincia = safe($conex, $_GET['provincia']);
            $SQL = "CALL sp_iCMED (2, ".$provincia.", '', '', '', '')";
            break;
      case "espec":
            $provincia = safe($conex, $_GET['provincia']);
            $poblacion = safe($conex, $_GET['poblacion']);
            $SQL = "CALL sp_iCMED (3, ".$provincia.", ".$poblacion.",
                  '', '', '')";
            break;
```

```
        case "asegurados":
              $nombre = safe($conex, $_GET['nombre']);
              $apellido1 = safe($conex, $_GET['apellido1']);
              $apellido2 = safe($conex, $_GET['apellido2']);
              $NIF = safe($conex, $_GET['NIF']);
              $tarjeta = safe($conex, $_GET['tarjeta']);
              $SQL = "CALL sp_iCMED (10, '".$nombre."', '".$apellido1."',
                    '".$apellido2."', '".$NIF."', '".$tarjeta."')";
              break;
        case "medicos":
              $provincia = safe($conex, $_GET['provincia']);
              $poblacion = safe($conex, $_GET['poblacion']);
              $especialidad = safe($conex, $_GET['especialidad']);
              $SQL = "CALL sp_iCMED (40, ".$provincia.", ".$poblacion.",
                    ".$especialidad.", '', '')";
              break;
  }

  // Consulta a base de datos
  $rs = mysqli_query($conex, $SQL);

  // Se pasa cada linea del recordset al array $result
  while($row = mysqli_fetch_assoc($rs)) {
        $result[] = $row;
  }

  // Devuelve los datos codificados con JSON
  echo '{"datos":'.json_encode($result).'}';

  // Cerrar la conexión
  mysqli_close($conex);
?>
```

3.5.1.2 *Procedimiento almacenado*

Como podemos observar en el código del archivo consulta_Mysql.php anterior, se accede a un procedimiento almacenado sp_iCMED), que dependiendo de los parámetros que se faciliten al mismo, ejecutará la consulta apropiada.

La estructura de este procedimiento almacenado es la siguiente y en él están comentados los diferentes parámetros utilizados:

```
PROCEDURE `sp_iCMED`(
  IN opc SMALLINT, IN var1 VARCHAR(50), IN var2 VARCHAR(50),
  IN var3 VARCHAR(50), IN var4 VARCHAR(50), IN var5 VARCHAR(50))
BEGIN
--
-- Las opciones (opc) validas son:
-- 1  -> Devuelve todas las provincias donde hay cuadro médico.
--       Se usa en buscar medicos.
-- 2  -> Devuelve todas las poblaciones de una provincia donde hay cuadro
--       médico.
--       Se usa en buscar medicos. var1 = cod_provincia.
-- 3  -> Devuelve las especialidades con servicio en una poblacion.
--       Se usa en buscar medicos.
-- 10 -> Devuelve los datos del asegurado si este esta contratado por la
--       compañía. Se usa en buscar asegurado
--             var1 = nombre
--             var2 = apellido1
```

```
--              var3 = apellido2
--              var4 = NIF
--              var5 = tarjeta
-- 20 -> Devuelve los datos de usuario. Se usa en login donde va1 =
--       usuario.
-- 22 -> Devuelve la clave de usuario. Se usa en el cambio de clave.
--       var1 = cod_usuario.
-- 23 -> Actualiza la clave de usuario. Se usa en el cambio de clave.
--       var1 = cod_usuario.
-- 30 -> Devuelve las opciones de la aplicación a las que tiene acceso un
--       usuario. var1 = tipo_usuario
-- 40 -> Devuelve los médicos y centros existentes en una población y que
--       tengan la especialidad médica elegida.

   DECLARE S VARCHAR (5000);

   CASE opc
          WHEN 1 THEN
                 // Aquí va la consulta de provincias
          WHEN 2 THEN
                 // Aquí va la consulta de poblaciones
          WHEN 3 THEN
                 // Aquí va la consulta de especialidades
          WHEN 10 THEN
                 // Aquí va la consulta preparada de asegurados
          WHEN 20 THEN
                 // Aquí va la consulta de usuarios
          WHEN 22 THEN
                 // Aquí va la consulta de clave de usuario
          WHEN 23 THEN
                 // Aquí va la consulta de actualización de clave de usuario
          WHEN 30 THEN
                 // Aquí va la consulta de opciones de aplicación
          WHEN 40 THEN
                 // Aquí va la consulta de servicios médicos
   END CASE;
END
```

Las consultas integradas en el procedimiento ya han sido comentadas en el punto 3.3 y no se incluyen en la estructura general del procedimiento almacenado anteriormente mostrado.

3.5.1.3 JSON

Ya hemos visto que el servicio Web devuelve los datos formateados con JSON gracias a la función 'json_encode'; para su decodificación con Xcode se utiliza un diccionario y un array y de él se van extrayendo los datos con 'objectForKey'. Esto lo podemos ver a continuación:

```
     NSError * err_json = nil;

     NSDictionary * dict = [NSJSONSerialization JSONObjectWithData:urlData
options:kNilOptions error:&err_json];

     NSArray * array = [dict objectForKey:@"datos"];

     if(array.count !=0){
          NSString * clave = [NSString stringWithFormat:@"%@",[[array
objectAtIndex:0] objectForKey:@"clave"]];
}
```

Por otro lado, hay que señalar que el acceso a las API de Google para obtener la ruta entre dos puntos también nos devuelve los datos de dicha ruta en JSON.

3.5.2 Clase de inicio

Este es el módulo de acceso a la aplicación donde podremos validarnos como usuario autorizado para el uso de sus funcionalidades.

Se especifican en este apartado el uso del archivo UserDefault.plist, la codificación de la contraseña y por último también se especifica la forma en que se realiza la conexión a los servicios Web desde Xcode, aunque este método se utilice en, prácticamente todas las clases de la aplicación.

3.5.2.1 Fichero UserDefault.plist

Esta aplicación utiliza este archivo para guardar parámetros como la cadena que contiene el servidor Web al que debe conectarse o el último usuario que se validó correctamente en la aplicación y que no se tenga que volver a teclear si el siguiente usuario que quiere usar la aplicación es el mismo. También se está guardando la localización del usuario por defecto para las pruebas con el simulador iPhone.

El contenido del archivo UserDefault.plist es el siguiente:

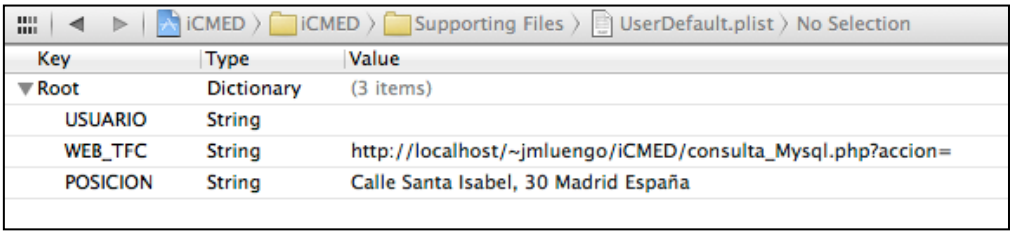


Ilustración 31: Archivo UserDefault.plist

Como ejemplo, para guardar el valor del usuario correctamente validado en el archivo UserDefault.plist se puede realizar mediante las siguientes líneas de código:

```
     NSUserDefaults *defaults = [NSUserDefaults standardUserDefaults];

     [defaults registerDefaults:[NSDictionary
dictionaryWithContentsOfFile:[[NSBundle mainBundle]
pathForResource:@"UserDefault" ofType:@"plist"]]];

     [defaults setObject:[self user] forKey:@"USUARIO"];


     [defaults synchronize];
```

Para recuperar el valor de usuario del archivo UserDefault.plist se puede hacer con las siguientes líneas de código:

```
     NSUserDefaults *defaults = [NSUserDefaults standardUserDefaults];

     [defaults registerDefaults:[NSDictionary
dictionaryWithContentsOfFile:[[NSBundle mainBundle]
pathForResource:@"UserDefault" ofType:@"plist"]]];

     [[self userText] setText:[defaults objectForKey:@"USUARIO"]];
```

3.5.2.2 Encriptación de contraseña

La contraseña de usuario se almacena encriptada mediante un algoritmo de codificación MD5. Para ello hemos hecho uso de la librería CommonDigest:

```
#import "CommonCrypto/CommonDigest.h"
```

El código para calcular la clave de usuario encriptada es el siguiente:

```
- (NSString *) md5:(NSString *) input
{
  const char *cStr = [input UTF8String];
  unsigned char digest[16];

  CC_MD5( cStr, strlen(cStr), digest ); // This is the md5 call

  NSMutableString *output = [NSMutableString
    stringWithCapacity:CC_MD5_DIGEST_LENGTH * 2];

  for(int i = 0; i < CC_MD5_DIGEST_LENGTH; i++)
      [output appendFormat:@"%02x", digest[i]];

  return  output;
}
```

3.5.2.3 Acceso servicios Web desde Xcode

El acceso desde Xcode a los servicios Web se realiza a través de la definición de una URL, haciendo la petición de datos en un con un NSURLRequest y realizando la conexión con NSURLConnection.

El siguiente ejemplo accede al servicio Web para recuperar la clave del usuario tecleado:

```
    [self setWeb:[NSString stringWithFormat:@"%@%s%@",WEB_TFC,
"usu&usuario=", [self user]]];

    [self setWeb:[[self web]
stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding]];

    [self setUrl:[NSURL URLWithString:[self web]]];

    [self setUrlData:[NSData dataWithContentsOfURL:[self url]]];
```

3.5.3 Clase de opciones de la aplicación

Este apartado de la aplicación es donde se muestran las opciones disponibles de la aplicación iCMED; estás opciones variarán dependiendo del tipo de usuario que haya accedido a la aplicación.

La particularidad de este módulo es que se deja abierto a funcionalidades futuras ya que estas se pueden desarrollar con Xcode sin variar este módulo, porque las nuevas funcionalidades estarán definidas y accesibles una vez desarrolladas, en la tabla opciones_menu de la base de datos.

La carga de las opciones de esta pantalla se realiza de forma dinámica dependiendo del usuario y también se carga la referencia del segue que nos llevará a la pantalla de la funcionalidad elegida.

3.5.4 Clase de cambio de contraseña de usuario

Esta clase permite cambiar la clave del usuario validado en la aplicación.

3.5.5 Clase de búsqueda de asegurados

Esta clase permite saber si el la persona que nos llama es asegurado o no. Se permite la búsqueda por nombre, apellidos, NIF y número de tarjeta de asegurado.

3.5.6 Clase de búsqueda de servicios médicos

Esta funcionalidad de la aplicación permite localización dentro del cuadro médico de la compañía de cualquier servicio médico, localizarlo en un mapa e incluso permite buscar la ruta desde la posición actual del usuario.

3.5.6.1 Geolocalización

Para el ahorro de batería del dispositivo se ha realizado un posicionamiento por cambio significativo que utiliza triangulación de antenas de telefonía que sería suficiente para nuestros propósitos, frente a otros métodos que utilizan el GPS de los dispositivos y que consumen más energía.

El código para realizar este posicionamiento es como sigue:

```
     if (nil == [self locationManager])
         [self setLocationManager:[[CLLocationManager alloc] init]];

    [self setPosition:[NSString stringWithFormat:@"%f,%f", [self
locationManager].location.coordinate.latitude, [self
locationManager].location.coordinate.longitude]];
```

3.5.6.2 Marcas en el mapa

Se ha posicionado en el mapa el servicio médico elegido y se ha marcado con un icono personalizado de una cruz roja; si pulsamos sobre ella aparece la información del servicio.

Para ello se ha utilizado MKPointAnnotation en las coordenadas del domicilio del servicio, previamente calculadas. El código que realiza esta acción es el siguiente:

```
if (!geocoder) {
    geocoder = [[CLGeocoder alloc] init];
}

[geocoder geocodeAddressString:_destino completionHandler:^(NSArray
*placemarks, NSError *error) {
    if ([placemarks count] > 0) {
        CLPlacemark *placemark = [placemarks objectAtIndex:0];
        CLLocation *location = placemark.location;
        CLLocationCoordinate2D coordinate = location.coordinate;
        coordinates = [NSString stringWithFormat:@"%f, %f",
coordinate.latitude, coordinate.longitude];
        if ([placemark.areasOfInterest count] > 0) {
            NSString *areaOfInterest = [placemark.areasOfInterest
objectAtIndex:0];
            nombre = areaOfInterest;
```

```
        }

        CLLocationCoordinate2D Medico;
        Medico.latitude = coordinate.latitude;
        Medico.longitude = coordinate.longitude;

        MKPointAnnotation *aPoint = [[MKPointAnnotation alloc] init];
        aPoint.coordinate = Medico;
        aPoint.title = [NSString stringWithString: _nombre_completo];
        aPoint.subtitle = [NSString stringWithFormat:@"%@ en %@",
_desc_especialidad, _domicilio];

        [mapa addAnnotation:aPoint];

        MKCoordinateRegion viewRegion =
MKCoordinateRegionMakeWithDistance(Medico, 400, 400);
        MKCoordinateRegion adjustedRegion = [mapa regionThatFits:viewRegion];
        [mapa setRegion:adjustedRegion animated:YES];
    }
}];
```

Para cambiar el icono por defecto, se ha utilizado el siguiente método:

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id
<MKAnnotation>)annotation {
    static NSString *identifier = @"MyLocation";

    if ([annotation isKindOfClass:[MKPointAnnotation class]]) {
        MKAnnotationView *annotationView = (MKAnnotationView *) [mapView
dequeueReusableAnnotationViewWithIdentifier:identifier];
        if (annotationView == nil) {
            annotationView = [[MKAnnotationView alloc]
initWithAnnotation:annotation reuseIdentifier:identifier];
            annotationView.enabled = YES;
            annotationView.canShowCallout = YES;
            annotationView.image = [UIImage imageNamed:@"CRUZ-ROJA 3D
pin.png"];
        } else {
            annotationView.annotation = annotation;
        }

        return annotationView;
    }

    return nil;
}
```

3.5.6.3 Ruta entre dos puntos con Google API

Se utiliza un servicio Web con la API de Google para averiguar la ruta entre la posición del usuario y el domicilio del servicio médico elegido.

Los datos se devuelven parseados con JSON como ya se ha comentado en el punto 3.5.1.3

Señalar que se puede calcular la ruta a pie, en coche o en transporte público (esta última opción dependiendo de la ciudad), variando el parámetro mode; en nuestra aplicación solo se utlizan los dos primeros métodos.

El código para la llamada al servicio es como sigue:

```
    NSString *urlPath = [NSString
stringWithFormat:@"https://maps.googleapis.com/maps/api/directions/json?origi
n=%@&destination=%@&waypoints=%@|%@&mode=walking&sensor=false", ORIGEN,
_destino, ORIGEN, _destino];
    urlPath = [urlPath
stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding];

    NSURLRequest *theRequest =
      [NSURLRequest requestWithURL:[NSURL URLWithString:urlPath]
      cachePolicy:NSURLRequestUseProtocolCachePolicy
      timeoutInterval:60.0];

    NSURLConnection *theConnection=
      [[NSURLConnection alloc] initWithRequest:theRequest delegate:self];
    if (theConnection) {
        _receivedData = [[NSMutableData alloc] initWithLength:0];
    } else {
        // Inform the user that the connection failed.
    }
```

3.6 Pruebas unitarias

Gracias a las pruebas unitarias (Unit Testing) se permite comprobar el correcto funcionamiento de una aplicación, se facilita y simplifica la integración continua, se documenta el código y se localizan más fácilmente los errores.

Para que unas pruebas se consideren efectivas no deben requerir intervención manual (automáticas), tienen que cubrir todo el código posible (completas), no deben influir entre si (independientes) y deben ser repetibles.

En Xcode está dispone el Framework SenTestingKit que permite programar test unitarios.

Existen dos tipos de pruebas:

- Lógicas, que comprueban el correcto funcionamiento del código por si mismo de forma independiente de la aplicación.
- De aplicación, que comprueban el funcionamiento del código en el propio contexto de la aplicación.

Ambos tipos de prueba pueden realizarse en el simulador iOS, pero sólo las pruebas de aplicación pueden realizarse en los dispositivos físicos.

Un pequeño ejemplo del código programado para las pruebas unitarias de la aplicación en relación a la clase de inicio, se puede ver a continuación:

```
//
//  iCMEDApplicationTests.m
//  iCMEDApplicationTests
//
//  Created by Jose Mariano Garcia Luengo on 07/06/13.
//  Copyright (c) 2013 Jose Mariano Garcia Luengo. All rights reserved.
//

#import "iCMEDApplicationTests.h"
```

```
@implementation iCMEDApplicationTests

- (void)setUp
{
    [super setUp];

    appDelegate = [[UIApplication sharedApplication] delegate];
    inicioController = (InicioController*)
appDelegate.window.rootViewController;
    view = inicioController.view;
}

- (void)tearDown
{
    [super tearDown];
}

- (void)testApplicationDelegate {

    STAssertTrue([appDelegate isMemberOfClass:[AppDelegate class]], @"bad
UIApplication delegate");

    STAssertTrue([inicioController isMemberOfClass:[InicioController class]],
@"bad inicioController");
}

- (void) testLogin {
    NSLog(@"%@ start", self.name);

    // Test 1: Usuario y clave vacios.
    [[inicioController userText] setText:@""];
    [[inicioController pwdText] setText:@""];

    [inicioController initSessionButtonClick:self];

    STAssertTrue([[inicioController userDefError] isEqualToString:@"Falta el
nombre de usuario."], @"Test 1: Fallo con usuario y clave vacios.");

    // Test 2: Usuario introducido y clave vacia.
    [[inicioController userText] setText:@"usuario"];
    [[inicioController pwdText] setText:@""];

    [inicioController initSessionButtonClick:self];

    STAssertTrue([[inicioController userDefError] isEqualToString:@"Falta la
clave."], @"Test 2: Fallo con usuario introducido y clave vacia.");

    // Test 3: Usuario vacio y clave introducida.
    [[inicioController userText] setText:@""];
    [[inicioController pwdText] setText:@"clave"];

    [inicioController initSessionButtonClick:self];

    STAssertTrue([[inicioController userDefError] isEqualToString:@"Falta el
nombre de usuario."], @"Test 3: Fallo con usuario vacio y clave
introducida.");

    // Test 4: Usuario y clave introducidos pero no correctos.
```

```
    [[inicioController userText] setText:@"usuario"];
    [[inicioController pwdText] setText:@"clave"];

    [inicioController initSessionButtonClick:self];

    STAssertTrue([[inicioController userDefError] isEqualToString:@"No existe
el usuario tecleado."], @"Test 4: Fallo con usuario y clave introducidos pero
no correctos.");

    // Test 5: Usuario introducido y existente pero clave introducida pero no
correcta.
    [[inicioController userText] setText:@"jgarcialue"];
    [[inicioController pwdText] setText:@"clave"];

    [inicioController initSessionButtonClick:self];

    STAssertTrue([[inicioController userDefError] isEqualToString:@"Usuario
y/o clave incorrectos."], @"Test 5: Fallo con usuario y clave introducidos
pero clave no correcta.");

    // Test 6: Usuario y clave introducidos y correctos.
    [[inicioController userText] setText:@"jgarcialue"];
    [[inicioController pwdText] setText:@"1234"];

    [inicioController initSessionButtonClick:self];

    STAssertTrue([inicioController userCode] == 1, @"Test 6: Fallo validacion
usuario (codigo devuelto incorrecto).");

    NSLog(@"%@ end", self.name);
}

@end
```

4 Conclusiones

Una vez finalizado el presente proyecto obtenemos las siguientes conclusiones:

- Se ha realizado un estudio previo de las diferentes alternativas tecnológicas para el desarrollo del proyecto eligiendo la más adecuada a nuestros fines.
- Se ha realizado el modelado de datos necesario para el correcto funcionamiento de la aplicación.
- Se ha realizado el desarrollo del servicio Web basado en PHP devolviendo los datos con JSON.
- Se han estudiado diferentes funcionalidades en Xcode para poder desarrollar la aplicación iOS para iPhone.
- Finalmente podemos decir que se han conseguido los objetivos establecidos para el proyecto.

En el transcurso del TFC se han seguido las pautas de entrega de las 3 PEC, lo cual ha facilitado la consecución con éxito de los objetivos del proyecto.

El inicio del desarrollo con Xcode supuso un esfuerzo extra ya que los conocimientos de la herramienta eran prácticamente nulos.

También se han tenido que realizar pequeños ajustes en el diseño inicial, fruto del mencionado desconocimiento de la herramienta de desarrollo.

Aun así, y aunque mis conocimientos en el desarrollo de aplicaciones para dispositivos móviles basados en iOS eran casi nulos, el aprendizaje ha sido gratificante y la experiencia adquirida a lo largo del proyecto ha sido bastante alta, tanto es así que me queda la sensación de que ahora, con los conocimientos adquiridos podría mejorar el proyecto de forma sustancial.

Este proyecto que conforma el TFC me ha permitido realizar un proyecto desde el inicio del mismo hasta la entrega final del producto y además, me ha servido para iniciarme en el mundo del desarrollo de aplicaciones para dispositivos móviles basados en iOS y no descarto continuar profundizando en el desarrollo con Xcode e incluso inscribirme en el programa Apple para desarrolladores y publicar alguna aplicación en el App Store de Apple.

5 Líneas de evolución futuras

En un futuro se pueden incluir funcionalidades a medida que sea necesario. La aplicación está abierta al desarrollo de cualquier funcionalidad que la compañía de seguros estime oportuno para el desarrollo de su negocio.

Partimos de la base de una compañía ya funcional con aplicativos corporativos ya desarrollados, tanto para acceso en oficinas como acceso en remoto a través de su Web corporativa.

Como ejemplo, se podrían incluir las siguientes:

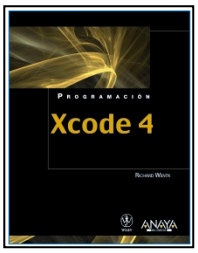
- Consulta de los datos completos de la póliza.
- Petición de autorización para prueba médica.
- Consulta de histórico médico del asegurado.
- Petición de cita.
- Dirección a servicio médico de especialidad concreta por proximidad.

Por otro lado, se podrían realizar las modificaciones oportunas para que la aplicación se pudiese ejecutar el un iPad aprovechando el mayor tamaño de pantalla de este dispositivo.

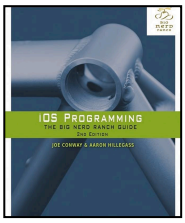
También se podría adaptar de forma sencilla el desarrollo a los dispositivos Android, ya que la parte de servicio Web debería ser la misma por ser independiente.

6 Bibliografía

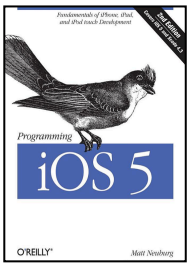
6.1 Publicaciones



Programación Xcode 4
Wentk Richard.
Anaya Multimedia.

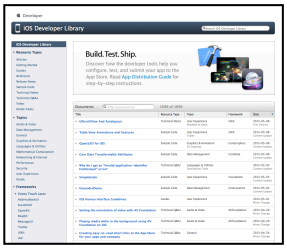


iOS Programming. The big nerd ranch Guide (2nd Edition)
Joe Conway & Aaron Hillegas

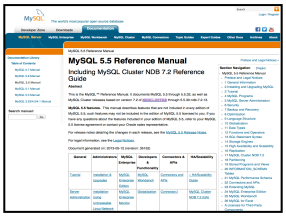


Programming iOS 5
Matt Neburg
O'REILLY

6.2 Internet

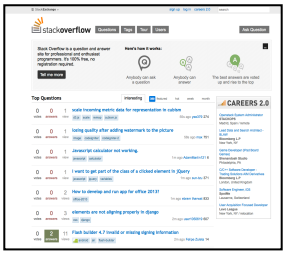


iOS Developer Library
https://developer.apple.com/library/ios/navigation/

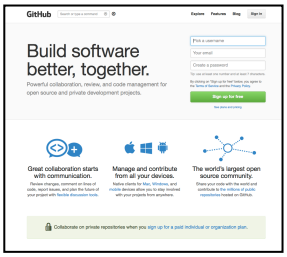


MySQL 5.5 Reference Manual
http://dev.mysql.com/doc/refman/5.5/en/index.html

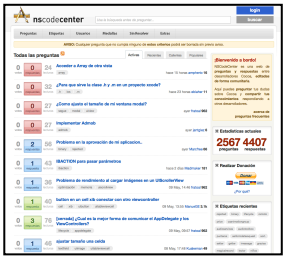
PHP: Referencia del lenguaje - Manual
http://www.php.net/manual/es/langref.php



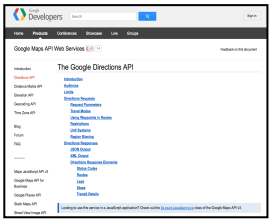
Stackoverflow
http://stackoverflow.com



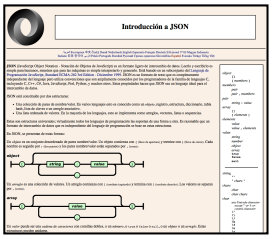
GitHub
https://github.com



NSCodeCenter: Preguntas y respuestas para desarrolladores Cocoa
http://www.nscodecenter.com



The Google Directions API . Google Maps API Web Services - Google Developers
https://developers.google.com/maps/documentation/directions/



JSON
http://www.json.org/json-es.html