sanicen: Información y localización de centros sanitarios en España

Emilio Jiménez del Moral
Grado de Ingeniería Informática

Desarrollo Aplicaciones Móviles Dispositivos Móviles (Android)

Consultor: Albert Grau Perisé

Profesor responsable de la asignatura: Álvaro del Álamo Cortés

Junio 2016

FICHA DEL TRABAJO FINAL

Título del trabajo:	**sanicen**: Información y localización de centros sanitarios en España
Nombre del autor:	Emilio Jiménez del Moral
Nombre del consultor/a:	Albert Grau Perisé
Nombre del PRA:	Álvaro del Álamo Cortés
Fecha de entrega (mm/aaaa):	06/2016
Titulación:	Grado de Ingeniería Informática
Área del Trabajo Final:	*Desarrollo Aplicaciones Móviles Dispositivos Móviles (Android)*
Idioma del trabajo:	*Castellano*
Palabras clave	*Android, Sanidad, Mapa*

Resumen del Trabajo

Este proyecto lleva por nombre **"sanicen: Información y localización de centros sanitarios en España".** Se trata de una aplicación Android que pretende ayudar a ciudadanos y profesionales a encontrar información sobre los centros sanitarios a nivel nacional.

Para ello nos basaremos en los datos publicados en el Catálogo Nacional de Datos Abiertos. Este es un proyecto que se inició en 2009 y cuyo objetivo es ser un nexo de unión entre los organismos públicos y los ciudadanos.

Hasta el momento no había ninguna aplicación que mostrara esta información y la versión web que existe no tiene un diseño responsive adecuado que permita navegar a través del móvil.

Otra necesidad era localizar los centros en un mapa. Este punto es muy importante para que un ciudadano pueda visualizar donde están los centros de salud más cercanos.

Por tanto, con esta aplicación se consigue cubrir los dos objetivos anteriores: disponer de una aplicación movil con datos actualizados de los centros sanitarios a nivel nacional donde poder realizar búsquedas sobre ellos según las necesidades del usuario, y localizarlos en el mapa para su fácil localización. Además, nos permite conocer en detalle los datos de los centros sanitarios que sean de nuestro interés (dirección, titularidad, etc), así como llamar a cualquiera de ellos o establecer una ruta desde la ubicación del usuario hasta el centro seleccionado.

Abstract:

This project is called "sanicen: Information and location of health centers in Spain". This is an Android application that aims to help citizens and professionals to find information on health centers nationwide.

To do this we will build on the data published in the National Catalogue of Open Data. This is a project that began in 2009 and aims to be a link between public administration and citizens.

So far there was no application to show this information and the web version that exists doesn't have adequate responsive design that allows navigate through the mobile.

Another need was to locate the center on a map. This point is very important for a citizen to visualize where the nearest health centers are.

Therefore, with this application it is possible to cover the above two objectives: to have a mobile application with updated data on health centers nationwide where you can search on them according to user needs, and locate them on the map for easy location. It also allows us to know in detail the health data centers that are of interest to us (address, title, etc.) and call any of them or establish a route from the user's location to the selected center.

Índice de contenidos

Índice de ilustraciones

1 DEFINICIÓN DEL PROYECTO

1.1 Descripción del proyecto

Este proyecto lleva por nombre **"sanicen: Información y localización de centros sanitarios en España".** Se trata de una aplicación Android que pretende ayudar a ciudadanos y profesionales a encontrar información sobre los centros sanitarios a nivel nacional.

Se está observando a través de diversos estudios, que el tiempo que se dedica a las apps, se está aproximando cada vez más al tiempo que se dedica a navegar por internet. Por eso, es muy interesante llevar a los ciudadanos este tipo de información de manera rápida y fácil, a través de un smartphone o tableta Android.

Para ello nos basaremos en los datos publicados en el Catálogo Nacional de Datos Abiertos. Este es un proyecto que se inició en 2009 y que consiste en la difusión de información del sector público, siendo uno de sus objetivos ser un nexo de unión entre los organismos públicos y los ciudadanos o profesionales que demandan información. Entre los datos publicados encontramos el directorio de centros y servicios del sistema nacional de salud (SNS) y, que nos permite conocer toda la relación de centros de atención primaria, centros de atención urgente extrahospitalaria y hospitales, con datos como su dirección, teléfono, equipos de alta tecnología, etc.

1.2 Justificación

Actualmente no hay aplicaciones que muestren información actualizada procedente del Catálogo Nacional de Datos Abiertos sobre centros sanitarios, y la versión web que existe no tiene un diseño responsive adecuado que permita navegar a través del móvil.

Otra necesidad a cubrir es localizar los centros en un mapa. Este punto es muy importante para que un ciudadano pueda visualizar donde están los centros de salud más cercanos.

1.3 Objetivos del proyecto

El objetivo principal de este proyecto es realizar una aplicación Android que de respuesta a la necesidad de buscar información sobre los centros sanitarios de España en base a diferentes criterios, y de localizarlos en un mapa. Más concretamente se pretende:

- Desde una pantalla inicial poder seleccionar la comunidad autónoma, la provincia y el municipio a consultar.
- Poder filtrar los resultados a mostrar según el tipo de centro: Centros de Atención Primaria del SNS, Centros de Atención Urgente Extra hospitalaria y Hospitales. En función del tipo de centro que elija

aparecerán otra serie de filtros pero estos ya relacionados con el tipo de centro escogido.

- Situar sobre el mapa los centros sanitarios que cumplan los requisitos seleccionados.
- Al pinchar sobre un centro, se podrá:
  - Ver la información en detalle del centro.
  - La opción de llamar al centro.
  - La opción de navegar hasta el centro.

La aplicación android hará uso de mapas basados en la API de Google Maps. Hay muchas plataformas móviles disponibles en el mercado, pero la mayoría de las empresas prefieren el desarrollo Android, ya que ofrece grandes ventajas que no se pueden encontrar en cualquier otra plataforma móvil:

- Es un sistema para móviles gratuito y no hace falta pagar nada para adquirir las herramientas necesarias para programar aplicaciones para él.
- Ofrece excelentes gráficos de apoyo. Los usuarios de los dispositivos móviles siempre quedan impresionados y son atraídos por la alta calidad de los gráficos que la plataforma ofrece. Android proporciona soporte incorporado para potentes gráficos en 2D y 3D que permiten a las empresas atraer a los usuarios a través de aplicaciones originales.
- No está atado a un único fabricante de dispositivos.
- Campo de pruebas más ágil. Si tienes tu cuenta de desarrollador en Android, y creas tu aplicación, subirla al Play Store es tan sencillo como un par de clicks. Si quieres hacerlo en la tienda de iOS es necesario pasar un proceso de aprobación que tarda en torno a una semana, si todo va bien y sale aprobada a la primera.

Por otro lado, las ventajas de usar la API de Google Maps son las siguientes:

- Rapidez. El poder integrar en cuestión de segundos mapas en nuestras aplicaciones.
- Robustez. Está tan sumamente extendida que ha sido probada en toda clase de situaciones de estrés, por lo que podremos asegurar que su respuesta será adecuada en la inmensa mayoría de los casos.
- Imagen. Cuando los usuarios usan a menudo un buen servicio lo acaban reconociendo, y el asociar nuestra imagen a la de un grande como Google, puede ayudarnos a contagiarnos de la buena percepción que los usuarios tienen de este gigante.
- Mantenimiento. Los servicios accedidos a través de la API están en una constante mejora.

La estrategia a seguir para llevar a cabo el trabajo es realizar una aplicación nueva, los motivos son:

- La temática es muy específica
- No hay aplicaciones libres con estas características que se puedan reutilizar.
- Desde el punto de vista de aprendizaje es una buena opción desarrollar los trabajos desde cero.

1.4 Material

El material inicial que se va a utilizar para el desarrollo del proyecto se detalla a continuación.

Material hardware

El material hardware del que se dispone es el siguiente:

- Lenovo ThinkPad x220. Intel(R) Core(TM) i5-2520M CPU 2.50GHz, 8GB de RAM con Ubuntu Ubuntu 12.04.5 LTS

Material software

El material software que se utilizará en principio para el desarrollo es el siguiente:

- Android Studio 2.1.1.
- LibreOffice 3.5.7.2
- Gantt Project 2.5.5.
- Dia v0.97.2: editor de diagramas.
- http://www.balsamiq.com/ : Balsamiq es una herramienta interactiva de wireframing.
- https://github.com/ **:** servicio de alojamiento basado en web, usando repositorio Git.

2 ANÁLISIS DE REQUISITOS

En este aparatado se definirá de manera clara y precisa las funcionalidades y restricciones que tendrá el sistema que se desea construir. A partir de esta especificación estaremos en condiciones de establecer un diseño que se ajuste a los requerimientos aquí expuestos.

2.1 Funciones del producto

La aplicación tendrá funciones tales como:

- Localizar en el mapa: se mostrarán en el mapa los centros sanitarios del municipio seleccionado, que cumplan con las condiciones que se hayan especificado por parte del usuario (en caso de que el usuario haya especificado alguna).
- Datos de un centro: se trata de mostrar los datos conocidos del centro seleccionado; su nombre, dirección, tipo de gestión, teléfono, etc.
- Llamar al centro: a partir de los datos del centro, dispondremos del teléfono del mismo, por lo que se podrá establecer una llamada en caso de requerirlo.
- Localizar la ubicación del usuario en el mapa: se podrá situar y centrar al usuario en el mapa para ver los centros sanitarios que tiene alrededor.
- Calcular y mostrar ruta: se podrá mostrar la ruta desde un centro sanitario seleccionado hasta la ubicación del usuario.

2.2 Restricciones

La restricción más importante de esta aplicación es la información con la que podemos trabajar, ya que depende directamente del Catálogo Nacional de Datos Abiertos publicado para su explotación por este tipo de aplicaciones.

Sobre cada centro sanitario podremos consultar los siguientes datos:

Datos comunes:

- ✔ Nombre.
- ✔ Tipo de centro.
- ✔ Dirección.
- ✔ Código postal.
- ✔ Provincia.
- ✔ Municipio.
- ✔ Localidad.
- ✔ Teléfono.

Específicos de centros de atención primaria:

- ✔ Zona básica.
- ✔ Área de salud.
- ✔ Dependencia de gestión.
- ✔ Modalidad de gestión
- ✔ Subtipo de centro.

- ✔ Acreditación docente.

Específicos de atención urgente extrahospitalaria:

- ✔ Subtipo de centro.
- ✔ Horario.

Específicos de hospitales:

- ✔ Teléfono 2.
- ✔ Email.
- ✔ Número de camas.
- ✔ Concierto.
- ✔ Acreditación docente.
- ✔ Finalidad asistencial.
- ✔ Dependencia patrimonial.
- ✔ Dependencia funcional.
- ✔ Equipos de alta tecnología.

2.3 Requisitos específicos

Pasamos ahora a profundizar un poco más en los requisitos con los que debe cumplir nuestra aplicación:

Requisitos funcionales

- ✔ REQ01 Selección de la comunidad autónoma, provincia y municipio: el usuario podrá seleccionar la comunidad autónoma, provincia y municipio sobre la que quiere consultar los centros sanitarios.
- ✔ REQ02 Selección del tipo de centro sanitario: el usuario tendrá la posibilidad de seleccionar el tipo de centro sanitario que quiere consultar entre centros de atención primaria, centros hospitalarios, centros de atención urgente extrahospitalaria.
- ✔ REQ03 Localizar centros en el mapa: el usuario podrá visualizar la localización de los centros sanitarios que cumplan con los requisitos que él ha establecido.
- ✔ REQ04 Descripción del centro: el usuario tendrá la posibilidad de visualizar la información disponible del centro sanitario, tras pulsar con el dedo sobre él.
- ✔ REQ05 Llamar por teléfono al centro: el usuario podrá llamar al centro si se dispone de su número de teléfono entre los datos conocidos de dicho centro.

- ✔ REQ06 Situar al usuario en el mapa: el usuario podrá situar y centrar su localización en el mapa, de modo que pueda ver los centros que tiene en sus cercanías; siempre y cuando otorgue el permiso necesario para ello.
- ✔ REQ07 Calcular la ruta desde un centro sanitario a la ubicación del usuario: el usuario podrá visualizar la ruta desde su ubicación al centro seleccionado, de modo que disponga de la posibilidad de ver cómo se llega al centro. Este requisito también está vinculado a la concesión por parte del usuario, del permiso para que la aplicación pueda concocer su ubicación.

Requisitos de interfaces externas

- ✔ REQ08 Logotipo: El logotipo será propio y permitirá identificar el principal objetivo de la aplicación.
- ✔ REQ09 Interfaces del usuario: Se podrá comunicar con el usuario para aprovechar los requisitos del sistema, el usuario indicará al sistema las operaciones que debe realizar e introducirá los datos que el sistema le pida.
- ✔ REQ10 Interfaces del software: La comunicación entre los módulos del sistema se realizará mediante bases de datos relacionadas.

Requisitos de rendimiento

- ✔ REQ11 Tiempo de repuesta: La respuesta que dará el sistema con respecto a la petición del usuario deberá ser en tiempo real.

3 DISEÑO CENTRADO EN EL USUARIO

3.1 Usuarios y contexto de uso

Métodos de indagación

Unos de los métodos de indagación escogido es el de observación. En nuestro caso al ser una aplicación de datos abiertos, lo que hacemos es observar como actúan los usuarios como comunidad ante los distintos portales de este tipo de datos. El resultado es que los ciudadanos vienen siendo muy pro activos en cuanto a las consultas de estos tipos de datos y esto se refleja en el número de visitas y la participación en los foros de estás web oficiales. La tendencia es que cada vez usamos más los dispositivos móviles para buscar información, así que la conclusión es que si la información que queremos facilitar a los usuarios ya es demandada vía web, la aplicación móvil seguramente sea muy demandada por los usuarios.

Al usuario le interesa poder conocer los centros sanitarios más cercanos, por este motivo otro método de indagación es la entrevista a distintos ciudadanos para conocer la experiencia de uso que esperan de la aplicación.

Tras aplicar este método, las personas entrevistadas coinciden en que necesitan que la aplicación se fácil de usar y permita de forma rápida llegar en pocos pasos a los resultado de búsqueda.

Otro método de indagación realizado es el análisis competitivo. Se ha analizado la existencia de aplicaciones similares en el Play store y prácticamente la inexistencia de aplicaciones de este tipo, por un lado nos impide comparar los distintos productos para conocer las funcionalidades comunes y estudiar las interfaces, pero por otro lado llegamos a la conclusión que puede ser una ventaja para nuestra aplicación no tener competencia.

Perfiles de usuario

Los usuarios de esta aplicación van a ser muy heterogéneos, ya que el perfil al que va dirigido es al ciudadano. Los ciudadanos necesitan información de los servicios sanitarios de los que dispone o buscan información por intereses particulares. Este grupo abarca un gran rango de edad por lo que puede o no tener experiencia en aplicaciones móviles.

Listado de tareas que necesitarán realizar los ciudadanos:

- Poder buscar centros sanitarios por comunidad autónoma, provincia y municipio.
- Poder buscar centros de distinto tipo ya sean centros de atención primaria, centros de atención urgente extrahospitalaria y hospitales.
- Poder buscar centros por algunos criterios de importancia como la finalidad asistencial.
- Ver los resultados de la búsqueda localizados en un mapa.
- Centrar el mapa en la ubicación del usuario para visualizar los centros más cercanos.
- Ver la información en detalle de un centro.
- Tener la opción de llamar a un centro.
- Tener la opción de navegar hasta el centro.

El contexto de uso del usuario será en cualquier lugar, ya sea en su domicilio buscando información general en un municipio o en la calle buscando centros cerca de su ubicación. Además podrá hacer uso de la aplicación a cualquier hora del día, ya sea de madrugada para solicitar información sobre los centros con urgencias como por el día para pedir cita en un centro de atención primaria.

Como vemos, los usuarios candidatos a usar la aplicación puede variar mucho en edad, por lo que puede ser que el usuario esté poco o muy acostumbrado a usar aplicaciones de este tipo. Esto es un rasgo a tener en cuenta, ya que debemos diseñar una aplicación lo más intuitiva posible, fácil de usar y que ofrezca de manera sencilla y visual la información que les interesa.

En el apartado anterior se citan algunas conclusiones resultantes de aplicar los métodos de indagación.

3.2 Diseño conceptual

Escenarios

A continuación se exponen algunos escenarios que se pueden cubrir con la aplicación:

- Aitor jubilado de 65 años ha llegado a Andalucía para hacer turismo, al segundo día comienza a encontrarse mal y busca información sobre los centros de salud con servicio de urgencias, finalmente decide ir al más cercano.
- Pedro estudiante de bellas artes de 21 años ha tenido un accidente y necesita rehabilitación, por lo que busca centros de salud que provean del servicio de rehabilitación.
- María informática de 30 años acaba de mudarse a otra zona de la ciudad y necesita pedir cita en el centro de salud más cercano a su casa, por lo que necesita buscar y llamar a este centro.

Flujos de interacción

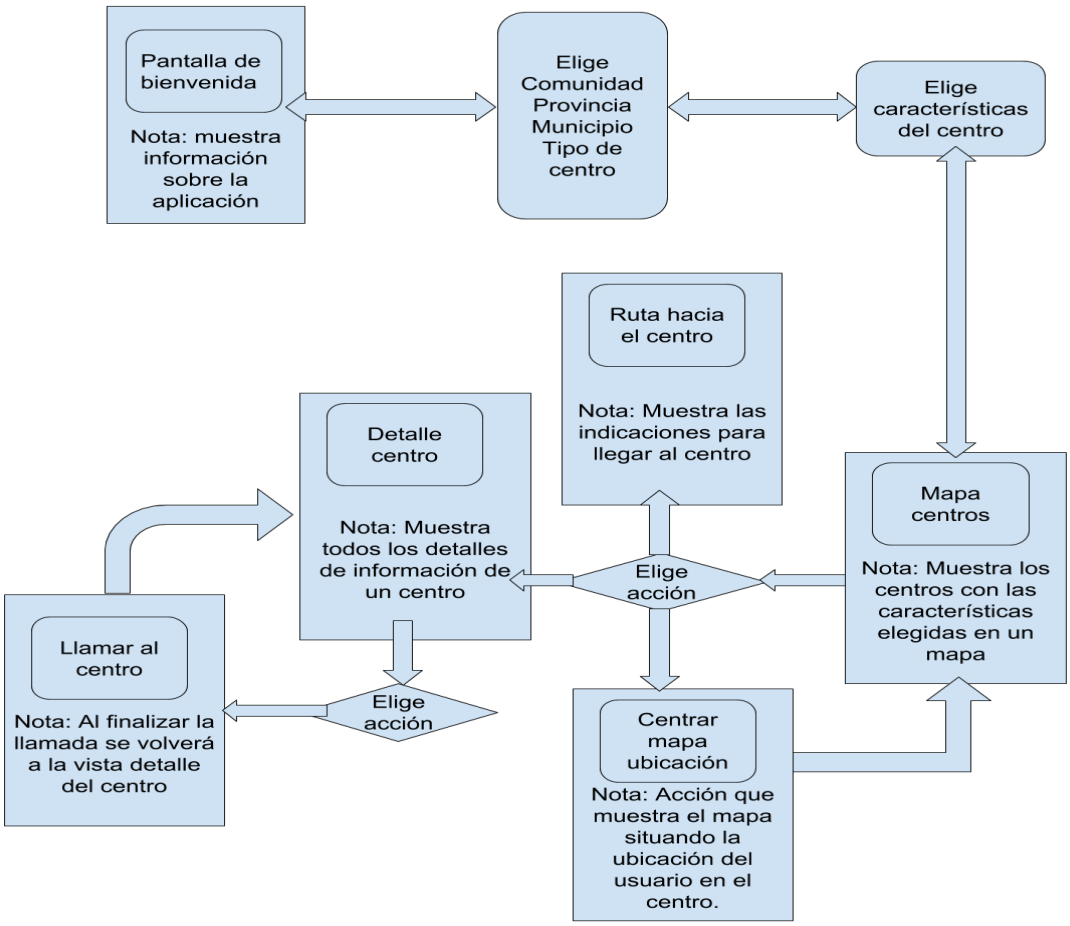
La aplicación comienza con una pantalla de bienvenida y le siguen varias pantallas para elegir comunidad autónoma, provincia, municipio, tipo y características del centro. Desde estas pantallas de filtros de búsqueda se puede volver a la anterior para poder modificar los criterios.

La siguiente pantalla después de introducir los criterios de búsqueda, es una vista donde se muestra un mapa con los centros localizados. Desde esta vista se pueden realizar varias acciones: centrar la ubicación del usuario en el mapa, obtener la ruta al centro o ver los detalles de un centro.

El usuario podrá llamar al centro desde la vista de detalle del mismo. Si el usuario llama al centro una vez finalizada la llamada volverá a la vista del detalle del centro.

Cada una de las pantallas de la aplicación se explican en más detalle en el apartado de prototipado.

A continuación se muestra el diagrama del flujo de interacción.



3.3 Prototipado

En este apartado diseñaremos las pantallas que se van a presentar al usuario y la información que se le muestra o solicita; en definitiva, se diseñará la navegación que se pretende obtener en la aplicación a través de bocetos.

A continuación se muestra pantalla a pantalla la interfaz que se va a construir. Debido a la facilidad de uso de la herramienta utilizada no he visto necesario realizar los bocetos en papel, por ello he realizado estos bocetos directamente con dicha herramienta.

Pantalla inicial

Esta será la pantalla de inicio de la aplicación. En la parte superior de la pantalla tendremos una barra de herramientas, donde encontraremos el nombre de la aplicación. Con esta pantalla se presenta al usuario una breve descripción de la aplicación a través de un texto y el icono de la aplicación. También se le ofrece un botón para que comience a utilizar la aplicación si lo desea.

Con ella, simplemente se pretende dar al usuario una idea de lo que se va a encontrar, antes de pasar a usar la aplicación.

Ilustración 2: Diseño pantalla "Inicio"

Selección de la zona geografica y del tipo de centro

En esta pantalla es donde realmente se comienza a utilizar la aplicación. Se comienza mostrando un listado (por orden alfabético) de las comunidades autónomas. Al seleccionar cualquiera de ellas, se motrarán en el selector de provincias las distintas provincias pertenecientes a la comunidad autónoma escogida. Igualmente al seleccionar una provincia se mostrarán para escoger los distintos municipios de la provincia seleccionada. Finalmente el usuario podrá elegir el tipo de centro.

Ilustración 3: Diseño pantalla "Zona geográfica"

Selección de la finalidad asistencial

Llegados a este punto, sabemos la zona geográfica que al usuario le interesa consultar. Ahora, vamos a refinar un poco la búsqueda de centros según los intereses del usuario. En esta pantalla sólo si el usuario ha escogido como tipo de centro hospitales, se le solicita de forma opcional la finalidad asistencial. Se ofrece una lista de checkbox, ya que es posible seleccionar más de un tipo de filtro.

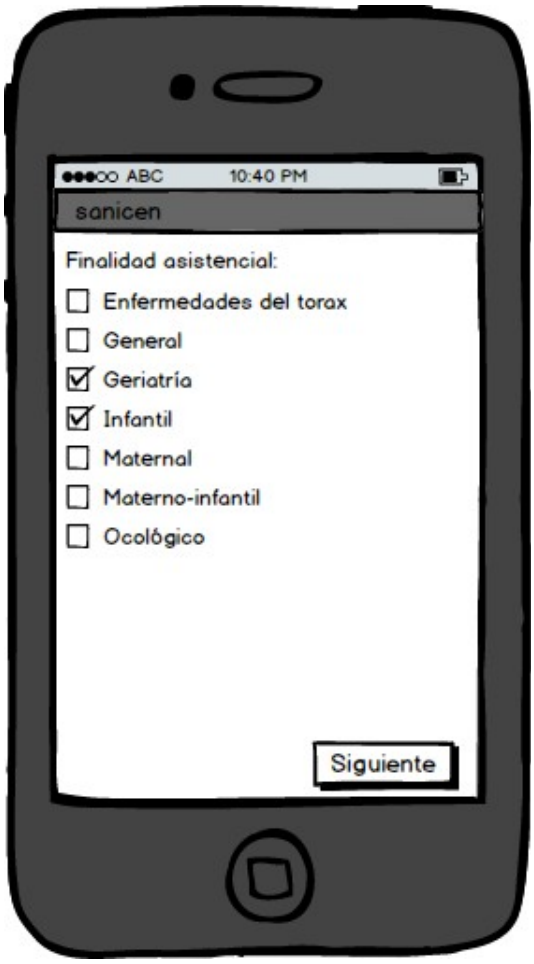


Ilustración 4: Diseño pantalla "Tipos de centros"

Visualizar centros en el mapa

En este punto se presentan en el mapa la localización de los centros que responden a los criterios establecidos, tanto geográficamente, como en tipo de centro y finalidad asistencial. En el mapa aparecerá un icono marcando la situación de cada centro.

Dentro del mapa vamos a encontrar dos iconos:

- Mi ubicación: el cual nos mostrará nuestra ubicación en pantalla.
- Mostrar ruta al centro: nos mostrará las indicaciones necesarias para poder llegar al centro una vez este haya sido seleccionado en el mapa.

Ilustración 5: Diseño pantalla "Centros en el mapa"

Información de un centro seleccionado

Desde la pantalla anterior, donde se nos muestra en el mapa todos los centros que cumplen los requisitos que el usuario busca; el usuario puede también pulsar sobre el icono del centro que le interese por su situación, para conocer toda la información disponible del mismo. Al pulsar sobre el marcador de un centro se mostrará un diálogo con el nombre del centro y un botón que nos llevará a ver los detalles del centro.

En el siguiente boceto vemos como se mostrará dicha información. El usuario podrá llamar directamente al centro pulsando sobre el teléfono.

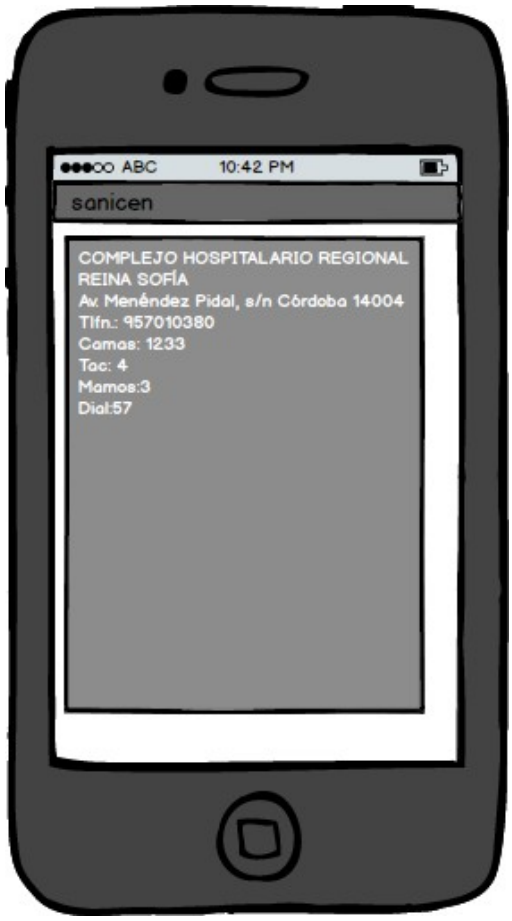


Ilustración 6: Diseño "Información en detalle de un centro"

Mi ubicación

Desde la vista del mapa, tras pulsar en el botón "Mi ubicación" se situará en el mapa la localización actual del usuario. El icono utilizado será distinto del usado para localizar a los centros.

Con esta pantalla se consigue ver de un vistazo los centros que el usuario tiene más cercanos, como otro dato que le puede ayudar a la hora de decidir el centro que más le interesa, una vez conocidos todos los centros que cumplen los requisitos deseados.

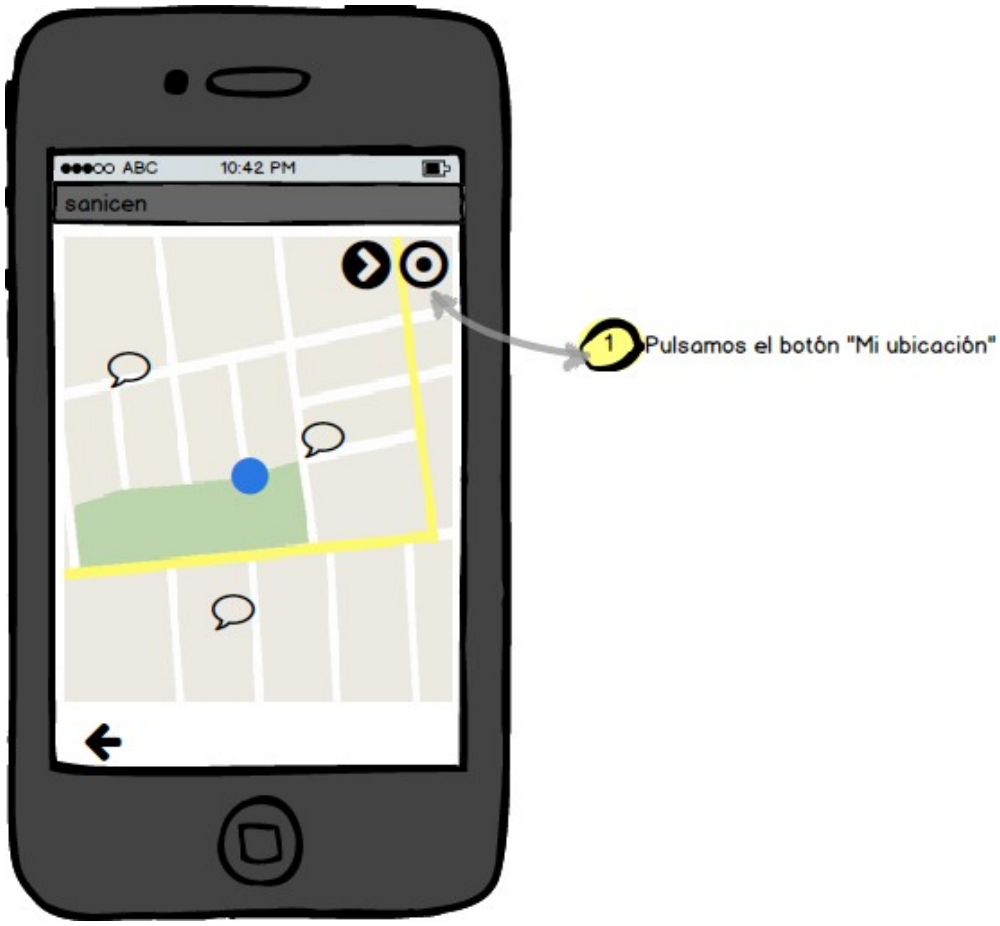


Ilustración 7: Diseño pantalla “Mi ubicación”

3.4 Evaluación

Para realizar la evaluación se escogerán una serie de usuarios de distintas edades y profesiones para que prueben la aplicación. Mientras usan la aplicación se realizarán las siguientes preguntas:

- ¿Se visualiza correctamente en la vista de inicio el botón de comenzar y se entiende que su función es comenzar a realizar una búsqueda?
- ¿Qué esperas que ocurra cuando elijas una comunidad autónoma?
- ¿Qué esperas que ocurra cuando elijas una provincia?
- ¿Qué esperas que ocurra cuando elijas un municipio?
- ¿Qué esperas que ocurra cuando elijas el tipo de centro?
- ¿Qué esperas que ocurra si no eliges ninguna finalidad asistencial?
- ¿Que esperas que ocurra si pulsas el botón de android hacia atrás?

- ¿Entiendes la sucesión de pantallas como un asistente de búsqueda?

- En la vista del mapa ¿que esperarías ver cuando pulses unos de los puntos?
- En la vista del mapa ¿que esperarías ver cuando pulses el botón ver detalle de alguno de los centros?
- ¿Qué pantalla esperas que salga cuando cierres la vista detalle de un centro?
- ¿Puedes predecir el significado de los iconos situados en el interior del mapa?
- En la vista detalle de un centro ¿Puedes predecir que ocurre si pulsas en el teléfono?

Las tareas que los usuarios deberán realizar son:

- Realizar una búsqueda en tu municipio de hospitales con una finalidad asistencial determinada.
- Escoger uno de los centros desde la vista mapa.
- Ver la información del centro y llamar al centro.
- Ver la información del centro y obtener la ruta a su dirección.
- Salir de la vista detalle del centro.
- Centrar el mapa en tu ubicación.

Las preguntas sobre estas tareas están recogidas en el apartado anterior, y se irán realizando antes y después de decirle al usuario que realice dichas tareas.

4 DISEÑO TÉCNICO

El objetivo de esta fase es realizar el diseño del sistema desde un punto de vista más técnico, para lo cual, se usarán técnicas de modelado que nos ayudarán a comprender mejor el sistema que estamos desarrollando. Se pretende definir completamente la solución que se va a implementar.

4.1 Modelo de datos

Para empezar, vamos concretar el dominio de la información que se tratará en el sistema. Para ello partiremos de los siguientes supuestos:

- S1 – Los tipos de centros sanitarios a consultar serán: centro de atención primaria, centro de atención urgente extrahospitalaria y centro hospitalario.

- S2 – Se mantendrá la siguiente información de todos los centros sanitarios: nombre, tipo de centro, dirección, código postal, provincia, municipio, localidad y teléfono.
- S3 – Además, para los centros de atención primaria también se mantendrán los siguientes datos: zona básica, área de salud, dependencia de gestión, modalidad de gestión, subtipo de centro y acreditación docente.
- S4 – Si se trata de un centro atención urgente extrahospitalaria, además de lo expuesto en el supuesto 2 (S2), también se mantendrá la siguiente información: subtipo de centro y horario.
- S5 – Si es un centro hospitalario, además de lo explicado en el S2, se mantendrá la siguiente información: teléfono 2, email, número de camas, concierto, acreditación docente, finalidad asistencial, dependencia patrimonial, dependencia funcional y equipos de alta tecnología.
- S6 – Se deben obtener las coordenadas latitud/longitud de los centros a partir de su dirección, para poder geolocalizarlos en el mapa, ya que este dato no está disponible en el Catálogo Nacional de Datos Abiertos.

Análisis de los tipos de entidad

Tipo de entidad CCAA:

Representa a las comunidades autónomas con las que podremos trabajar en nuestro sistema.

Se consideran los siguientes atributos:

- ID: identificador de la comunidad autónoma.
- NOMBRE: nombre de la comunidad autónoma.

Tipo de entidad PROVINCIAS:

Representa a las provincias con las que podremos trabajar en nuestro sistema.

Se consideran los siguientes atributos:

- IDPROV: Identificador de la provincia.
- NOMBRE: nombre la provincia.
- IDCAA: identificador de la comunidad autónoma a la que pertenece

Tipo de entidad MUNICIPIOS:

Representa a los municipios sobre las que podremos visualizar los centros sanitarios.

Se consideran los siguientes atributos:

- CODMU: identificador del municipio.
- CODPROV: identificador de la provincia a la que pertenece.
- MUNICIPIO: nombre del municipio.

Tipo de entidad SIAP_AREASALUD_CD:

Representa a las distintas áreas de salud que podemos encontrar en una provincia.

Se consideran los siguientes atributos:

- IDAREA: identificador del área de salud.
- NOMBRE: nombre del área de salud.
- IDCOMUNIDAD: identificador de la comunidad autónoma a la que pertenece el área.
- IDPROVINCIA: identificador de la provincia donde se encuentra el área de salud.

Tipo de entidad SIAP_ZONABASICA:

Representa a las distintas zonas básicas que nos podemos encontrar dentro de un área de salud.

Se consideran los siguientes atributos:

- IDZONA: identificador de la zona.
- IDAREA: identificador del área de salud al que pertenece.
- NOMBRE: nombre de la zona.

Tipo de entidad SIAP_TIPOCENTRO:

Representa a los distintos tipos de centros de atención primaria que podemos encontrar.

Se consideran los siguientes atributos:

- ID: identificador del tipo de centro.
- TIPOCENTRO: nombre descriptivo del tipo de centro.

Tipo de entidad SIAP_CENTROS:

Representa a los centros de atención primaria, y para ello, se consideran los siguientes atributos:

- IDCENTRO: identificador del centro de atención primaria.
- IDAREA: identificador del área de salud al que pertenece.
- IDZONABASICA: identificador de la zona.

- TIPOCENTRO: identificador del tipo de centro que le corresponde.
- NOMBRE: nombre del centro de atención primaria.
- DIRECCIÓN: dirección del centro.
- LOCALIDAD: localidad en la que se encuentra situado el centro.
- TELEFONO: teléfono de contacto del centro.
- ESDOCENTE: indica si se trata de un centro con docencia o no.
- CODMU: identificador del municipio al que pertenece.
- CP: código postal del centro sanitario.
- D_GESTION: descripción de la gestión del centro.
- C_GESTION_ID: identificador del modelo de gestión.
- Lat y Long: oordenadas latitud/longitud del centro.

Tipo de entidad SIAP_MOD_GESTIONES:

Representa los distintos tipos de gestión que podemos encontrar en los centros de atención primaria.

Se consideran los siguientes atributos:

- C_GESTIION_ID:identificador del modelo de gestión.
- T_NOMBRE: nombre que describe el modelo de gestión.

Tipo de entidad CH_CATALOGO:

Representa a los centros hospitalarios, y para ello, se consideran los siguientes atributos:

- CODID: identificador del centro hospitalario.
- NOMBRE: nombre del centro hospitalario.
- DIRECCIÓN: dirección del centro hospitalario.
- TELEFONO: teléfono de contacto del centro.
- TELEFONO2: segundo teléfono de contacto del centro.
- TELEFAX: fax del centro.
- CODMU: identificador del municipio al que pertenece.
- CODPROV: identificador de la provincia donde se encuentra el centro.
- CODAUTO: identificador de la comunidad donde se encuentra el centro.
- CODPOSTAL: código postal del centro.
- NCAMAS: número de camas en el hospital.

- CODFI: indentificador de la finalidad asistencial.
- CODPAT: identificador de la dependencia patrimonial.
- CODFU: identificador de la dependencia funcional.
- ACREDOCENT: indica si dispone de acreditación docente.
- CONCIERTO.
- EMAIL: email de contacto con el centro.
- Lat y Long: oordenadas latitud/longitud del centro.

Tipo de entidad CH_DOTACION:

Representa la dotación de la que dispone un centro hospitalario, y para ello, se consideran los siguientes atributos:

- CODID: identificador del centro al que le correponde la dotación.
- TAC: Tomografía Axial Computerizada
- RM: Resonancia Magnética
- GAM: Gammacámara
- HEM: Sala de Hemodinámica
- ASD: Angiografía por Sustracción Digital
- LIT: Litotricia Extracorporea por Ondas de Choque
- BCO: Bomba de Cobalto
- ALI: Acelerador de Partículas
- SPECT: Tomografía por emisión de fotones
- PET: Tomografía por emisión de positrones
- MAMOS: Mamógrafo
- DO: Densitómetros Óseos
- DIAL: Equipos de Hemodiálisis

Tipo de entidad CH_FUNCIONAL:

Representa la dependencia funcional de la que dispone un centro hospitalario, y para ello, se consideran los siguientes atributos:

- CODFU: identificador de la dependencia funcional.
- DESFU: descripción de la dependencia funcional.

Tipo de entidad CH_PATRIMONIAL:

Representa la dependencia patrimonial de la que dispone un centro hospitalario, y para ello, se consideran los siguientes atributos:

- CODPAT: identificador de la dependencia patrimonial.
- DESDE: descripción de la dependencia patrimonial.

Tipo de entidad CH_FINALIDAD:

Representa la finalidad asistencial de la que dispone un centro hospitalario, y para ello, se consideran los siguientes atributos:

- CODFI: identificador de la finalidad asistencial.
- DESFI: descripción de la finalidad asistencial.

Tipo de entidad SIAP_DISP_EXTRAS:

Representa los centros de atención urgente extrahospitalaria. Se compone de los siguientes atributos:

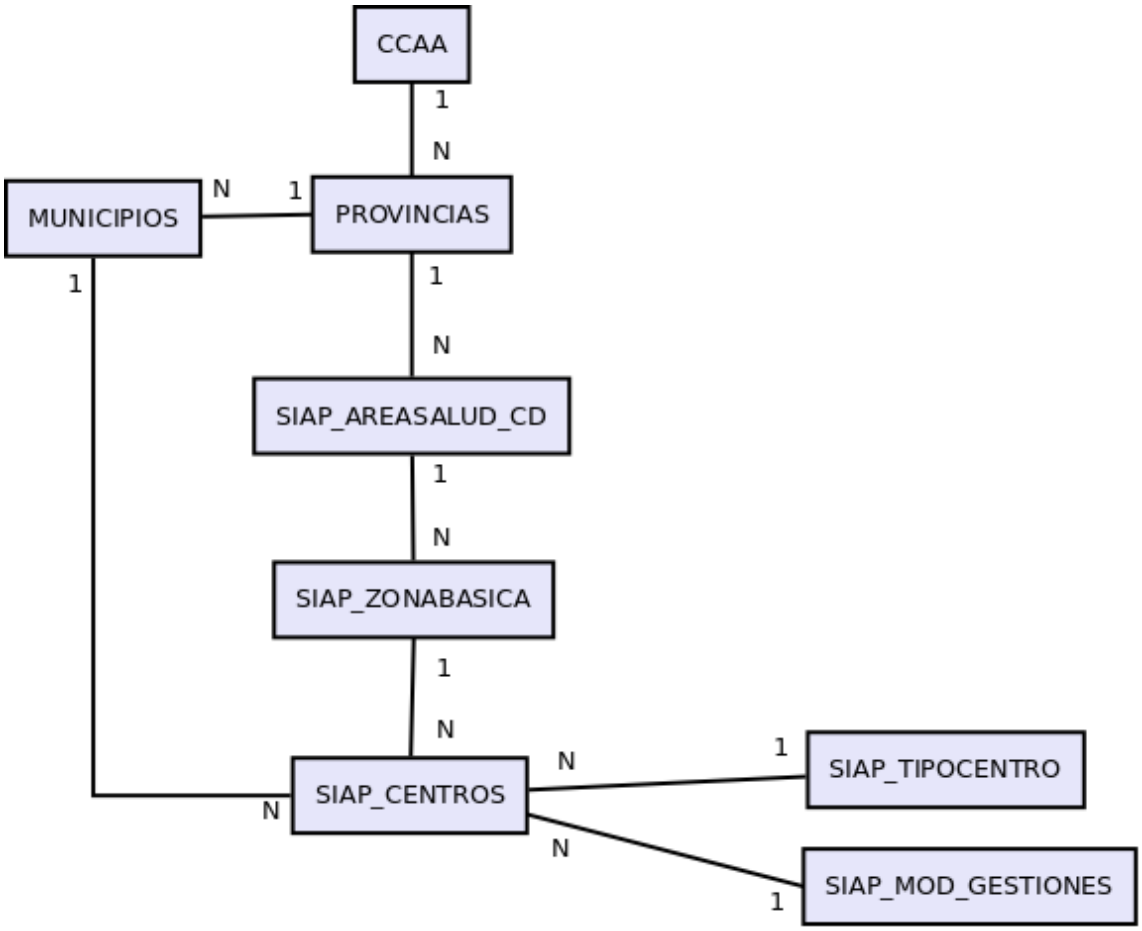
- DISP_EXTRA_ID: identificador del centro.
- T_NOMBRE: tipo de centro.
- T_UBICACION: nombre del centro.
- C_CODPOSTAL: código postal del centro.
- T_LOCALIDAD: localidad del centro.
- C_TELEFONO: teléfono de contacto del centro.
- T_TIPO: abreviatura del tipo de centro.
- T_HORARIO: horario de atención.
- C_CODMU_ID: identificador del municipio al que pertenece.
- T_DISPOSITIVO: tiene el mismo valor que T_NOMBRE
- T_DIRECCION: dirección del centro.
- Lat y Long: oordenadas latitud/longitud del centro.

Esquemas Entidad-Interrelación

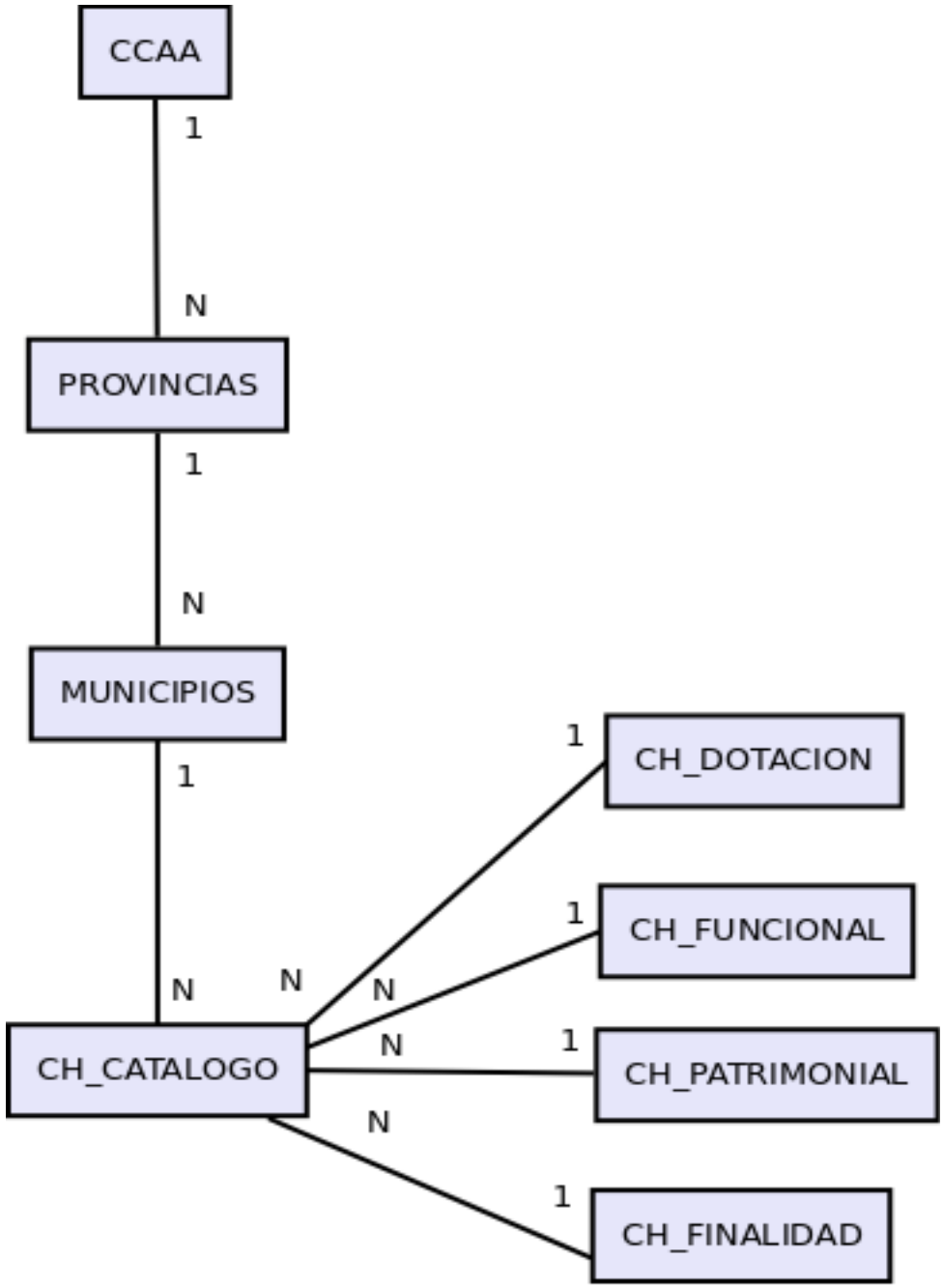
Partiendo de la descripción del dominio de la información comentada anteriormente, se han realizado unos esquemas conceptuales para definir las estructuras de datos que necesitaremos en el sistema.

Los diagramas Entidad-Interrelación que se han obtenido son los siguientes:

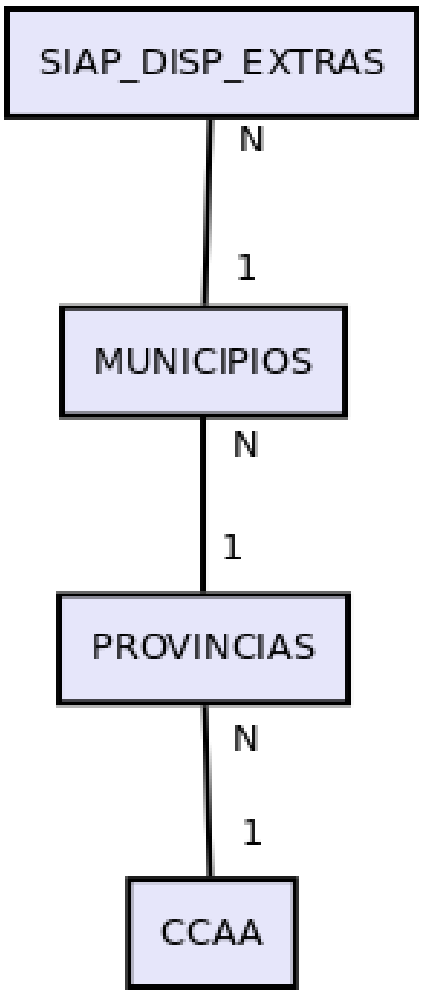
Modelo centros atencíon primaria



Modelo centros hospitalarios



Modelo centros de atención urgente extrahospitalaria



El modelo de datos diseñado se corresponde con el originalmente definido en el Catálogo Nacional de Datos Abiertos, esto nos permitirá realizar las futuras actualizaciones en la base de datos de la aplicación de una forma sencilla.

Como se ha comentado, los datos de latitud y longitud son necesarios para poder geolocalizar a un centro en el mapa. Estos datos no son proporcionados inicialmente en el Catálogo Nacional de Datos Abiertos. Por tano, se han tenido que obtener a través de varios script Python que se adjunta en el **"Anexo I. Scripts Python para generar la codificación geográfica de los centros"**.

Se ha decidido incorporar esta información a la base de datos por dos motivos:

- *Como medida para mejorar el rendimiento*: Debido al volumen de centros con los que vamos a trabajar, y a que se pueden mostrar muchos de ellos como resultado de una búsqueda, se ha estimado conveniente hacer una carga previa de la latitud y longitud de cada centro en la base de datos. De este modo, en vez de ir obteniendo estos datos a partir de la dirección del centro cada vez que se vaya a pintar un

centro en el mapa (en tiempo de ejecución), simplemente se lanzarán consultas contra la base de datos.

- *Solicitudes de codificación geográfica limitadas por día*: Otro motivo no menos importante, es que hay un límite en el número de solicitudes de codificación geográfica por día (15.000). Por tanto, de esta manera evitaremos que un usuario se encuentre en la situación de no poder utilizar la aplicación porque se han agotado las solicitudes de codificación geográfica de ese día.

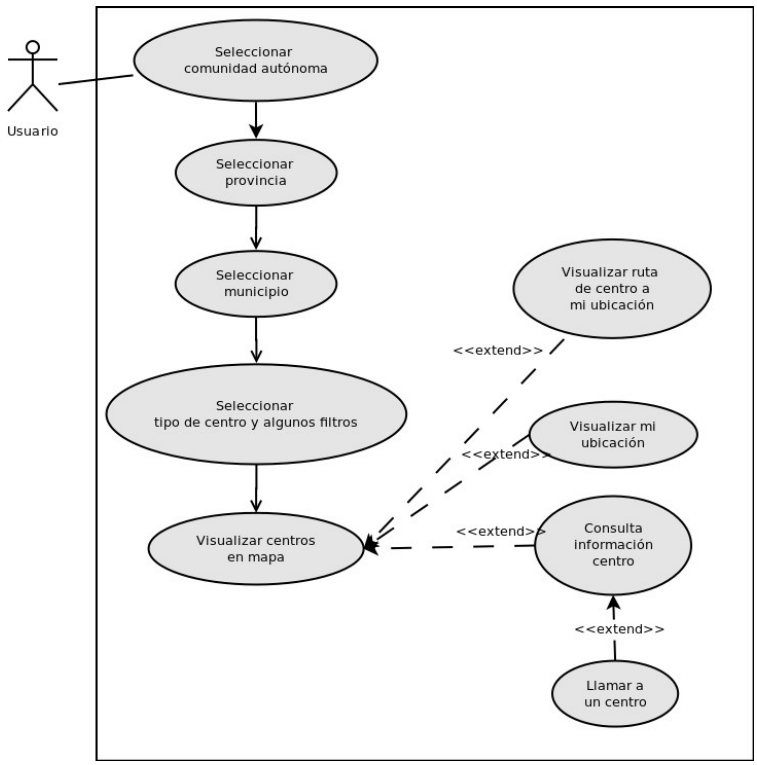
También hay que mencionar que para gestionar esta base de datos se utilizará un sistema de gestión de bases de datos relacional SQLite 3.0.

A diferencia de los sistema de gestión de bases de datos cliente-servidor, el motor de SQLite no es un proceso independiente con el que el programa principal se comunica. En lugar de eso, la biblioteca SQLite se enlaza con el programa pasando a ser parte integral del mismo. El programa utiliza la funcionalidad de SQLite a través de llamadas simples a subrutinas y funciones. Esto reduce la latencia en el acceso a la base de datos, debido a que las llamadas a funciones son más eficientes que la comunicación entre procesos. El conjunto de la base de datos (definiciones, tablas, índices, y los propios datos), son guardados como un sólo fichero estándar en la máquina host.

La provisión de los datos en la base de datos SQLite, se realizará a través de comandos nativos del gestor que permiten importar los datos a través de los ficheros csv obtenidos en el Catálogo Nacional de Datos Abiertos.

4.2 Definición de los casos de uso

A continuación, se desarrollan los distintos casos de uso identificados para la aplicación a desarrollar.



lustración 8: Diagrama de casos de uso

Caso de uso Seleccionar comunidad autónoma

Caso de uso	**Seleccionar comunidad autónoma**
Actor	Usuario final de la aplicación
Descripción	El usuario desea iniciar la aplicación para poder hacer uso de las diferentes opciones que ésta ofrece. Las principales son: ➢ Localizar en el mapa los centros que cumplen los requisitos establecidos por el usuario. ➢ Consultar la información correspondiente a un centro seleccionado. ➢ Realizar una llamada a un centro seleccionado. ➢ Conocer la ruta desde la ubicación del usuario hasta el centro seleccionado.

	Por tanto, para comenzar a usar la aplicación, se le solicita al usuario que seleccione la comunidad autónoma en la que quiere realizar la búsqueda de centros.
Precondiciones	La aplicación debe funcionar correctamente. La aplicación debe dar la opción de elegir una de las comunidades autónomas.
Escenario de éxito principal	La aplicación está disponible para su uso. El usuario decide utilizar la aplicación. El usuario decide seleccionar una comunidad autónoma.
Postcondiciones	Al seleccionar una comunidad autónoma, automáticamente se rellena el filtro de las provincias, con las provincias de la comunidad autónoma elegida.

Caso de uso Seleccionar provincia

Caso de uso	**Seleccionar provincia**
Actor	Usuario final de la aplicación
Descripción	El usuario continua utilizando la aplicación y se le ofrecen la opción de elegir una de las provincias pertenecientes a la comunidad autónoma seleccionada anteriormente.
Precondiciones	✓ La aplicación debe funcionar correctamente. ✓ La aplicación debe dar la opción de elegir una de las comunidades autónomas.
Escenario de éxito principal	✓ La aplicación está disponible para su uso. ✓ El usuario decide utilizar la aplicación. ✓ El usuario decide seleccionar una provincia.
Postcondiciones	Al seleccionar una provincia, automáticamente se rellena el filtro de los municipios, con los municipios de la provincia elegida.

Caso de uso Seleccionar municipio

Caso de uso	**Seleccionar municipio**
Actor	Usuario final de la aplicación
Descripción	El usuario continua utilizando la aplicación y se le ofrecen la opción de elegir uno de los municipios pertenecientes a la provincia seleccionada anteriormente.
Precondiciones	✓ La aplicación debe funcionar correctamente. ✓ El usuario ha seleccionado la provincia donde desea buscar los centros. ✓ La aplicación debe dar la opción de elegir uno de los municipios de la provincia seleccionada anteriormente.
Escenario de éxito principal	✓ La aplicación está disponible para su uso. ✓ El usuario decide utilizar la aplicación. ✓ El usuario decide seleccionar un municipio.
Postcondiciones	No aplica.

Caso de uso Seleccionar tipo de centro y otros filtros

Caso de uso	**Seleccionar tipo de centro y otros filtros**
Actor	Usuario final de la aplicación
Descripción	El usuario continua utilizando la aplicación y puede actuar de varias maneras: ➢ Seleccionar el tipo de centro y otros filtros como por ejemplo la finalidad asistencial. ➢ Volver atrás para seleccionar un municipio distinto.
Precondiciones	✓ La aplicación debe funcionar correctamente. ✓ El usuario ha seleccionado previamente una comunidad autónoma, provincia y un municipio donde desea buscar los

	centros.
Escenario de éxito principal	✓ La aplicación está disponible para su uso. ✓ El usuario decide utilizar la aplicación.
Postcondiciones	No aplica.

Caso de uso Visualizar centros en el mapa

Caso de uso	**Visualizar centros en el mapa**
Actor	Usuario final de la aplicación
Descripción	El usuario sigue adelante en el uso de la aplicación. Tras establecer el tipo de centro que busca y otros filtros con la finalidad asistencial; se le mostrará un mapa con los centros resultantes de la búsqueda marcados en el mapa. Llegado este punto, el usuario tendrá varias opciones: ➢ Mostrar un listado con los centros que se han localizado en el mapa, para que el usuario pueda dirigirse a uno en concreto. ➢ Consultar la información correspondiente a los centros localizados, seleccionando cada uno de ellos. ➢ Visualizar en el mapa la ubicación del usuario, viendo así cuales son los centros más cercanos a su posición.
Precondiciones	✓ La aplicación debe funcionar correctamente. ✓ El usuario ha seleccionado previamente una comunidad autónoma, provincia, y un municipio donde desea buscar los centros. También ha eligido un tipo de centro y opcionalmente la finalidad asistencial siempre y cuando se haya elegido el tipo hospitales. ✓ La aplicación debe dar la opción de localizar los centros resultado de la búsqueda en el mapa.
Escenario de éxito principal	✓ La aplicación está disponible para su uso.

	✓ El usuario decide utilizar la aplicación. ✓ El usuario decide visualizar en el mapa los centros que cumplan con los requisitos que él ha establecido.
Postcondiciones	No aplica.

Caso de uso Visualizar mi ubicación

Caso de uso	**Visualizar mi ubicación**
Actor	Usuario final de la aplicación
Descripción	Una vez el usuario ha llegado al punto de visualizar en pantalla el mapa del municipio seleccionado, con los centros que cumplen los requisitos que ha marcado previamente; tiene la posibilidad de visualizar a la vez su localización. Esto le va a permitir al usuario, comprobar fácilmente cuales son los centros más cercanos a su ubicación.
Precondiciones	✓ La aplicación debe funcionar correctamente. ✓ El usuario ha seleccionado previamente una comunidad autónoma, provincia y un municipio donde desea buscar los centros. ✓ El usuario ha tenido la opción de seleccionar el tipo de centro y otros filtros como la finalidad asistencial, aunque no es obligatorio en el caso de la finalidad asistencial. ✓ El usuario ha decidido visualizar los centros resultantes de la búsqueda en el mapa. ✓ La aplicación debe dar la opción de localizar en el mapa la ubicación del usuario.
Escenario de éxito principal	✓ La aplicación está disponible para su uso. ✓ El usuario decide utilizar la aplicación. ✓ El usuario decide situar su ubicación en el mapa para hacerse una idea de cuales son los centros que tiene más

	cercanos.
Postcondiciones	No aplica.

Caso de uso Consultar información de un centro

Caso de uso	**Consultar información de un centro**
Actor	Usuario final de la aplicación
Descripción	Partimos de que el usuario se encuentra visualizando en pantalla el mapa del municipio seleccionado, con los centros que cumplen los requisitos que él ha establecido. Una de las posibilidades que tiene el usuario es pulsar sobre uno de los centros que tiene en el mapa, y se le abrirá un diálogo con la posibilidad de realizar dos acciones: ➢ Acceder a la información en detalle del centro y poder realizar una llamada telefónica. ➢ Mostrar la ruta desde la ubicación del usuario a dicho centro.
Precondiciones	✓ La aplicación debe funcionar correctamente. ✓ El usuario ha seleccionado previamente una provincia y un municipio donde desea buscar los centros. ✓ El usuario ha tenido la opción de seleccionar el tipo de centro y otros filtros como la finalidad asistencial, aunque no es obligatorio en el caso de la finalidad asistencial. ✓ El usuario ha decidido visualizar los centros resultantes de la búsqueda en el mapa. ✓ La aplicación debe dar la opción de seleccionar cualquiera de los centros que aparecen localizados en el mapa, para consultar toda la información disponible del mismo.
Escenario de éxito principal	✓ La aplicación está disponible para su uso. ✓ El usuario decide utilizar la aplicación. ✓ El usuario decide consultar la información disponible de alguno de los centros localizados en el mapa.

Postcondiciones	No aplica.

Caso de uso Llamar a un centro

Caso de uso	**Llamar a un centro**
Actor	Usuario final de la aplicación
Descripción	Una vez que el usuario se encuentra consultando la información de uno de los centros resultantes de su búsqueda; una de las opciones que se le ofrecen es la de llamar al centro (siempre y cuando, el número de teléfono sea uno de los datos disponibles).
Precondiciones	✓ La aplicación debe funcionar correctamente. ✓ El usuario ha seleccionado previamente una comunidad autónoma, una provincia y un municipio donde desea buscar los centros. ✓ El usuario ha tenido la opción de seleccionar el tipo de centro y otros filtros como la finalidad asistencial. ✓ El usuario ha decidido visualizar los centros resultantes de la búsqueda en el mapa. ✓ El usuario ha pulsado sobre uno de los centros que se le muestran en el mapa para consultar su información. ✓ La aplicación debe dar la opción de realizar una llamada al centro seleccionado, en caso de disponer de su número de teléfono.
Escenario de éxito principal	✓ La aplicación está disponible para su uso. ✓ El usuario decide utilizar la aplicación. ✓ El usuario decide realizar una llamada al centro que ha seleccionado, y se dispone de su número de teléfono.
Postcondiciones	No aplica.

Caso de uso Visualizar ruta de centro a mi ubicación

Caso de uso	**Visualizar ruta de centro a mi ubicación**
Actores	Usuario final de la aplicación
Descripción	Una vez que el usuario ha pulsado sobre uno de los puntos en el mapa, se habilitará un botón que permite visualizar la ruta para llegar desde su ubicación al centro seleccionado.
Precondiciones	✓ La aplicación debe funcionar correctamente. ✓ El usuario ha seleccionado previamente una comunidad autónoma, provincia y un municipio donde desea buscar los centros. ✓ El usuario ha tenido la opción de seleccionar el tipo de centro y otros filtros como la finalidad asistencial. ✓ El usuario ha decidido visualizar los centros resultantes de la búsqueda en el mapa. ✓ El usuario ha pulsado sobre uno de los que se le muestran en el mapa para consultar su información. ✓ La aplicación debe dar la opción de mostrar al usuario la ruta desde su ubicación al centro seleccionado.
Escenario de éxito principal	✓ La aplicación está disponible para su uso. ✓ El usuario decide utilizar la aplicación. ✓ El usuario decide visualizar la ruta para llegar desde su ubicación al centro seleccionado.
Postcondiciones	No aplica.

4.3 Arquitectura general de la aplicación

El objetivo de este apartado es realizar el diseño del sistema, para lo cual, se usarán técnicas de modelado que nos ayudarán a comprender mejor el sistema que estamos desarrollando. Al igual que para analizar el

comportamiento del sistema, para aplicar estas técnicas se usará UML (Lenguaje Unificado de Modelado)[1], ya que nos aporta la sintaxis y semántica necesaria para alcanzar este fin.

Para ello se va a realizar el modelo de paquetes, donde se mostrará la agrupación de los elementos del sistema, pudiendo manejarlos como grupo y facilitando la comprensión del mismo. Esto se representará mediante diagramas de paquetes.

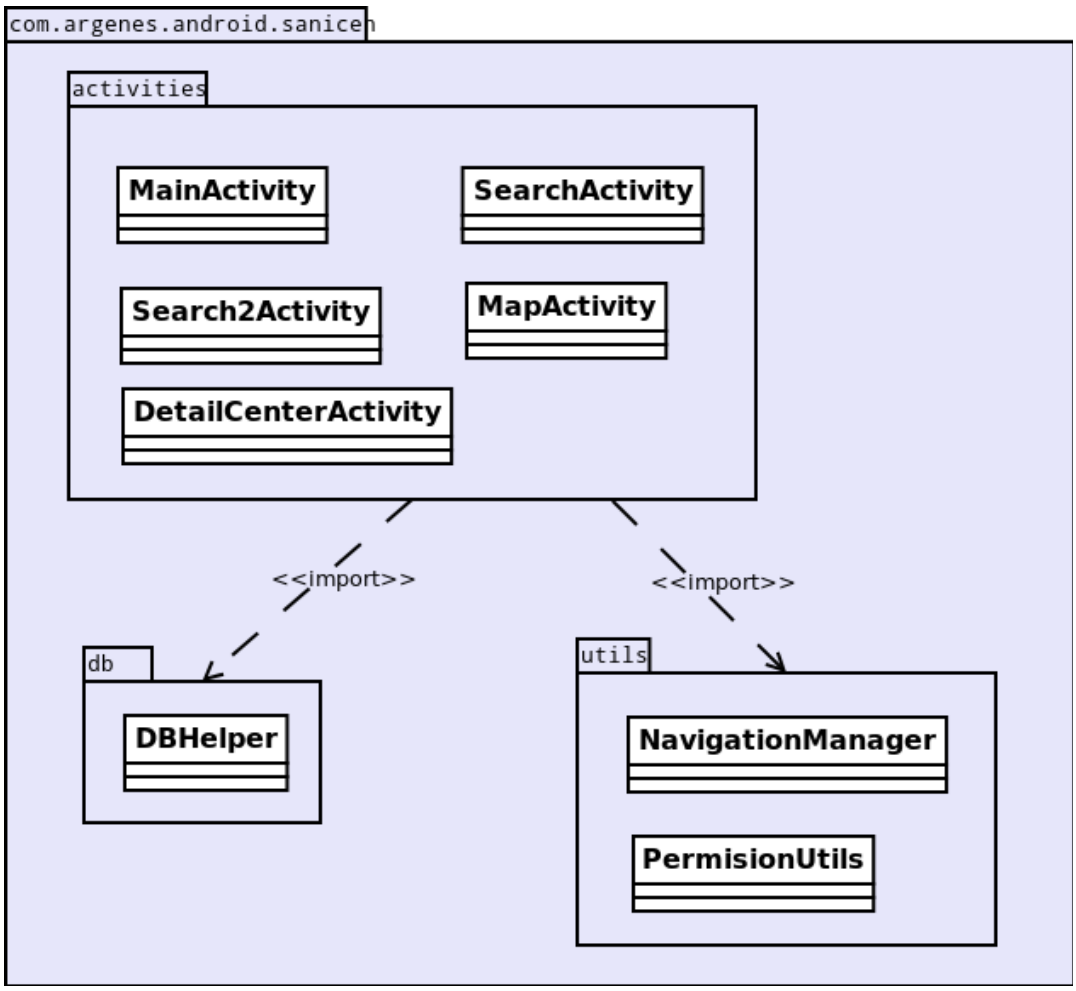


Ilustración 9: Diagrama de paquetes

Paquete raíz com.argenes.android.sanicen

En este paquete raíz de la aplicación, es donde vamos a englobar todos los componentes del sistema. En él encontraremos los paquetes de la aplicación que detallaremos a continuación:

Paquete Activities

Las Activities son el motor fundamental de cualquier aplicación Android.

[1] Consulta bibliográfica (1).

Una Activity es una pieza de código ejecutable que siendo iniciada por el usuario o por el sistema operativo, se ejecuta mientras es necesitada.

La característica principal de un Activity es que puede interactuar con el usuario, aunque también puede pedir datos o servicios de otras activities por medio de queries o Intents.

En la mayoría de los casos cada Activity corresponde a una vista (layout). Esto quiere decir que por cada pantalla debe crearse un Activity, aunque no es requisito obligatorio vincular la parte visual a un Activity. Concretamente los Activity que necesitaremos son los siguientes:

- ***MainActivity***: Actividad principal con la que se inicia la aplicación y donde se muestra una descripción de la aplicación.
- ***SearchActivity***: Actividad que permite al usuario elegir la comunidad autónoma, la provincia, el municipio y tipo de centro.
- ***Search2Activity****:* Actividad que permite al usuario elegir la finalidad asistencial en el caso que haya elegido como tipo de centro hospitales.
- **DetailCenterActivity**: Actividad que muestra en detalle la información de un centro.
- ***MapaActivity***: Esta actividad extiende de MapActivity, la cual encapsula toda la funcionalidad para mostrar un MapView. Todas las actividades anteriores extendían de Activity, pero en este caso necesitamos del MapActivity para poder visualizar el mapa. Esta actividad es la encargada de pintar en el mapa los centros resultantes de la búsqueda de centros sanitarios realizada. También será la que ofrezca al usuario las demás funcionalidades planteadas, como visualizar la ubicación del usuario, y mostrar una ruta al centro que se seleccione del mapa.

Paquete bd

En este paquete encontramos la clase ***DBHelper****,* la cual extiende de *SQLiteOpenHelper*, clase que nos va a permitir crear la base de datos.

A grandes rasgos, la clase *DBHelper* va a ser la que nos va a permitir crear la base de datos, y donde vamos a configurar las consultas que vamos a realizar a dicha base de datos.

Paquete utils

En este paquete encontramos una serie de clases genéricas:

- ***NavigationManager***: Clase que se encarga de gestionar la navegación entre las Activities; es decir, en ella encontraremos los métodos que utilizaremos para pasar de una pantalla a otra de la aplicación, así como de los métodos necesarios para obtener los elementos seleccionados en ellas (comunidades autónomas, provincia, municipios, tipo de centros, etc.).

- ***PermisionUtils***: Clase que se encarga de gestionar la solicitud de permisos al usuario para realizar determinadas acciones como obtener la ubicación actual.

5 PLANIFICACIÓN

Para alcanzar los objetivos marcados para este proyecto, vamos a definir las fases y entregables que se van a ir desarrollando, así como la planificación temporal para cada una de ellas.

5.1 Entregables

PEC1 – Plan de trabajo

En el actual documento se definen las características del proyecto a desarrollar. Se establece el título, una descripción general del proyecto y los objetivos que se pretenden alcanzar.

En base a todo lo anterior, se definirá también una planificación temporal con las metas a conseguir para cada uno de los hitos o puntos de revisión establecidos.

PEC2 – Análisis y diseño de la aplicación. Prototipo

En este documento se realizará un análisis de los requisitos del usuario, así como el diseño de la aplicación.

PEC3 – Desarrollo de la aplicación

En esta entrega se pretende tener prácticamente terminado el desarrollo de la aplicación. Se deben de cumplir para entonces los siguientes objetivos.

- Tener una base de datos con todos los centros sanitarios de España, y con toda la información necesaria para esta aplicación.
- Disponer de una pantalla inicial donde seleccionar la comunidad autónoma, la provincia y municipio a consultar.
- Establecer filtros de búsqueda, para mostrar sólo los centros que cumplan los requisitos deseados.
- Mostrar los centros sanitarios que cumplan las características elegidas, así como la información del centro al pinchar sobre uno de ellos.

Entrega Final

Para este momento se entregará la memoria final del proyecto, así como la aplicación con los últimos objetivos marcados:

- Disponer las opciones de llamar y navegar al centro.

- Pintar la ubicación del usuario, mostrando así los centros más cercanos al usuario.
- Refinamiento o mejoras que se consideren oportunas.

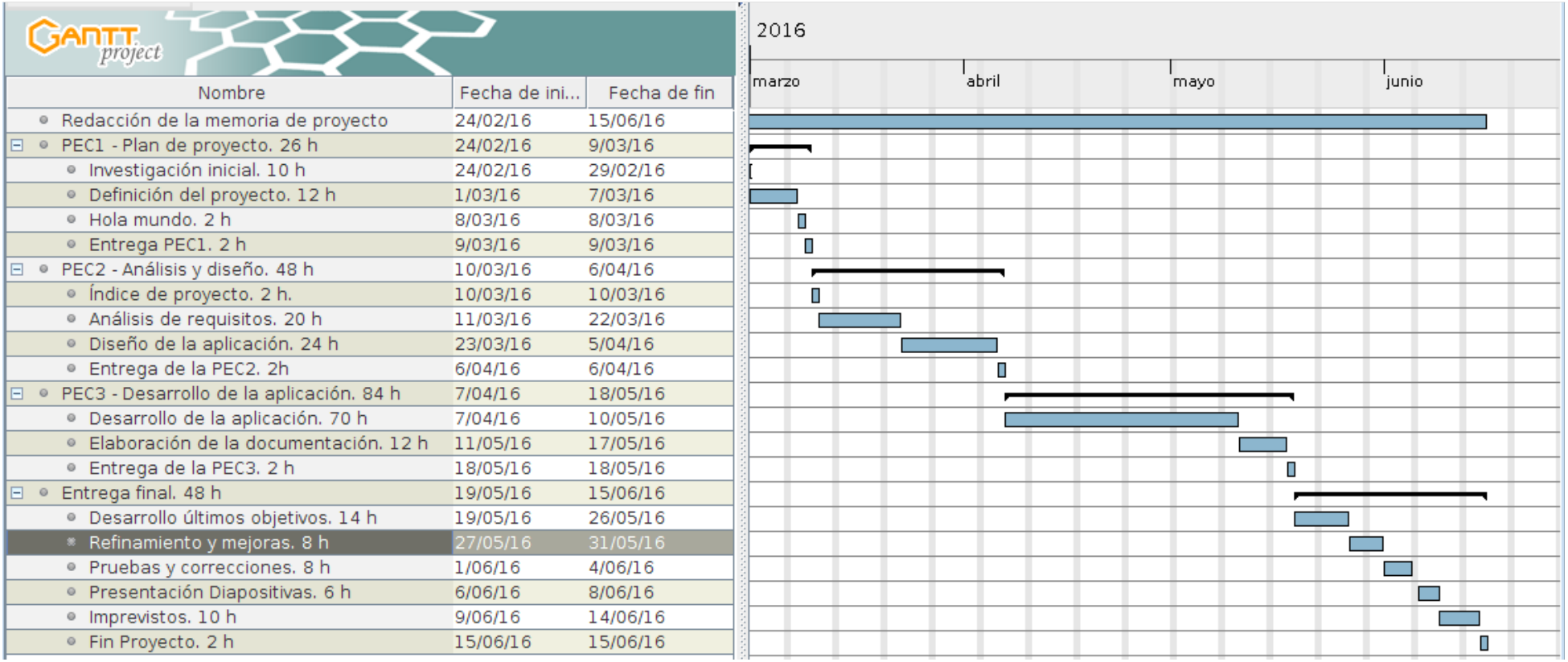
5.2 Planificación temporal

El horario de trabajo durante el desarrollo del proyecto será el siguiente: De lunes a sábado de 19:00 a 21:00.

Además, los entregables anteriormente mencionados se considerarán hitos a la hora de realizar la planificación de las tareas:

- PEC 1 – Presentación del plan de trabajo. Día 9 de marzo de 2016.
- PEC 2 – Análisis y diseño de la aplicación. Día 6 de abril de 2016.
- PEC 3 – Desarrollo de la aplicación . Día 18 de mayo de 2016.
- Entrega final. Día 15 de junio de 2016.

Teniendo en cuenta todo lo anterior, se ha realizado el siguiente diagrama de Gantt donde se plasma la planificación de las tareas (indicando la duración en horas) y actividades que se llevarán a cabo (aunque durante el desarrollo del proyecto puede sufrir algún cambio):



Nombre	Fecha de ini...	Fecha de fin
Redacción de la memoria de proyecto	24/02/16	15/06/16
PEC1 - Plan de proyecto. 26 h	24/02/16	9/03/16
Investigación inicial. 10 h	24/02/16	29/02/16
Definición del proyecto. 12 h	1/03/16	7/03/16
Hola mundo. 2 h	8/03/16	8/03/16
Entrega PEC1. 2 h	9/03/16	9/03/16
PEC2 - Análisis y diseño. 48 h	10/03/16	6/04/16
Índice de proyecto. 2 h.	10/03/16	10/03/16
Análisis de requisitos. 20 h	11/03/16	22/03/16
Diseño de la aplicación. 24 h	23/03/16	5/04/16
Entrega de la PEC2. 2h	6/04/16	6/04/16
PEC3 - Desarrollo de la aplicación. 84 h	7/04/16	18/05/16
Desarrollo de la aplicación. 70 h	7/04/16	10/05/16
Elaboración de la documentación. 12 h	11/05/16	17/05/16
Entrega de la PEC3. 2 h	18/05/16	18/05/16
Entrega final. 48 h	19/05/16	15/06/16
Desarrollo últimos objetivos. 14 h	19/05/16	26/05/16
Refinamiento y mejoras. 8 h	27/05/16	31/05/16
Pruebas y correcciones. 8 h	1/06/16	4/06/16
Presentación Diapositivas. 6 h	6/06/16	8/06/16
Imprevistos. 10 h	9/06/16	14/06/16
Fin Proyecto. 2 h	15/06/16	15/06/16

6 PLAN DE PRUEBAS

A continuación se describen las pruebas funcionales realizadas sobre el proyecto:

Nombre	Descripción
Pantalla principal	Comprobar la maquetación y el correcto funcionamiento del botón para comenzar a usar la aplicación.
Listado de comunidades autónomas	Comprobar que el listado de comunidades autónomas se muestra correctamente en contenido y maquetación.
Listado de provincias	Comprobar que el listado de provincias se muestra correctamente en contenido y maquetación.
Listado de municipios	Comprobar que el listado de municipios se muestra correctamente en cuanto a maquetación, y que los municipios mostrados corresponden a la provincia seleccionada.
Listado de tipo centros	Comprobar que el listado de tipo de centros se muestra correctamente en contenido y maquetación.
Listado de finalidades asistenciales	Comprobar que el listado de finalidades asistenciales que puede ofrecer un centro se muestra correctamente en contenido y maquetación.
Navegación entre pantallas	Comprobar que se pasa correctamente a la siguiente pantalla (desde cualquiera de ellas), y que se puede retroceder también a la pantalla previa, visualizando la información marcada con anterioridad si fuera el caso.
Localización de centros resultantes de la búsqueda	Comprobar que los centros mostrados en el mapa corresponden a los criterios de búsqueda establecidos y que su localización es correcta.
Mostrar “mi ubicación”	Comprobar que se muestra la ubicación del usuario correctamente, y en caso de no ser posible, comprobar que se muestra un mensaje informativo.
Mostrar información de un centro seleccionado	Comprobar que al seleccionar cualquiera de los centros mostrados en el mapa, se muestra la información correspondiente al mismo. Comprobar la maquetación de la vista detalle de un centro.
Llamar a un centro	Comprobar que si se tiene disponible el teléfono del centro seleccionado, se establece una llamada con el mismo correctamente.
Mostrar ruta de “mi ubicación” a	Comprobar que se muestra correctamente la ruta desde la ubicación del usuario a un centro

un centro	seleccionado. En caso de no estar disponible la ubicación del usuario, comprobar que se muestra un mensaje informativo.
Visualización del mapa	Comprobar que el zoom del mapa se ajusta automáticamente para visualizar en pantalla todos los centros resultantes de la búsqueda. Comprobar también que se puede modificar el zoom de la pantalla manualmente.

Se ha llevado a cabo el anterior plan de pruebas y se ha superado con éxito.

7 LICENCIA Y CRÉDITOS DE LOS RECURSOS GRÁFICOS

La licencia que se ha decidido utilizar para este proyecto es la licencia GNU GPLv2.

La Licencia Pública General de GNU o más conocida por su nombre en inglés GNU General Public License (GNU GPL), es una licencia creada por la Free Software Foundation en 1989 (la primera versión, escrita por Richard Stallman), y está orientada principalmente a proteger la libre distribución, modificación y uso de software. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

Esta licencia garantiza una serie de libertades al usuario, como el libre uso y adaptación del código fuente del programa, obligando a que las modificaciones de este código estén disponibles para el resto de usuarios si se redistribuye el programa.

Esta licencia se usa para la mayoría de los programas de GNU y para más de la mitad de los paquetes de software libre.

La versión, GNU GPLv2, fue publicada en 1991 y es accesible a través del Portal de GNU[2].

En cuanto a los iconos utilizados en la aplicación, se han hecho uso de recursos con licencia libre. A continuación se muestran las distintas imágenes utilizadas con su licencia correspondiente y una referencia a su autor original.

[2] http://www.gnu.org/licenses/gpl-2.0.html

Icono original:	
Icono usado:	
Icon set:	Cross, doctor, health
Autor:	Dod Cosmin
Licencia:	Creative Commons Attribution 3.0
Referencia:	https://www.iconfinder.com
Uso:	Icono de la aplicación. Icono utilizado como icono de la aplicación sanicen.

Icono original:	
Iconos usados	  
Icon set:	Medical Elements
Autor:	Freepik in signs
Licencia:	Free license (with attribution)
Referencia:	http://www.flaticon.com/
Uso:	Marcadores de los centros en el mapa.

8 CONCLUSIONES Y FUTURAS MEJORAS

El trabajo desarrollado en este proyecto ha cumplido los objetivos definidos en el punto 1.3 Objetivos del proyecto.

Este proyecto ha sido desarrollado durante un semestre y se ha ajustado a la planificación establecida en el punto 5. Planificación, a excepción del tratamiento de la información procedente del Catálogo Nacional de Datos Abiertos que ha llevado más tiempo del esperado.

Aunque los objetivos del proyecto se hayan cumplido, se proponen una serie de posibles ampliaciones que enriquecerían la funcionalidad del sistema desarrollado:

- Se podría dar la posibilidad al usuario de guardar una búsqueda realizada, para poder consultarla nuevamente sin necesidad de tener que pasar por las pantallas de selección, sino directamente desde la pantalla de inicio.
- Otra posibilidad para guardar centros ya consultados y que le interesan al usuario, sería poder marcarlos como favoritos, y dar la opción desde la pantalla inicial de visualizar directamente los centros señalados como favoritos.
- Mejorar el formulario de búsqueda para no tener obligatoriamente que seleccionar comunidad autónoma, provincia y municipio, así se podrían mostrar todos los centros del país, o una comunidad autónoma o una provincia. Igualmente el tipo de centro podría ser opcional y mezclar los tres tipos en los resultados de búsqueda.
- Añadir más botones de acceso directo de resultados en la vista principal, por ejemplo obtener directamente los centros más cercanos a mi ubicación.
- Poder refinar los resultados de búsqueda sobre el mapa.
- También se puede trabajar en la apariencia de la aplicación, especialmente en la vista de detalle de un centro.

BIBLIOGRAFÍA

Consultas bibliográficas

(1) Booch, G.; Rumbaugh, J. Et al., El Lenguaje Unificado de Modelado, Addison Wesley, 1999. 464 p. ISBN: 84-7829-028-1.

(2) Pressman, R., Ingeniería del Software: Un Enfoque Práctico, 4a Edición, 1997. 581 p. ISBN: 84-481-1186-9.

(3) Luque Ruiz, I.; Gómez-Nieto, M. A., Ingeniería del Software: Fundamentos para el desarrollo de sistemas informáticos. Servicio de

Publicaciones de la Universidad de Córdoba. 1999. 304 p. ISBN: 84-7801-486-1.

Consultas web

(a) Curso Programación Android: Aprende a crear tus propias aplicaciones Android.

- http://www.sgoliver.net/blog/?page_id=2935

(b) Programación Android

- http://www.nosinmiubuntu.com/p/programacion-android.html
- Última actualización: Febrero de 2015.

(c) Documentación de la API de Google Maps

- https://developers.google.com/maps/documentation/android/
- Última actualización: Marzo 2016.

(d) Patrones de diseño para aplicaciones Android

- http://www.androidpatterns.com/

(e) Ejemplos android

- https://github.com/googlemaps/android-samples

(f) Artículos varios sobre programación en Android

- http://androideity.com/category/desarrollo/
- Última actualización: Febrero 2016.

ANEXO I. SCRIPTS PYTHON PARA GENERAR LA CODIFICACIÓN GEOGRÁFICA DE LOS CENTROS.

Localización de centros de atención primaria

```
# -*- coding: utf-8 -*-

import csv

from geopy import geocoders

# Partimos de este fichero, donde tenemos todos los centros con su dirección
CSV_CENTERS_FILE = 'siap_centros.csv'
CSV_MUNICIPALTIES_FILE = 'ch_municipios.csv'
CSV_PROVINCES_FILE = 'siap_provincias.csv'
CSV_CCAA_FILE = 'siap_ccaa.csv'

# Los centros localizados los iremos volcando a este fichero, ya con dos columnas más,
# una para latitud y otra para longitud.
CSV_OUTPUT_FILE = 'centros_localizados.csv'
CSV_LOCATED = CSV_OUTPUT_FILE

# En caso de no conseguir localizar a un centro, lo pasaremos a este fichero, para realizar
# posteriormente una búsqueda manual del mismo.
CSV_NO_LOCATED = 'centros_no_localizados.csv'

# Clave para poder utilizar las API de Google Maps
GOOGLE_MAPS_API_KEY = <your_api_key>

if __name__ == "__main__":
    g = geocoders.GoogleV3(GOOGLE_MAPS_API_KEY)

    # Cargamos municipios en un diccionario
    csv_municipalties_file = open(CSV_MUNICIPALTIES_FILE, 'r')
    reader = csv.DictReader(csv_municipalties_file, delimiter=';', restval=None, quotechar="")
      municipalties = {row['CODMU']: {'name': row['MUNICIPIO'], 'cod_prov': row['CODPROV']} for row in
reader}
    csv_municipalties_file.close()


    # Cargamos provincias en un diccionario
    csv_provinces_file = open(CSV_PROVINCES_FILE, 'r')
    reader = csv.DictReader(csv_provinces_file, delimiter=';', restval=None, quotechar="")
    provinces = {row['IDPROV']: {'name': row['NOMBRE'], 'cod_ca': row['IDCCAA']} for row in reader}
    csv_provinces_file.close()

    # Cargamos comunidades autónomas en un diccionario
    csv_ccaa_file = open(CSV_CCAA_FILE, 'r')
    reader = csv.DictReader(csv_ccaa_file, delimiter=';', restval=None, quotechar="")
    ccaa = {row['ID']: {'name': row['NOMBRE']} for row in reader}
    csv_ccaa_file.close()

    # Cargamos los centros ya localizados en un diccionario
    csv_located_file = open(CSV_LOCATED, 'r')
    reader = csv.DictReader(csv_located_file, delimiter=';', restval=None, quotechar="")
    centers_located = {row['IDCENTRO']: {'name': row['NOMBRE']} for row in reader}
    csv_located_file.close()

    # Abrimos para lectura el fichero donde tenemos cargados todos los centros
    csv_centers_file = open(CSV_CENTERS_FILE, 'r')
```

```
    reader = csv.DictReader(csv_centers_file, delimiter=';', restval=None, quotechar='"')

    # Abrimos para escritura el fichero donde volcaremos los centros con la latitud y longitud calculada
    csv_output_file = open(CSV_OUTPUT_FILE, 'a')
    writer = csv.DictWriter(csv_output_file, delimiter=';',
                            fieldnames=reader.fieldnames + ['lat', 'long'], quotechar='"')

      # Abrimos para escritura el fichero donde volcaremos los centros no localizados, para buscarlos manualmente
    csv_no_located_file = open(CSV_NO_LOCATED, 'w')
    writer_nolocated = csv.DictWriter(csv_no_located_file, delimiter=';',
                            fieldnames=reader.fieldnames, quotechar='"')

    writer.writeheader()
    no_localizados = []

    # Recorremos cada centro del fichero de entrada
    for row in reader:
        # Reemplazamos algunas abreviaturas para evitar errores
        if row['IDCENTRO'] in centers_located:
            print "Centro %s ya localizado" % row['IDCENTRO']
            continue
                    address = row['DIRECCION'].replace('nº', '').replace('C/', 'calle').replace('Avda.', 'avenida').replace('Ctra.', 'carretera').replace('Bda.', 'barriada').replace('Urb.', 'urbanización').replace('s/n', '')
        postal_code = row['CP']
        municipalty = municipalties[row['CODMU']]['name']
        cod_prov = municipalties[row['CODMU']]['cod_prov']
        province = provinces[cod_prov]['name']
        #cod_ca = provinces[cod_prov]['cod_ca']
        #ca = ccaa[cod_ca]['name']

        # Formamos la consulta que vamos a lanzar con los datos que tenemos del centro
        query = "%s, %s, %s, %s España" % (address, postal_code, municipalty, province)
        lat, lon = None, None
        try:
            # Lanzamos la consulta
            geocodes = g.geocode(query, exactly_one=False)
            place_google, (lat, lon) = geocodes[0]
            print "Localizado: %s: %s %s" % (query, lat, lon)
        except Exception as e:
            # Si no se ha encontrado, lo escribimos en el fichero de no localizados
            print "error: %s" % query, e
            if "The given key has gone over the requests limit" in e.message:
                break
            no_localizados.append(address)
            writer_nolocated.writerow(row)
        if lat and lon:
            # Si tenemos latitud y longitud, lo escribimos en el fichero de centros localizados
            data = {'lat': lat, 'long': lon}
            data.update(row)
            writer.writerow(data)

    # Mostramos un resumen de lo sucedido
    print "Resumen de no localizados"
    print "%d centros no localizados:" % len(no_localizados)
    print no_localizados

    # Cerramos los ficheros
    csv_centers_file.close()
    csv_output_file.close()
    csv_no_located_file.close()
```

Localización de centros de atención urgente extrahospitalaria

```
# -*- coding: utf-8 -*-

import csv

from geopy import geocoders

# Partimos de este fichero, donde tenemos todos los centros con su dirección
CSV_CENTERS_FILE = 'siap_disp_extras.csv'
CSV_MUNICIPALTIES_FILE = 'ch_municipios.csv'
CSV_PROVINCES_FILE = 'siap_provincias.csv'
CSV_CCAA_FILE = 'siap_ccaa.csv'


# Los centros localizados los iremos volcando a este fichero, ya con dos columnas más,
# una para latitud y otra para longitud.
CSV_OUTPUT_FILE = 'centros_localizados.csv'
CSV_LOCATED = CSV_OUTPUT_FILE

# En caso de no conseguir localizar a un centro, lo pasaremos a este fichero, para realizar
# posteriormente una búsqueda manual del mismo.
CSV_NO_LOCATED = 'centros_no_localizados.csv'

# Clave para poder utilizar las API de Google Maps

GOOGLE_MAPS_API_KEY = <your_api_key>


if __name__ == "__main__":
    g = geocoders.GoogleV3(GOOGLE_MAPS_API_KEY)

    # Cargamos municipios en un diccionario
    csv_municipalties_file = open(CSV_MUNICIPALTIES_FILE, 'r')
    reader = csv.DictReader(csv_municipalties_file, delimiter=';', restval=None, quotechar='"')
      municipalties = {row['CODMU']: {'name': row['MUNICIPIO'], 'cod_prov': row['CODPROV']} for row in
reader}
    csv_municipalties_file.close()

    # Cargamos provincias en un diccionario
    csv_provinces_file = open(CSV_PROVINCES_FILE, 'r')
    reader = csv.DictReader(csv_provinces_file, delimiter=';', restval=None, quotechar='"')
    provinces = {row['IDPROV']: {'name': row['NOMBRE'], 'cod_ca': row['IDCCAA']} for row in reader}
    csv_provinces_file.close()

    # Cargamos comunidades autónomas en un diccionario
    csv_ccaa_file = open(CSV_CCAA_FILE, 'r')
    reader = csv.DictReader(csv_ccaa_file, delimiter=';', restval=None, quotechar='"')
    ccaa = {row['ID']: {'name': row['NOMBRE']} for row in reader}
    csv_ccaa_file.close()

    # Cargamos los centros ya localizados en un diccionario
    centers_located = {}
    csv_located_file = open(CSV_LOCATED, 'r')
    reader = csv.DictReader(csv_located_file, delimiter=';', restval=None, quotechar='"')
    centers_located = {row['DISP_EXTRA_ID']: {'name': row['T_UBICACION']} for row in reader}
    csv_located_file.close()

    # Abrimos para lectura el fichero donde tenemos cargados todos los centros
    csv_centers_file = open(CSV_CENTERS_FILE, 'r')
    reader = csv.DictReader(csv_centers_file, delimiter=';', restval=None, quotechar='"')

    # Abrimos para escritura el fichero donde volcaremos los centros con la latitud y longitud calculada
    csv_output_file = open(CSV_OUTPUT_FILE, 'a')
    writer = csv.DictWriter(csv_output_file, delimiter=';',
                            fieldnames=reader.fieldnames + ['lat', 'long'], quotechar='"')
```

```
    # Abrimos para escritura el fichero donde volcaremos los centros no localizados, para buscarlos manualmente
    csv_no_located_file = open(CSV_NO_LOCATED, 'w')
    writer_nolocated = csv.DictWriter(csv_no_located_file, delimiter=';',
                        fieldnames=reader.fieldnames, quotechar='"')

    writer.writeheader()
    no_localizados = []

    # Recorremos cada centro del fichero de entrada
    for row in reader:
        # Reemplazamos algunas abreviaturas para evitar errores
        if row['DISP_EXTRA_ID'] in centers_located:
            print "Centro %s ya localizado" % row['DISP_EXTRA_ID']
            continue
                        address = row['T_DIRECCION'].replace('nº', '').replace('C/', 'calle').replace('Avda.', 'avenida').replace('Ctra.', 'carretera').replace('Bda.', 'barriada').replace('Urb.', 'urbanización').replace('s/n', '')
        postal_code = row['C_CODPOSTAL']
        municipalty = municipalties[row['C_CODMU_ID']]['name']
        cod_prov = municipalties[row['C_CODMU_ID']]['cod_prov']
        province = provinces[cod_prov]['name']

        # Formamos la consulta que vamos a lanzar con los datos que tenemos del centro
        query = "%s, %s, %s, %s España" % (address, postal_code, municipalty, province)
        lat, lon = None, None
        try:
            # Lanzamos la consulta
            geocodes = g.geocode(query, exactly_one=False)
            place_google, (lat, lon) = geocodes[0]
            print "Localizado: %s: %s %s" % (query, lat, lon)
        except Exception as e:
            # Si no se ha encontrado, lo escribimos en el fichero de no localizados
            print "error: %s" % query, e
            if "The given key has gone over the requests limit" in e.message:
                break
            no_localizados.append(address)
            writer_nolocated.writerow(row)
        if lat and lon:
            # Si tenemos latitud y longitud, lo escribimos en el fichero de centros localizados
            data = {'lat': lat, 'long': lon}
            data.update(row)
            writer.writerow(data)

    # Mostramos un resumen de lo sucedido
    print "Resumen de no localizados"
    print "%d centros no localizados:" % len(no_localizados)
    print no_localizados

    # Cerramos los ficheros
    csv_centers_file.close()
    csv_output_file.close()
    csv_no_located_file.close()
```

Localización de hospitales

```
# -*- coding: utf-8 -*-

import csv

from geopy import geocoders

# Partimos de este fichero, donde tenemos todos los centros con su dirección
CSV_CENTERS_FILE = 'ch_catalogo.csv'
CSV_MUNICIPALTIES_FILE = 'ch_municipios.csv'
CSV_PROVINCES_FILE = 'ch_provincias.csv'
CSV_CCAA_FILE = 'ch_ccaa.csv'


# Los centros localizados los iremos volcando a este fichero, ya con dos columnas más,
# una para latitud y otra para longitud.
CSV_OUTPUT_FILE = 'centros_localizados.csv'
CSV_LOCATED = CSV_OUTPUT_FILE

# En caso de no conseguir localizar a un centro, lo pasaremos a este fichero, para realizar
# posteriormente una búsqueda manual del mismo.
CSV_NO_LOCATED = 'centros_no_localizados.csv'

# Clave para poder utilizar las API de Google Maps

GOOGLE_MAPS_API_KEY = <your_api_key>


if __name__ == "__main__":
    g = geocoders.GoogleV3(GOOGLE_MAPS_API_KEY)

    # Cargamos municipios en un diccionario
    csv_municipalties_file = open(CSV_MUNICIPALTIES_FILE, 'r')
    reader = csv.DictReader(csv_municipalties_file, delimiter=';', restval=None, quotechar="")
    municipalties = {row['CODMU']: {'name': row['municipio'], 'cod_prov': row['codprov']} for row in reader}
    csv_municipalties_file.close()

    # Cargamos provincias en un diccionario
    csv_provinces_file = open(CSV_PROVINCES_FILE, 'r')
    reader = csv.DictReader(csv_provinces_file, delimiter=';', restval=None, quotechar="")
    provinces = {row['CODPROV']: {'name': row['NOMBRE'], 'cod_ca': row['CODAUTO']} for row in reader}
    csv_provinces_file.close()

    # Cargamos comunidades autónomas en un diccionario
    csv_ccaa_file = open(CSV_CCAA_FILE, 'r')
    reader = csv.DictReader(csv_ccaa_file, delimiter=';', restval=None, quotechar="")
    ccaa = {row['CODAUTO']: {'name': row['AUTONOMIA']} for row in reader}
    csv_ccaa_file.close()

    # Cargamos los centros ya localizados en un diccionario
    centers_located = {}
    csv_located_file = open(CSV_LOCATED, 'r')
    reader = csv.DictReader(csv_located_file, delimiter=';', restval=None, quotechar="")
    centers_located = {row['CODID']: {'name': row['NOMBRE']} for row in reader}
    csv_located_file.close()

    # Abrimos para lectura el fichero donde tenemos cargados todos los centros
    csv_centers_file = open(CSV_CENTERS_FILE, 'r')
    reader = csv.DictReader(csv_centers_file, delimiter=';', restval=None, quotechar="")

    # Abrimos para escritura el fichero donde volcaremos los centros con la latitud y longitud calculada
    csv_output_file = open(CSV_OUTPUT_FILE, 'a')
    writer = csv.DictWriter(csv_output_file, delimiter=';',
                            fieldnames=reader.fieldnames + ['lat', 'long'], quotechar="")
```

```
    # Abrimos para escritura el fichero donde volcaremos los centros no localizados, para buscarlos manualmente
    csv_no_located_file = open(CSV_NO_LOCATED, 'w')
    writer_nolocated = csv.DictWriter(csv_no_located_file, delimiter=';',
                        fieldnames=reader.fieldnames, quotechar='"')

    writer.writeheader()
    no_localizados = []

    # Recorremos cada centro del fichero de entrada
    for row in reader:
        # Reemplazamos algunas abreviaturas para evitar errores
        if row['CODID'] in centers_located:
            print "Centro %s ya localizado" % row['CODID']
            continue
                        address = row['DIRECCION'].replace('nº', '').replace('C/', 'calle').replace('Avda.', 'avenida').replace('Ctra.', 'carretera').replace('Bda.', 'barriada').replace('Urb.', 'urbanización').replace('s/n', '')
        postal_code = row['CODPOSTAL']
        municipalty = municipalties[row['CODMU']]['name']
        cod_prov = municipalties[row['CODMU']]['cod_prov']
        province = provinces[cod_prov]['name']
        #cod_ca = provinces[cod_prov]['cod_ca']
        #ca = ccaa[cod_ca]['name']

        # Formamos la consulta que vamos a lanzar con los datos que tenemos del centro
        query = "%s, %s, %s, %s España" % (address, postal_code, municipalty, province)
        lat, lon = None, None
        try:
            # Lanzamos la consulta
            geocodes = g.geocode(query, exactly_one=False)
            place_google, (lat, lon) = geocodes[0]
            print "Localizado: %s: %s %s" % (query, lat, lon)
        except Exception as e:
            # Si no se ha encontrado, lo escribimos en el fichero de no localizados
            print "error: %s" % query, e
            if "The given key has gone over the requests limit" in e.message:
                break
            no_localizados.append(address)
            writer_nolocated.writerow(row)
        if lat and lon:
            # Si tenemos latitud y longitud, lo escribimos en el fichero de centros localizados
            data = {'lat': lat, 'long': lon}
            data.update(row)
            writer.writerow(data)

    # Mostramos un resumen de lo sucedido
    print "Resumen de no localizados"
    print "%d centros no localizados:" % len(no_localizados)
    print no_localizados

    # Cerramos los ficheros
    csv_centers_file.close()
    csv_output_file.close()
    csv_no_located_file.close()
```

ANEXO II. ESTRUCTURA DE UN PROYECTO ANDROID.

Para comprender como se construye una aplicación Android vamos a detallar cual es la estructura de carpetas básica que lo compone.

Enumeremos aquellos puntos clave que nos interesa conocer:

Carpeta *src*

Esta es por defecto la carpeta donde se depositará el código fuente Java: clases principales, auxiliares, actividades, servicio, etc.

Todo el código que pongamos en esta carpeta será compilado cuando se requiera. También, como en los proyectos tradicionales de java, el código se organizará en carpetas que resultaran en paquetes.

Carpeta *res*

Esta es una carpeta más compleja ya que se subdivide en múltiples subdirectorios. Esta carpeta contiene los recursos necesarios para el proyecto: imágenes, vídeos, sonidos, cadenas de texto, vistas, etc. Los subdirectorios en los que podemos distribuir los recursos son:

- **Drawable: Contiene las imágenes [y otros elementos gráficos] usados en por la aplicación. Para definir diferentes recursos dependiendo de la resolución y densidad de la pantalla del dispositivo se suele dividir en varias subcarpetas:**
  - drawable-ldpi (densidad baja)
  - drawable-mdpi (densidad media)
  - drawable-hdpi (densidad alta)
  - drawable-xhdpi (densidad muy alta)
- Layout: Contiene los ficheros de definición XML de las diferentes pantallas de la interfaz gráfica.
- Menu: Contiene la definición XML de los menús de la aplicación.
- Values: Contiene otros ficheros XML de recursos de la aplicación, como por ejemplo cadenas de texto (*strings.xml*), estilos (*styles.xml*), colores (*colors.xml*), arrays de valores (*arrays.xml*), etc.

Carpeta *build*

Contiene una serie de elementos de código generados automáticamente al compilar el proyecto. Cada vez que generamos nuestro proyecto, la maquinaria de compilación de Android genera por nosotros una serie de ficheros fuente java dirigidos al control de los recursos de la aplicación.

Carpeta assets

Contiene todos los demás ficheros auxiliares necesarios para la aplicación (y que se incluirán en su propio paquete), como por ejemplo ficheros de configuración, base de datos, etc.

La diferencia entre los recursos incluidos en la carpeta *res* y los incluidos en la carpeta *assets* es que para los primeros se generará un ID en la clase R y se deberá acceder a ellos con los diferentes métodos de acceso a recursos. Para los segundos sin embargo no se generarán ID y se podrá acceder a ellos por su ruta como a cualquier otro fichero del sistema. Usaremos uno u otro según las necesidades de nuestra aplicación.

Carpeta libs

Contendrá las librerías auxiliares, normalmente en formato ".jar" que utilicemos en nuestra aplicación Android.

Fichero AndroidManifest.xml

Es importante e imprescindible y describe como se empaquetará la aplicación. Contiene la definición en XML de los aspectos principales de la aplicación, como por ejemplo su identificación (nombre, versión, icono, …), sus componentes (pantallas, mensajes, …), las librerías auxiliares utilizadas, o los permisos necesarios para su ejecución.