



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA



Escola Tècnica
Superior d'Enginyeria
Informàtica

Escuela Técnica Superior de Ingeniería Informática
Universitat Politècnica de València

Utilización de servicios Web en dispositivos móviles Android

Proyecto Final de Carrera
Ingeniería Superior en Informática

Autor: Mohamed Amine, Sraïri

Director: Juan Sánchez Díaz

Valencia, Marzo 2014

Índice

Agradecimiento	Pág. 3
Capítulo 1: Introducción	
1.1. Presentación del Proyecto	Pág. 4
1.2. Ámbito General	Pág. 4
1.3. Objetivos del Proyecto	Pág. 5
1.4. Solución Propuesta	Pág. 6
Capítulo 2: Contexto Tecnológico	
2.1. La Plataforma Android	Pág. 7
2.2. Geolocalización	Pág. 8
2.3. Servicios Web	Pág. 9
2.4. Aplicación Android	Pág. 9
2.5. Integración	Pág. 10
2.6. Conclusiones	Pág. 11
Capítulo 3: Descripción del Proyecto	
3.1. Descripción de los Requerimientos	Pág. 12
3.2. Análisis de los requerimientos	Pág. 13
3.2.1. Diagrama de clases	Pág. 13
3.2.2. Casos de uso	Pág. 14
Capítulo 4: Aplicación Android	
4.1. Estructura y componentes	Pág. 18
4.2. Actividad Android	Pág. 20
4.3. Servicios Android	Pág. 22
4.4. Intent en Android	Pág. 24
4.5. El fichero Androidmanifest.xml	Pág. 25
4.6. Resumen	Pág. 25
Capítulo 5: Análisis	
5.1. Actividades de la Aplicación	Pág. 26
5.2. Primera Actividad	Pág. 27
5.3. Pantalla Principal	Pág. 28
5.3.1. Clientes Actuales	Pág. 29
5.3.1.1. Detalle Cliente	Pág. 30
5.3.1.1.1. Ver Stock	Pág. 31
5.3.1.1.2. Realizar Pedido	Pág. 32
5.3.1.1.3. Concertar Cita	Pág. 33
5.3.1.1.4. Histórico de Pedidos	Pág. 34
5.3.1.1.5. Histórico de Citas	Pág. 35
5.3.2. Clientes Nuevos	Pág. 36
5.3.3. Catálogo de Productos	Pág. 38
5.3.4. Confirmar Pedidos	Pág. 40
5.3.5. Gestión Material	Pág. 41
5.3.6. Agenda	Pág. 43
5.3.7. Mandar Informe	Pág. 45
Capítulo 6: Mejoras Aplicables	Pág. 46
Capítulo 7: Conclusiones	Pág. 47
Capítulo 8: Apéndice	
8.1. Definiciones y Acrónimos	Pág. 48
8.2. Índice de Figuras	Pág. 49
8.3. Bibliografía	Pág. 50

Agradecimiento

Quiero expresar mis agradecimientos a todas las personas que me han ayudado durante todo el proceso de mi formación. A todos los profesores de Ingeniería Informática, así como a la Universidad Politécnica de Valencia, por darme la oportunidad de formarme en una disciplina que siempre me ha apasionado.

También quería agradecer muy especialmente a mi tutor y director de proyecto, Juan Sánchez Díaz, que en todo momento ha velado por llevar a cabo este proyecto en los plazos que habíamos acordado.

Por último, un cariñoso recuerdo a mi familia y amigos, por estar siempre conmigo.

Capítulo 1: Introducción

1.1-Presentación del Proyecto

El objetivo de este proyecto es desarrollar una aplicación Android que permita a un comercial gestionar sus tareas de trabajo cotidiano.

Esta aplicación ayudará al comercial en su labor diaria, aportándole información en tiempo real sobre sus clientes y sus necesidades; gestionará su agenda de visitas e informará de sus actividades a su empresa.

La aplicación integrará varias tecnologías para llevar a cabo sus funcionalidades, geolocalización y servicios web y está destinada a ser usada desde cualquier dispositivo Android. De esta forma será ubicua y funcional desde cualquier lugar, puesto que el trabajo del comercial se realiza en su mayoría fuera de la oficina.

1.2-Ámbito General

El paradigma de computación predominante en informática ha variado últimamente con el desarrollo tecnológico de los teléfonos y la aparición de las tabletas, haciendo que cada vez sea más necesario aportar movilidad a las aplicaciones.

Hasta hace poco, el paradigma de computación se basaba en el ordenador personal o de escritorio conectado a internet y ubicado en un lugar determinado. Pero el surgimiento de teléfonos dotados de una capacidad de cómputo cada vez más importante y la aparición de tabletas (dispositivos compactos con características híbridas entre netbooks, computadores y smartphones), así como la posibilidad de conectarse a internet desde cualquier lugar, ha hecho que surja la necesidad de llevar consigo las aplicaciones que uno usaba en su ordenador y así poder tener acceso a sus funcionalidades.

De aquí ha nacido toda una industria orientada a satisfacer esa necesidad aportando a los usuarios aplicaciones que se pueden instalar en sus dispositivos.

Este campo se caracteriza por ser bastante nuevo y en constante desarrollo y mejora, debido a la rapidez con la que se van generando y agregando actualizaciones, tanto de programación, forma o contenido.

Cabe señalar que las aplicaciones requieren de una plataforma para ser utilizadas. Esta última hace referencia al sistema operativo en el cual pueden ser ejecutadas.

Los sistemas operativos para móviles adquirieron mayor importancia a medida que los dispositivos móviles crecieron en popularidad.

Android es un sistema operativo basado en Linux, creado por la empresa *Android Inc* que fue comprada por Google en el 2005. En el 2008 se popularizó gracias a la unión al proyecto de la *Open Handset Alliance*, una alianza formada por aproximadamente treinta organizaciones dispuestas a instaurar una telefonía abierta y de mejor calidad en el mercado.

Android fue diseñado inicialmente para teléfonos móviles. Sin embargo, al ser una plataforma independiente del hardware ha conseguido que en la actualidad lo usen tabletas, netbooks, reproductores multimedia e incluso algunos PC's .

Una característica de la plataforma Android es que tanto las aplicaciones incorporadas, como las que pueda crear un programador, pueden aprovechar los recursos disponibles en el dispositivo. Además, las aplicaciones para Android se diseñan en Java excepto los ejercicios sobre Linux. Android incluye una variedad de funciones para aplicaciones móviles, lo que hace que sea atractivo para desarrollar aplicaciones, por lo que hoy en día es el sistema operativo más presente en el sector de los dispositivos móviles.

En este contexto de Android y de la computación ubicua o móvil se encuentra orientado el proyecto final de carrera, siendo el objetivo principal el de desarrollar una aplicación cliente sobre la plataforma Android para gestionar la labor de un comercial de productos cosméticos.

En los siguientes capítulos se describirá con más detalle la plataforma Android y la aplicación realizada.

1.3-Objetivos del Proyecto

En el contexto de trabajo de un comercial de una marca de productos cosméticos surgió la idea de desarrollar una aplicación que aporte soporte a la labor diaria del comercial.

La aplicación debe aportar al usuario la información necesaria para cumplir con las exigencias de su trabajo y así poder servir de puente entre el usuario y la empresa para la cual trabaja.

Esta aplicación debe ser capaz de llevar el control del estado de stock de cada uno de los clientes, avisando al comercial cuando se llega al punto de pedido. También debe gestionar los pedidos que realizan los clientes a través del comercial y dar seguimiento a estos pedidos, controlando las fechas de entrega y confirmando el suministro de la mercancía.

A su vez, cada comercial debe disponer de suficientes muestras, packs y catálogos que le manda la empresa a su domicilio para ofrecérselos a sus clientes. La aplicación hará seguimiento del material del que dispone el comercial y lanzará pedidos automáticamente a la empresa.

Además ofrecerá al comercial el camino más óptimo para visitar a sus clientes dependiendo de la cita contratada. De este modo la aplicación administrará la agenda del comercial.

Por último, la aplicación mandará a la empresa informes diarios de las actividades realizadas por el comercial, pedidos realizados, descuentos aplicados, clientes visitados y material utilizado.

1.4-Solución Propuesta

La propuesta es una aplicación móvil sobre la plataforma Android para gestionar de manera integral los requerimientos descritos en la sección anterior.

La aplicación hará uso de llamadas a servicios web para sincronizar los datos con la cuenta del usuario y así descargar los listados con la información necesaria y mandar informes a la empresa.

Usará además la geolocalización para ofrecer al usuario el camino más óptimo, y de esta forma, proponerle realizar su trabajo de forma más productiva.

Integrando estas tecnologías, el usuario dispondrá de una herramienta que le facilite toda la información para llevar a cabo su labor de una forma eficiente y reducirá los tiempos de suministro y de abastecimiento de mercancía tanto al propio comercial como a sus clientes.

La empresa, por su parte, llevará un control de las actividades que realiza el comercial a diario puesto que le llegarán informes diarios resumiendo sus tareas.

Capítulo 2: Contexto Tecnológico

En este capítulo se describirán las tecnologías que se usarán para desarrollar la aplicación, explicando cada una de ellas, así como la forma de integrarlas y las soluciones que aportan.

2.1-La plataforma Android

Android es un sistema operativo basado en Linux y destinado principalmente a dispositivos móviles. Se estructura en cinco capas:

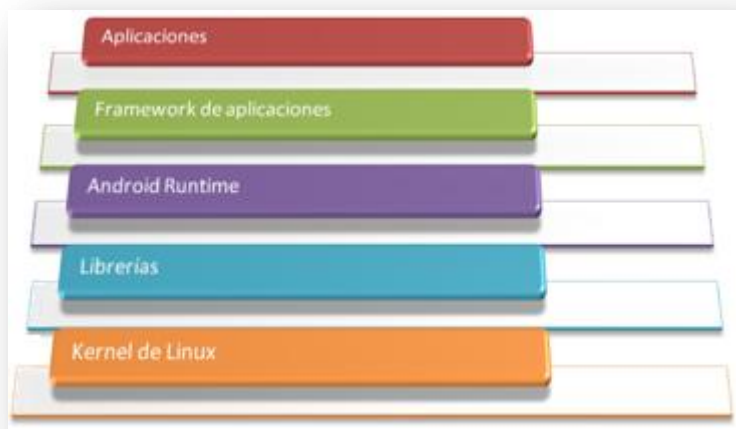


Figura 1: Capas de la plataforma Android

En la primera capa se encuentra el kernel de Linux ⁽¹⁾ y contiene los manejadores hardware. Esta capa ofrece servicios como gestión de procesos, memoria y sistema de archivos. De esta forma proporciona una abstracción de hardware que permite mantener los niveles superiores independientes del cambio que pueda sufrir esta capa.

Por encima del núcleo de Linux se tiene la capa de las bibliotecas, tal como *WebKit* ⁽²⁾, *OpenGL* ⁽³⁾ o *SQLite* ⁽⁴⁾ y *Surface Manager* ⁽⁵⁾ entre otras. Esta capa, junto a la anterior constituyen el corazón de Android, ya que proporcionan funcionalidades a las tareas que se repiten con frecuencia, evitando así codificarlas cada vez y garantizando llevarlas a cabo eficientemente.

Más arriba encontramos la capa del entorno de ejecución Android, o RunTime, constituida por librerías Java. Aunque la pieza más importante de esta capa es la máquina virtual de Android llamada *Dalvik* (en honor a un pueblo de

pescadores en Islandia, país de su creador *Dan Bornstein*). Ésta máquina virtual fue creada para permitir la ejecución de varias aplicaciones al mismo tiempo, usando un hardware con recursos limitados (poca memoria y poca capacidad de cálculo). Las instrucciones que ejecuta esta máquina son basadas en registros, mientras que las instrucciones que ejecuta la máquina virtual clásica de Java son basadas en una pila. Una máquina basada en registros requiere menos instrucciones para ejecutar el mismo código que una máquina basada en pila. Siendo los dispositivos móviles de pocos recursos esto hace de *Dalvik* la máquina virtual ideal para ellos.

Pero los ficheros en Bytecode de java pueden ejecutarse una vez transformados a DEX (*Dalvik EXecutable*) transformación que realiza la máquina virtual *Dalvik*.

A continuación se encuentra la capa del Framework de aplicaciones, que contiene las clases y los servicios que utilizarán directamente las aplicaciones para llevar a cabo su función. Los componentes de esta capa son librerías Java que acceden a los servicios de la capa anterior a través de máquina virtual *Dalvik*.

Por último, encontramos la capa de aplicaciones que incluye todas las aplicaciones del dispositivo tanto las nativas como las que el usuario haya instalado.

Sobre esta última capa vendrá la aplicación de este proyecto.

2.2-Geolocalización

La geolocalización define la localización o ubicación de un objeto mediante un sistema de coordenadas que serán utilizadas, una vez obtenidas, para posicionar ese objeto en un mapa y así poder tratar la información de su emplazamiento.

En este proyecto, se utilizará la geolocalización para definir un camino óptimo para recorrer los clientes que tiene que visitar el usuario partiendo de su localización actual y de la hora a la que se fijó la visita.

La obtención de las coordenadas que definen la posición se hace de dos formas, por GPS si el dispositivo dispone de esta utilidad o por un método llamado triangulación.

El GPS ofrece una localización muy precisa y usa los satélites para obtener las coordenadas, mientras que la triangulación se apoya en la intensidad de la

señal emitida por el dispositivo y de las antenas que están a su alrededor, y de esta forma se obtiene la posición del emisor.

2.3-Servicios Web

Es un mecanismo que permite comunicar dos dispositivos usando un estándar y un protocolo de comunicación predefinidos independientemente del lenguaje de programación que usa cada uno de los interlocutores.

Los servicios web permiten invocar un método o una operación remotos y obtener un resultado.

Cada servicio web tiene una interfaz definida que describe las funciones que ofrece, sus parámetros y el tipo de dato que devuelve. Esta interfaz se define usando WSDL (*Web Service Description Language*) que es un fichero XML ⁽⁶⁾.

Los interlocutores comunican con el servicio conforme a su descripción usando SOAP ⁽⁷⁾ y transportando el mensaje por HTTP.

El funcionamiento de los servicios web se puede resumir de esta forma, el cliente manda una solicitud al servidor; el servidor analiza el mensaje recibido y extrae la función que debe ejecutar, la procesa y acto seguido manda al cliente un nuevo mensaje con el resultado de la función.

Se distinguen dos tipos de servicios web, los que exponen sus funcionalidades como un conjunto de recursos identificables y accesibles por la sintaxis y la semántica del protocolo HTTP y los que cuya especificación está basada en los estándares WSDL y SOAP.

2.4-Aplicación Android

Una aplicación para Android es un programa escrito en el lenguaje de programación Java que, después de compilarlo con el SDK ⁽⁸⁾ Android, se convierte en un paquete que contiene todos los recursos que requiere la aplicación.

Cada aplicación Android se ejecuta en un proceso independiente con su propia instancia de la máquina virtual *Dalvik* y con sus propios archivos privados en el sistema de archivos, se le asigna un identificador de usuario y un entorno de operación seguro.

Las aplicaciones pueden acceder a los recursos compartidos del sistema según los privilegios que le han sido concedidos.

El SDK Android proporciona todo lo necesario para desarrollar una aplicación e incluye seis componentes claves para facilitar la implementación:

- **Actividades:** son las funciones que ejecutará la aplicación. Una aplicación Android es un conjunto finito de actividades en las que cada una tiene una única tarea u objetivo, normalmente destinadas a gestionar la visualización de una pantalla.

- **Vistas:** definen el diseño de la aplicación y son la pieza básica de la interfaz de usuario.

- **Intents:** Android utiliza un mecanismo de mensajes asíncronos para asociar peticiones de tareas con su actividad correspondiente. Cada petición se empaqueta en un *Intent*. Es el método principal de comunicación entre actividades y servicios.

- **Servicios:** procesos que se lanzan en segundo plano sin la intervención del usuario.

- **Notificaciones:** alertas que recibe el usuario.

- **Proveedores de contenido:** se encargan de compartir datos entre diferentes aplicaciones.

Los datos de una aplicación pueden alojarse en el sistema de ficheros, en una base de datos SQLite u otro sistema de almacenamiento persistente al que pueda acceder la aplicación. Usando los proveedores de contenido, otra aplicación puede consultar esos datos.

Este diseño del sistema Android tiene una ventaja única, que consiste en que una aplicación puede lanzar un componente de otra aplicación. Eso hace que una aplicación Android no tenga un único punto de lanzamiento (no existe una función `main()` tal como es habitual en otros sistemas). Pero como cada aplicación se ejecuta en un proceso separado, no puede acceder directamente a los componentes de otra aplicación. Es el sistema que se encarga de lanzar componentes que no pertenecen a la aplicación en curso.

2.5-Integración

El SDK Android proporciona un gran conjunto de interfaces de programación de aplicaciones (API) ⁽⁹⁾. Se utilizan librerías de clases conocidas a través de los paquetes Java de Android para realizar tareas comunes como

el uso de los gráficos, el acceso a bases de datos, el acceso a la red y más utilidades.

Los paquetes de Android incluyen soporte para los controles de la interfaz de usuario y el diseño de la misma, la navegación Web, soporte para la manipulación de ficheros XML, soporte para el almacenamiento estructurado de datos en bases de datos relacionales y el acceso a hardware mediante gestores.

Así mismo, para integrar la geolocalización en la aplicación se hará uso del paquete *Android.location* cuyo componente central es el servicio *locationManager* que proporciona la localización. Más tarde se añadirán mapas a la aplicación, usando el API de *Google Maps*, que proporciona funcionalidades como la descarga de mapas y su manipulación. Usando llamadas a funciones del API podemos marcar puntos en el mapa y calcular caminos entre dos puntos marcados.

Para integrar *Google Maps* en la aplicación están las librerías de *Google Play Services* en el *SDK*. La integración de los *Web Services* es sencilla puesto que se basa en llamadas a métodos HTTP para acceder a ellos. Se crearán clases que manejarán la comunicación entre el dispositivo y el servicio web.

2.6-Conclusiones

En este capítulo se ha presentado la plataforma Android y las tecnologías que harán falta usar para desarrollar la aplicación objetivo de este proyecto. También se ha visto la forma de integrarlos. Android proporciona un entorno para desarrollar aplicaciones dotándolas de funcionalidades útiles y a medida de los requerimientos del usuario.

En el siguiente capítulo se detallan los requerimientos de la aplicación para más tarde, documentarlos utilizando el lenguaje unificado de modelado (UML)

Capítulo 3: Descripción del Proyecto

Este capítulo dará una visión general del proyecto, describirá la solución que se va a aportar y detallará las funcionalidades claves que presenta.

3.1-Descripción de los Requerimientos

El trabajo de un agente comercial de una empresa de productos cosméticos, consiste en promocionar los productos que ofrece su empresa y presentar a sus clientes las nuevas variedades de los géneros que actualmente están usando, así como capturar nuevos clientes.

Para ello debe disponer de un catálogo actualizado de productos con las ofertas vigentes y el listado de sus clientes, cada uno con una previsión del estado actual del stock de los productos que le han sido enviados anteriormente. Si el stock de algún cliente llega al punto de pedido, el comercial se pone en contacto con él para confirmar un nuevo pedido y ofrecerle muestras de las nuevas variedades de productos que le pueden interesar. Cada producto cosmético tiene un número de utilizaciones y dependiendo de cada producto, se envía en una o varias unidades.

Según el volumen del pedido, el comercial puede ofrecer a sus clientes un porcentaje de descuento.

Con frecuencia, la empresa le manda a sus comerciales un listado de nuevos clientes que deben visitar e intentar convencerles para que utilicen los productos de la empresa, haciéndoles demostraciones u ofreciéndoles precios atractivos.

Según la zona geográfica en la que actúa cada comercial, se le manda el listado de los nuevos clientes a capturar y un pack de bienvenida que contiene muestras de productos básicos que el comercial entregará a los nuevos clientes así como un catalogo de todos los productos que vende su empresa.

A los nuevos clientes, se les hace un descuento durante el primer año, además del descuento por volumen de compra.

Por todo ello, el comercia, diariamente, tiene que gestionar a los clientes a los que hay que suministrar genero por ruptura de stock, ofrecer a su clientela actual muestras de los nuevos productos y visitar los nuevos clientes para

intentar contratarlos, llevar la cuenta del material de que dispone (catálogos, muestras y packs de bienvenida) y por último, mandar un informe de sus actividades a la empresa al final de cada jornada laboral.

3.2-Análisis de requerimiento

3.2.1- Diagrama de clases

En este apartado se reflejaran los requerimientos descritos anteriormente usando el lenguaje de modelado UML.

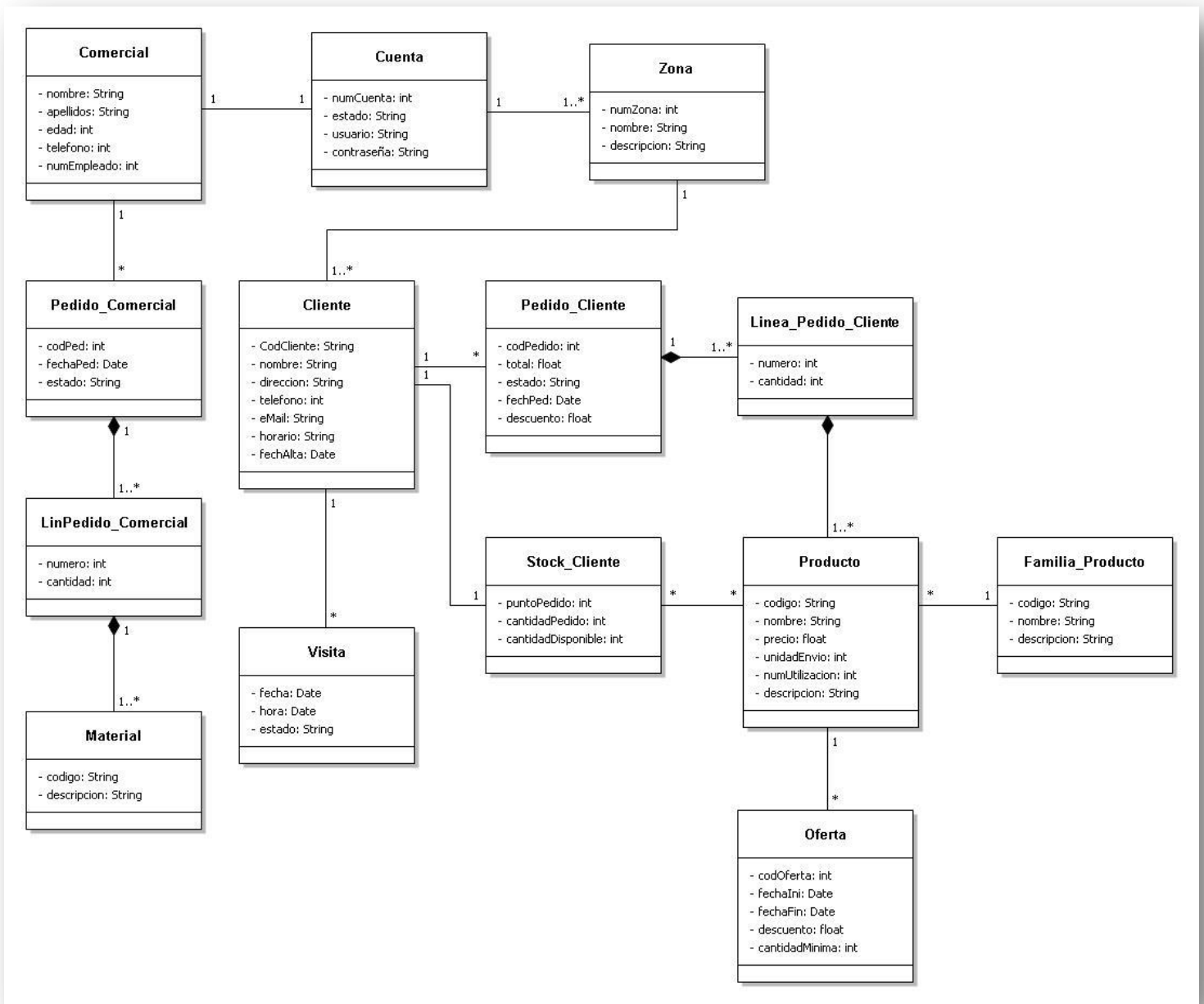
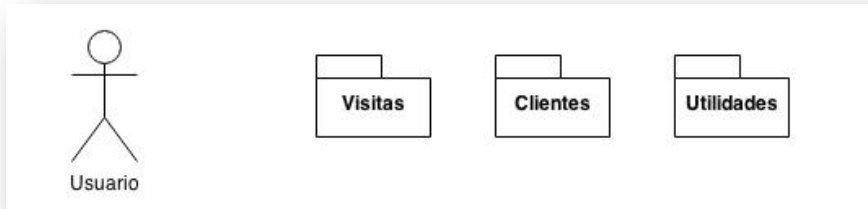


Figura 2: Diagrama de clases del caso de estudio

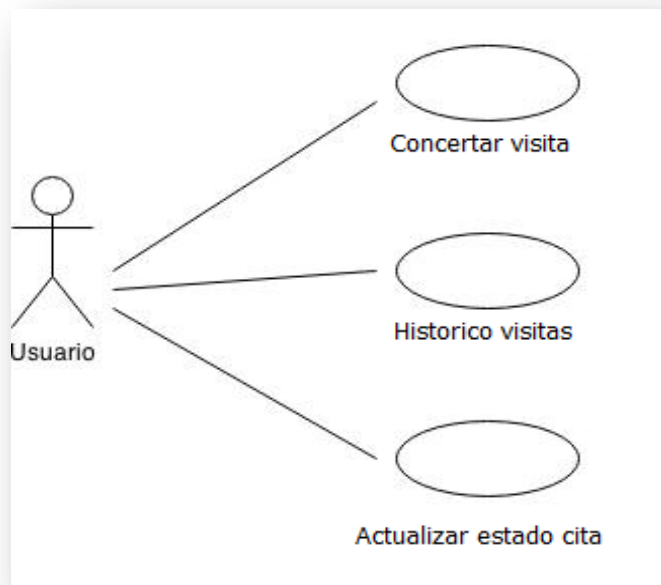
3.2.2- Casos de uso:

Después del diagrama de clases, paso a describir los requerimientos del usuario usando diagramas de casos de uso.

Por una parte, para el actor Usuario tendremos:

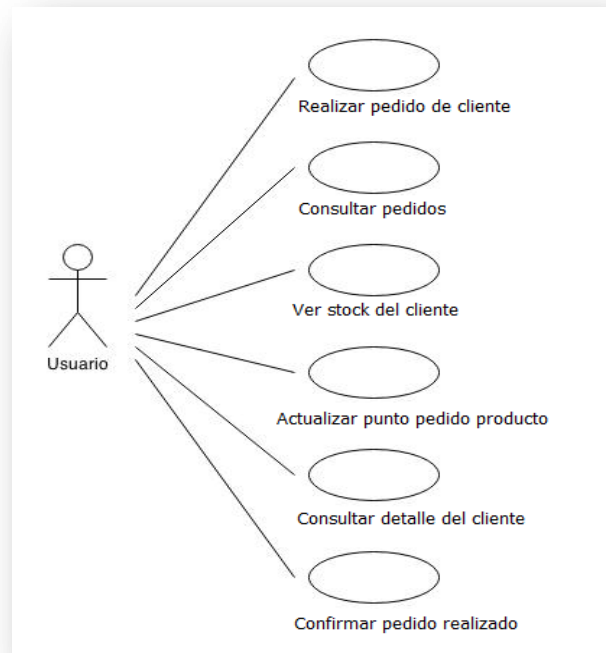


El paquete visitas contendrá los siguientes casos de uso:



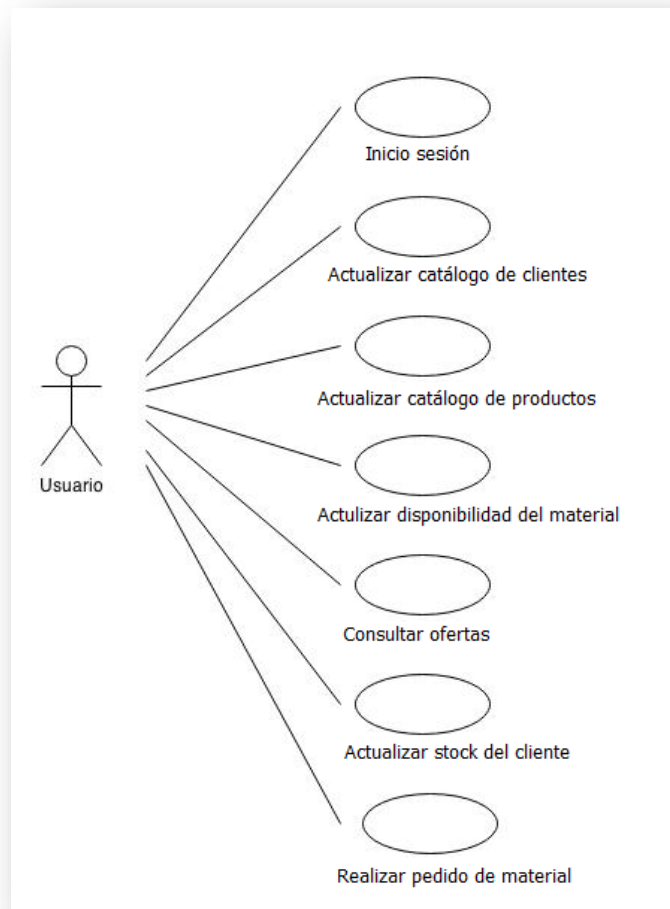
- **Concertar visita:** ofrecerá al usuario la posibilidad de contactar al cliente para acordar una cita.
- **Histórico visitas:** para cada cliente, el usuario puede consultar el histórico de las visitas que haya realizado.
- **Actualizar estado cita:** una vez concertada una visita y después de haberla realizado, el usuario debe marcarla como realizada.

El paquete cliente recoge los siguientes casos de uso:



- **Realizar pedido de cliente:** se accede al catalogo de productos para realizarle un pedido al cliente.
- **Consultar pedidos:** ver el histórico de los pedidos realizados y confirmar la llegada de mercancía.
- **Ver stock del cliente:** ver el estado de existencias de cada producto que haya pedido el cliente anteriormente. Se puede actualizar el número de artículos disponibles.
- **Actualizar punto pedido producto:** definir el número de artículos mínimo a partir del cual hay que realizar un pedido de dicho artículo.
- **Consultar detalle del cliente:** ver la información del cliente, su número de teléfono, su dirección, fecha de alta, etc.
- **Confirmar pedido realizado:** confirmar que al cliente le ha llegado la mercancía que ha pedido previamente y actualizar el estado de stock de cada artículo pedido.

Por último el paquete utilidades contendrá los siguientes casos de uso:



- **Inicio sesión:** la primera vez que el usuario lance la aplicación deberá identificarse aportando su usuario y su contraseña.
- **Actualizar catálogo de clientes:** recibir el listado de los clientes nuevos a los que tiene que contactar el usuario.
- **Actualizar catálogo de productos:** recibir los catálogos actualizados de los productos que comercializa la empresa junto con las ofertas vigentes para cada uno de los artículos.
- **Actualizar disponibilidad del material:** mandar a la empresa la disponibilidad real de cada uno de los materiales de los que dispone el usuario.

- **Consultar ofertas:** ver los productos que están de oferta actualmente.
- **Actualizar stock del cliente:** poner al día las existencias de las que dispone el cliente.
- **Realizar pedido de material:** pedir a la empresa abastecer al usuario de material (muestras, catálogos, pack de bienvenida) para que los pueda ofrecer a sus clientes.

Capítulo 4: Aplicación Android

En este capítulo se verá la estructura que tiene una aplicación Android y se describirán los componentes que tendrá nuestra aplicación conforme a los requerimientos declarados en el capítulo anterior.

4.1-Estructura y componentes de una aplicación Android

Las actividades representan el eje fundamental de cualquier aplicación Android. Cada actividad ejecuta una tarea específica de la aplicación y es generalmente representada por una única pantalla.

Cada actividad es responsable de manejar y administrar sus propios recursos y datos. Las actividades pueden dividirse en unidades más pequeñas llamadas *Fragmentos*, de esta forma se evita duplicar código en el caso de que varias actividades necesiten mostrar componentes similares.

La transición de una actividad a otra se realiza mediante un mecanismo de mensajes asíncronos llamados *Intent*. Un objeto de la clase *Intent* es procesado por el sistema operativo Android y al que responde lanzando la actividad o el servicio correspondiente.

Visto así, la estructura de una aplicación se podría definir como el conjunto de las actividades que la componen, junto con los datos necesarios para realizar sus tareas y la manera de transitar de una actividad a otra.

También podríamos abstraer la estructura de la aplicación en un modelo de tres capas donde en cada capa se agrupan las clases que realizan una tarea similar.

Estas tres capas podrían ser las siguientes: capa de la interfaz gráfica, capa lógica o de negocio y capa de acceso a los datos.

Android, sin embargo, divide la aplicación en una serie de carpetas donde en cada una se albergan clases destinadas a realizar funciones coherentes con las demás. Al compilar el proyecto, se crea un paquete *apk* que contendrá esa relación de carpetas y será este paquete el que se instalará en el dispositivo de destino.

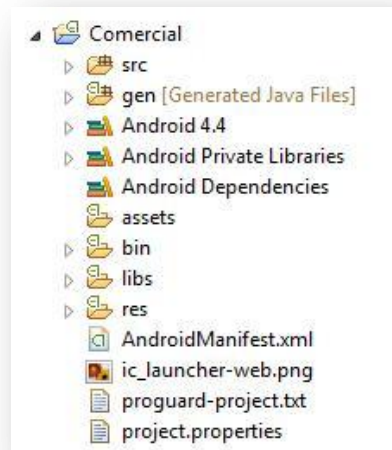


Figura 3: Carpetas de un proyecto Android

- **Carpeta src:** recoge la totalidad del código fuente organizado en paquetes
- **Carpeta gen:** esta carpeta guarda un conjunto de archivos (de código Java) creados automáticamente cuando se compila el proyecto, para poder dirigir los recursos de la aplicación. Contiene la clase R que ajusta automáticamente todas las referencias a archivos y valores de la aplicación (guardados en la carpeta res).
- **Carpeta res:** contiene los recursos necesarios para la generación de la aplicación. A su vez se divide en subcarpetas que contienen las imágenes según la resolución del dispositivo, los textos, colores y el *Layout* que incluye los ficheros XML que definen el diseño de la interfaz gráfica.
- **Carpeta libs:** contiene las librerías privadas al proyecto y las librerías de terceros.
- **Carpeta assets:** contiene el resto de los recursos de la aplicación. La diferencia entre los recursos que se almacenan aquí y los que están en la carpeta res, es que los recursos de la carpeta res generan un identificador por recurso, ese identificador está gestionado por la clase R y sólo se puede acceder a esos recursos con determinados métodos de acceso. Los recursos de la carpeta assets, son accesibles por su ruta, y se accede a ellos como flujo de bytes dejando al programador convertirlos al objeto adecuado.

- **Fichero AndroidManifest.xml:** es donde se define el nombre de la aplicación y se da información sobre la versión, así como los componentes por los que está formada, permisos que necesita para ejecutarse e información sobre su configuración.

El primer paso para diseñar y desarrollar aplicaciones Android es entender los diferentes estados dentro del ciclo de vida de una actividad. Por ello, vamos a adentrarnos un poco más en este mecanismo, explicando los diferentes estados por lo cual pasa una actividad y la forma de transitar de un estado a otro, así como la manera de comunicar una actividad con otra.

4.2-Actividad Android

Las actividades forman las pantallas de la aplicación, desempeñan un papel fundamental en su ciclo de vida y están compuestas por componentes secundarios denominados vistas.

Las vistas son lo que se ve en la pantalla y con lo que interactúa el usuario. Se encargan de controlar el diseño, proporcionan elementos de texto, botones, formularios para entrada de datos y dibujan gráficos en la pantalla. Las vistas se organizan de forma jerárquica, y necesitan recursos para desempeñar su papel.

Existe en Android una clase R.java, creada automáticamente, constituye una referencia a los recursos. Se utiliza esta clase para obtener una cadena o un color para luego añadirlo a una vista.

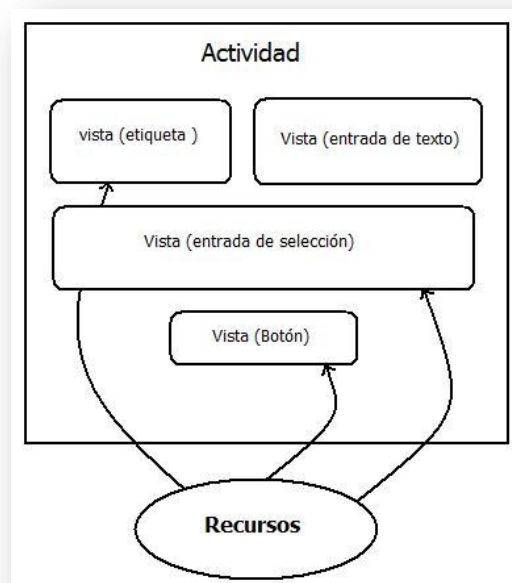


Figura 4: Relación entre actividad, vista y recursos

El sistema operativo Android utiliza una pila para controlar todas las actividades que se están ejecutando. Al iniciarse una nueva actividad, la que ocupaba la parte superior de la pila se detiene, y la nueva se coloca en su sitio. Cuando finaliza una actividad, se elimina de la pila y se reanuda la que se encuentra a continuación en la pila. Por lo cual, sólo puede existir una actividad activa y visible para el usuario a la vez.

Las actividades son las responsables de gestionar su estado. Los diferentes cambios de estado dentro del ciclo de vida de una actividad están señalados por llamadas a métodos.

Las actividades tienen tres posibles estados:

- **Activo:** ocurre cuando la actividad está ejecutándose en primer plano y por lo tanto está visible en la pantalla del dispositivo y es la actividad principal.
- **Pausado:** la actividad se encuentra semi-suspendida. Todavía se está ejecutando, pero no es la principal. El sistema puede prescindir de ella en caso de necesitar memoria, por lo cual hay que guardar toda la información de su contexto.
- **Parado:** Actividad detenida y no es visible en la pantalla. El sistema puede destruirla y así obtener más recursos. Si más tarde se decide reanudarla, hay que lanzarla desde el principio.

Los métodos más importantes para la transición de un estado a otro son:

- **onCreate():** es el método que crea la actividad, recibe un parámetro de tipo *Bundle* que contiene el estado anterior de la actividad. Si esa actividad ha sido eliminada por motivos de falta de recursos, también puede ser *null* en caso de no disponer de estado anterior o si la actividad es nueva. Después de este método, siempre se llama al método `onStart()`
- **onStart():** muestra en pantalla la actividad. Le sigue `onResume()` si la actividad pasa a ejecutarse en segundo plano o por `onStop()` si desaparece de la pantalla.
- **onResume():** establece la interacción con el usuario. En este momento la actividad se encuentra en la parte superior de la pila. Se invoca siempre, ya sea un inicio o un reinicio. Le sigue el método `onPause()`
- **onRestart():** reinicia una actividad si ha sido parada.
- **onPause():** se ejecuta cuando una actividad deja de ejecutarse para dar paso a otra. Aquí es donde hay que considerar la persistencia de los datos.

- **onStop():** se invoca cuando la actividad pasará al segundo plano para un periodo de tiempo largo, pudiendo ser expulsada de la memoria o reanudada desde el principio.
- **onDestroy():** es el destructor de la actividad y por tanto el método que la finaliza.

En la siguiente figura, se resumen los estados por los cuales pasa una actividad y los métodos involucrados.

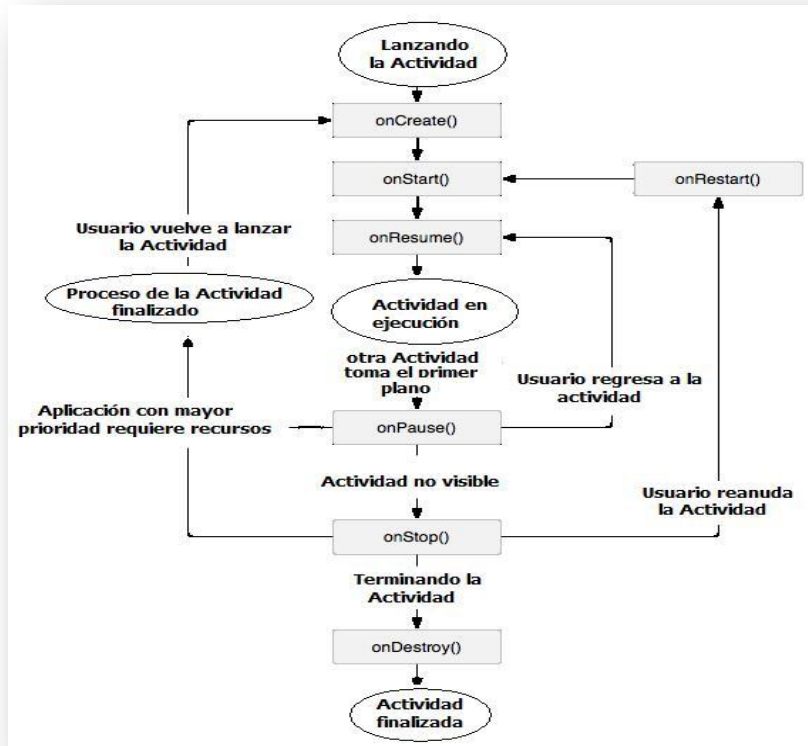


Figura 5: Ciclo de vida de una actividad

En el próximo capítulo se detallarán las actividades que formarán la aplicación de este proyecto.

4.3-Servicios Android

Son componentes de la aplicación que se ejecutan en segundo plano, sin tener una interfaz grafica para mostrar en pantalla llevando a cabo las operaciones de una actividad. Se ejecutan en el mismo proceso que la aplicación que los invoca.

En este proyecto, se implementará la sincronización de los datos como Service ya que no implican una interacción directa con el usuario.

Los servicios pueden estar vinculados a una actividad, al finalizar su ejecución el servicio retorna el resultado.

Hay dos formas de lanzar un servicio: si una actividad requiere hacer un

trabajo inicia el servicio correspondiente, o si una aplicación quiere exponer parte de su funcionalidad para que sea accesible para otras aplicaciones, se vincula al servicio requerido.

Dependiendo de la forma en la que se lanza el servicio, éste tiene un ciclo de vida distinto, iniciándose con el método `onCreate(Bundle)` y liberándose con el método `onDestroy()`.

Si se llama al método `startService()`, entonces el sistema recuperará el servicio llamando a su método `onCreate()`. Más tarde, llamará al método `onStartComand()`, y desde este momento el servicio estará ejecutándose hasta que se llame al método `stopService()`.

Si se quiere vincular a un servicio, se llamará al método `bindService()` para obtener una conexión persistente con el servicio. Si el servicio no se encuentra ya en ejecución, se llamará al método `onCreate()` tras el cual se llamará al método `onBind()`. En este caso, el servicio se termina llamando al método `onUnbind()`.

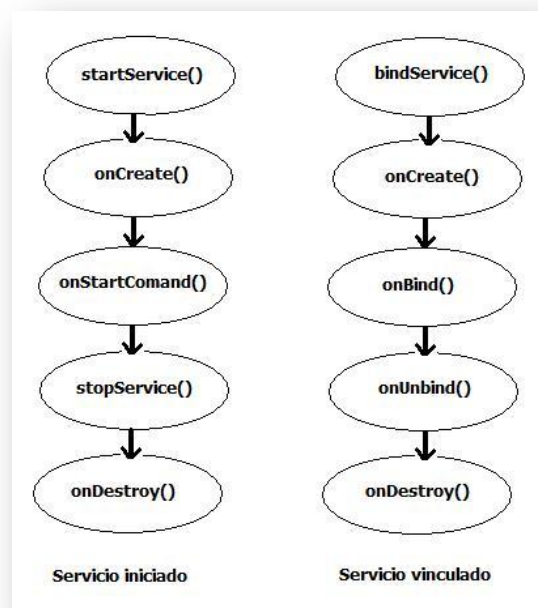


Figura 6: Ciclo de vida de los Servicios

Como ya hemos visto, un servicio se puede iniciar y vincular pero también en algunas ocasiones, puede realizar las dos cosas. El inicio de un servicio está relacionado con la presencia de una tarea de fondo. Una vez iniciado el servicio, se ejecuta hasta que se detiene explícitamente.

De momento se ha explicado dos componentes básicos de cualquier aplicación Android, las actividades y los servicios, pero ¿cómo se pasa de una actividad a otra?

4.4-Intent en Android

En una aplicación típica de Android, se crean clases *Activity* y *View* para definir la interfaz y posiblemente objetos *Service* subyacentes. Se pasa de una pantalla a otra por medio de invocaciones *Intent*. Aunque sea su principal uso, no es el único.

Los *intent* también se utilizan para difundir eventos a receptores, para enviar un evento de fondo. De esta forma un *intent* emitido no invoca ninguna actividad. También sirven para instanciar a servicios.

La aplicación de este proyecto utilizará internamente objetos *intent* para pasar de una actividad a otra, y también invocará *intent* desde las aplicaciones incorporadas de Android para llamar a un cliente o ver su dirección en el mapa...

Los objetos *intent* son la red de comunicación de las aplicaciones Android y están formados por tres fragmentos de información principales: acción, categorías y datos, además de un conjunto adicional de elementos. Siendo la acción una cadena que informa del tipo de acción llevada a cabo. Las acciones pueden ser dadas por la clase *Intent*, por una API de Android o definidas por el diseñador. La categoría es otra cadena que contiene información adicional sobre el tipo de componente al que va dirigido el *intent*. La lista de categorías está incluida en la clase *Intent*. Y datos son los que informan del identificador (URI) del dato que se asocia a la acción y del tipo de ese dato. Es importante la coherencia entre todos los elementos, ya que si la acción requiere un dato de tipo texto, un *intent* con un dato de tipo imagen no podría ser lanzado.

Cuando un componente como una actividad desea invocar a un *intent*, lo puede hacer de dos formas:

- **Invocación implícita:** dónde la plataforma determina qué componente es el más indicado para ejecutar a través de un proceso de resolución con la acción, datos y categorías.
- **Invocación explícita:** donde el código especifica directamente qué componente debe procesar el *intent*, especificando la clase o nombre del componente del receptor.

4.5-El fichero Androidmanifest.xml

Toda aplicación Android posee este fichero en el directorio raíz del proyecto. En él se describe el contexto de la aplicación, sus actividades, servicios, receptores y proveedores de contenido, el nombre de la aplicación, datos sobre su versión, la configuración del dispositivo requerida así como los permisos que se conceden a la propia aplicación. En definitiva, este fichero contiene la configuración de la aplicación, describiendo con detalle la identidad de la aplicación. Es utilizado por Android para instalar, actualizar y ejecutar la aplicación.

4.6-Resumen

Una aplicación Android gira en torno a la interfaz gráfica de usuario. En este capítulo se ha analizado los conceptos de actividad, vista, servicios, Intent y se ha visto los conceptos y métodos implicados en ellos. También se ha mostrado el fichero manifiesto donde se combinan todas las piezas para definir la aplicación.

Este capítulo constituye la base fundamental para el desarrollo de la aplicación, y recoge los conocimientos de los que hay que disponer antes de abordar la tarea de programación.

Capítulo 5: Análisis

En este capítulo se va a abordar el análisis de la aplicación. Se detallará el flujo de dicha aplicación para resaltar la relación entre la funcionalidad de ésta. Al afrontar este proceso, se asegura que la aplicación proporcione la funcionalidad necesaria para cubrir los requerimientos del caso.

5.1-Actividades de la aplicación

La aplicación contiene varias actividades, empezando por la primera, donde se invita al usuario a poner sus credenciales para acceder a su cuenta. A continuación, se muestra el árbol de navegación entre las actividades de la aplicación.

5.2. Login

5.3. Pantalla Principal

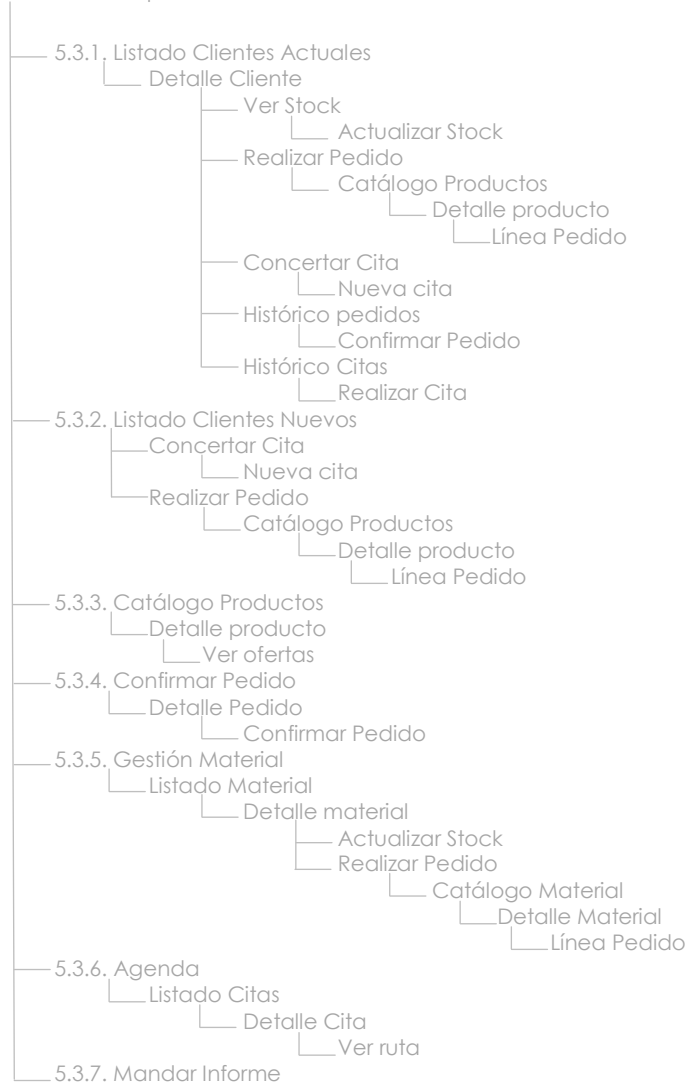


Figura 7: Árbol de actividades

A continuación se va a describir cada una de las actividades detallando las funcionalidades que aporta y los procesos que maneja.

5.2-Primera actividad

Se muestra al comenzar la aplicación. Consta de dos campos de texto; donde el usuario tiene que introducir su nombre de usuario y su contraseña; así como un botón para mandar la información y de esta forma, al identificarse, dar acceso a su cuenta.

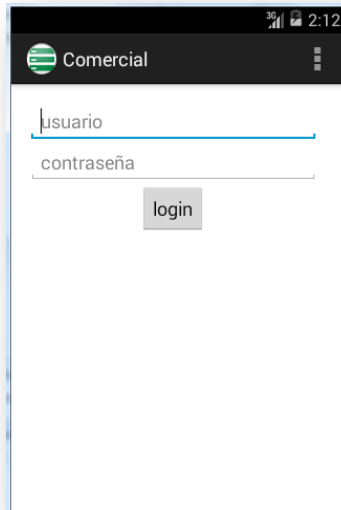


Figura 8: Interfaz de la actividad Login

Al recoger los datos del usuario, se abre una conexión https al servidor de la empresa para averiguar dichos datos: si son correctos se pasará a la siguiente actividad. Sin embargo, si no son correctos, se le muestra al usuario un mensaje de error y se le vuelve a ofrecer la misma interfaz para volverlo a intentar.



Figura 9: Acceso con datos incorrectos

5.3-Pantalla principal

Una vez el usuario está identificado, se muestra una pantalla donde se presentan varias opciones a elegir.

Al mostrarse esta pantalla, aparecen varios servicios que se ejecutarán en segundo plano. Son tareas de sincronización con la cuenta del usuario y serán las encargadas de actualizar los datos de los que dispone la aplicación. Se descargará el listado de los nuevos clientes que la empresa ha asignado al comercial y, en caso de tener nuevas ofertas en productos, se procederá a descargar el catálogo actualizado.

Esta información alimentará la base de datos de la aplicación.

Las opciones que puede elegir el usuario son:

- **Agenda:** donde el usuario podrá ver el calendario de las visitas que ha concertado con sus clientes.
- **Cientes actuales:** al elegir esta opción, el usuario accederá a ver el listado de sus clientes que tienen contratada alguna línea de productos.
- **Cientes nuevos:** listado de los clientes a los que tiene que llamar el usuario para concertar cita para visitarles.
- **Gestión material:** desde aquí el comercial gestionará el material que dispone, las muestras que tiene, los catálogos impresos y los packs de bienvenida.
- **Catálogo productos:** un listado actualizado de todos los artículos que comercializa la empresa, ordenados por familias de productos.
- **Confirmar pedidos:** listado de los pedidos realizados por los clientes y a los que el comercial tiene que llamar para confirmar que les ha llegado la mercancía.
- **Mandar informe:** esta opción recapitulará la información relativa a las actividades que ha realizado el usuario durante el día, y mandará un informe a la empresa. También es la encargada de mandar los pedidos que han realizado sus clientes, y los pedidos del material del que debe disponer el comercial.

Esta opción no presenta ninguna pantalla, solo un mensaje informando si la información se ha podido mandar con éxito o no.

A continuación se presenta el aspecto que tiene esta actividad, más adelante se detallaran cada una de sus funcionalidades:



Figura 10: interfaz de la actividad principal

5.3.1-Clientes actuales

Al acceder a esta opción, el usuario verá un listado de sus clientes. Como se ha comentado anteriormente, la aplicación debe llevar el control del estado de stock de cada uno de los productos que tienen cada uno de sus clientes.

Un cliente, al pedir un producto, le comunica al comercial la frecuencia de uso que hará del mismo, de esta forma se puede calcular una estimación de las unidades de las que dispone el cliente.

Haciendo este cálculo, los clientes a los que previsiblemente le quedarían pocas unidades de algún producto, se marcarán en el listado, para que el comercial se ponga en contacto con ellos y les invite a hacer un pedido. También podrá concertar una cita con sus clientes para ofrecerles muestras de las nuevas variedades de productos que ha lanzado su empresa. Y por supuesto, el usuario, por cada cliente, puede ver el historial de los pedidos que ha efectuado y actualizar el estado de dichos pedidos confirmando la entrega de mercancía, así como podrá ver el historial de las visitas que ha realizado.

Al confirmar que un pedido ha sido entregado, se actualizará la estimación de stock de cada producto solicitado

Esta actividad es muy sencilla, solo consta de una lista, residiendo la complejidad en los servicios que corren en segundo plano, que son los responsables de calcular y actualizar los datos.

A continuación se muestra la interfaz de esta actividad:

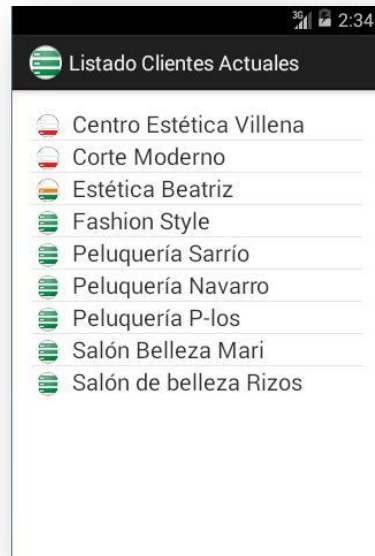





Figura 11: Listado clientes actuales

Como podemos comprobar en esta figura, se listan los clientes actuales del usuario. En la parte izquierda se indicará mediante un icono el estado de stock de cada cliente:

-  **color verde:** aparecerá cuando el stock del cliente esté abastecido
-  **color amarillo:** cuando se necesite suministrar al menos un producto
-  **color rojo:** cuando el cliente necesite suministro urgente por ruptura de stock.

Al seleccionar un cliente de la lista anterior se pasa a la siguiente actividad: detalle del cliente.

5.3.1.1-Detalle cliente

Esta actividad muestra por pantalla los detalles del cliente, y da opción a ver su estado de stock, realizar un nuevo pedido, ver el histórico de los pedidos ya realizados y las visitas efectuadas, así como concertar una nueva cita. Se podrá llamar al cliente desde esta actividad, o ver su ubicación en el mapa.



Figura 12: Detalle del cliente

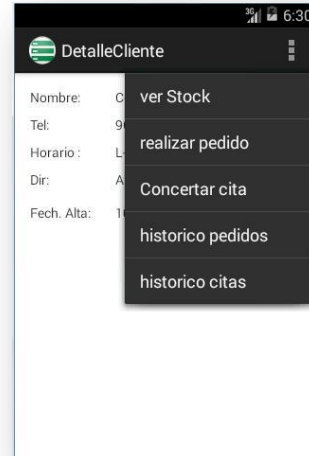


Figura 13: menú detalle del cliente

A continuación, se detallarán cada una de las opciones que ofrece esta actividad.

5.3.1.1.1-Ver stock

Esta actividad es la responsable de mostrar al usuario el estado de stock del cliente.

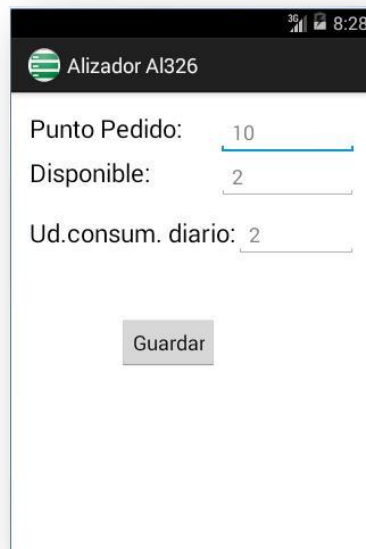
En ella se mostrará por cada artículo ya adquirido, una estimación de la cantidad que dispone. Se podrá modificar las unidades de consumo de cada artículo, atributo que se utiliza para calcular la estimación de stock disponible. También se puede actualizar, por cada producto la cantidad que dispone el cliente.



Figura 14: Detalle de stock del cliente

Como se puede ver en la figura, a la derecha de cada producto hay un indicador de existencia. Si la cantidad disponible es menor al punto de pedido el indicador esta en rojo, si la cantidad disponible se aproxima a la mitad del punto de pedido, el indicador es amarillo, y si hay suficiente género el indicador aparecerá en verde.

La cantidad disponible de cada producto se calcula en base a una estimación. Para obtener la cantidad disponible de un producto, el cliente debe comunicar al comercial la cantidad que se consume a diario. La aplicación calculará desde el día que se realizó el último pedido y en función de la cuantía pedida y los días laborales transcurridos, las reservas disponibles. Este dato se puede modificar por cada producto. También se puede cambiar la cantidad a partir de la cual hay que realizar el pedido (el punto de pedido) para así agilizar el abastamiento y garantizar la entrega antes de la ruptura de stock.



The screenshot shows a mobile application interface for updating product details. The title bar at the top displays 'Alizador AI326' and the time '8:28'. The main content area contains three input fields: 'Punto Pedido:' with a value of 10, 'Disponible:' with a value of 2, and 'Ud.consum. diario:' with a value of 2. A 'Guardar' button is located below the fields.

Figura 15: Actualización de los detalles de un producto

5.3.1.1.2-Realizar pedido

Una vez realizado el seguimiento de stock, y a petición del cliente, el comercial puede realizar un pedido de los productos que necesite. Los productos se agrupan en familias o categorías, para facilitar su búsqueda. Hay que tener en cuenta un detalle a la hora de realizar un pedido, los productos se presentan en paquetes que contienen las unidades, por lo tanto la cantidad pedida tiene que ser un múltiplo de la cantidad de expedición.

Para realizar un pedido, se muestra el catálogo de productos ordenados por familias. En una lista desplegable, al seleccionar un producto, se abre una ventana donde se deberá introducir la cantidad solicitada.

Según el producto que pida el cliente, si éste tiene asociada una oferta en vigor, se le aplicará el descuento correspondiente.

El pedido, una vez realizado, queda guardado en la base de datos de la aplicación y será enviado más tarde a la empresa para que lo prepare y lo envíe al cliente.

El comercial deberá dar seguimiento a los pedidos que realizan sus clientes, confirmando que han recibido la mercancía.

A continuación se muestran las actividades que gestionan la realización del pedido:

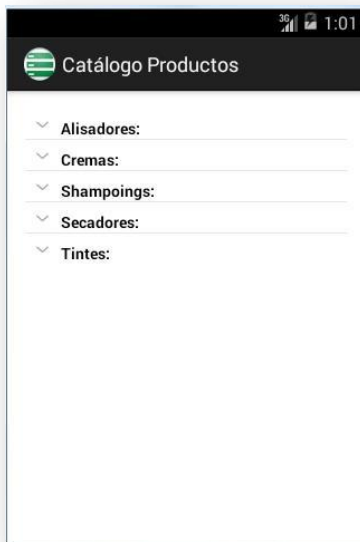


Figura 16: Catálogo productos



Figura 17: Detalle de una categoría



Figura 18: Línea pedido

5.3.1.1.3-Concertar cita

Cuando el usuario accede a los detalles del cliente, una de las opciones que se ofrece es poder concertar cita con el cliente por si éste está interesado en alguna nueva variedad de productos que ha lanzado la empresa, o por si quiere que el comercial le haga una demostración para mostrarle los beneficios de adquirir sus productos.

Esta actividad registra las citas que concierta el comercial con sus clientes. Más tarde, cuando el comercial acceda a su agenda, las verá reflejadas en su calendario (se explicarán las actividades de la agenda con más detalle posteriormente)

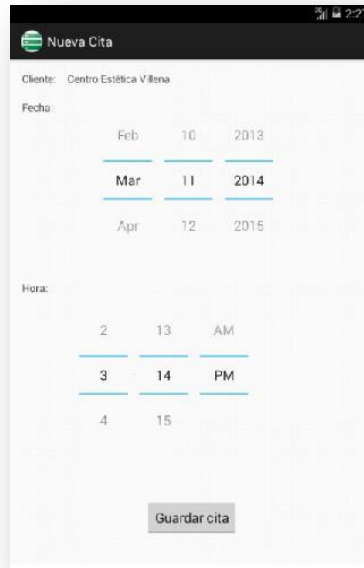


Figura 19: Interfaz para concertar cita con el cliente

5.3.1.1.4-Histórico pedidos

Una de las tareas del comercial consiste en dar seguimiento a los pedidos realizados por cada cliente para confirmar que la mercancía ha sido entregada. De esta forma, también se actualizan los valores de las existencias de la que dispone el cliente, valores que sirven a la aplicación para calcular la estimación de stock.

En esta actividad se mostrarán los pedidos que ha realizado el cliente hasta la fecha, su estado y el total de cada pedido, dando opción al usuario de cambiar el estado de los pedidos que todavía no han sido confirmados. A continuación se muestra el aspecto que tiene esta actividad:

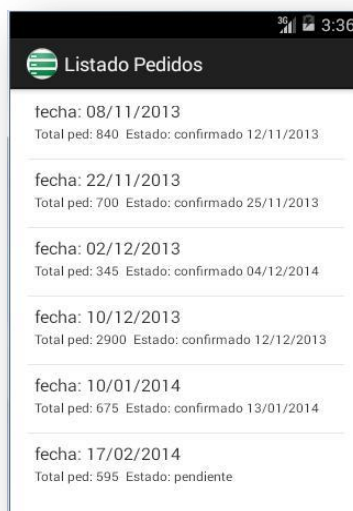


Figura 20: Histórico de pedidos de un cliente

Al seleccionar un pedido no confirmado, se abre un diálogo para poderlo confirmar con la fecha actual. Al realizar esta acción, se lanza un proceso en segundo plano, buscará ese pedido, y por cada línea que contiene, extraerá la cantidad pedida y las unidades que utiliza el cliente a diario, para actualizar el estado de stock del cliente.

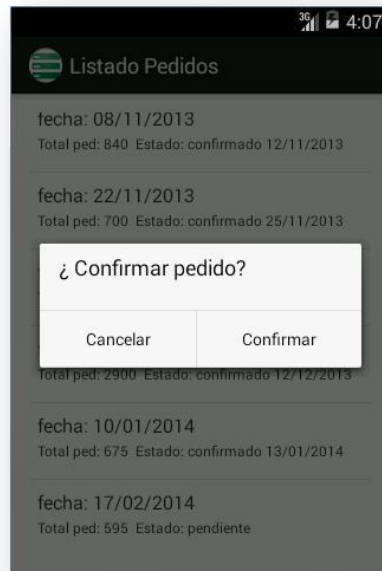


Figura 21: Confirmar un pedido pendiente

5.3.1.1.5-Histórico citas

Desde la actividad detalle del cliente se puede acceder al listado de las citas concertadas con dicho cliente. Esta actividad listará todas las citas que ha realizado el comercial, así como las nuevas citas que ha acordado y que todavía no han sido realizadas.



Figura 22: Listado citas concertadas con un cliente

Al seleccionar una cita que está pendiente de realizar, se abrirá un diálogo para poderla confirmar con la fecha actual, eso lo hará el comercial después de haberla realizado. La siguiente figura muestra el aspecto que tiene la interfaz.

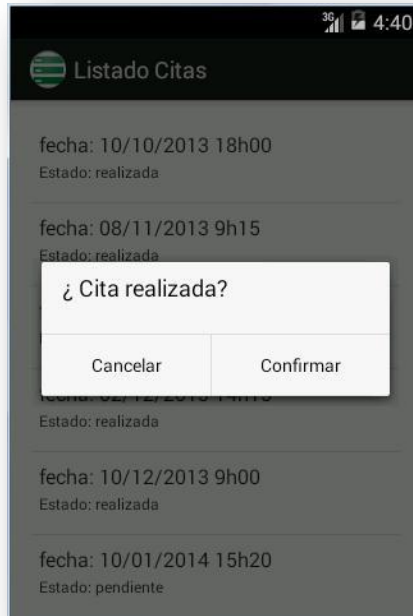


Figura 23: Confirmación de la realización de una cita

5.3.2-Clientes nuevos

Con frecuencia, la empresa manda a cada comercial, según su zona, un listado de clientes que no tienen contratados ninguno de sus productos. El comercial, al recibir este listado, debe concertar cita con estos clientes para presentarles los productos de su empresa.

Por cada cliente nuevo, la empresa manda automáticamente a casa del comercial un material que éste usará durante su visita. Se trata de un pack de bienvenida que contiene muestras de los productos básicos, y un catálogo impreso de todos los productos que comercializa la empresa.

A nivel de capa de datos, los clientes nuevos y actuales se distinguen por el campo *fechaAlta* en la tabla Clientes. Los clientes nuevos, al realizar su primer pedido, pasan a ser clientes actuales con fecha de alta aquella en la que realizaron su primer pedido.

La aplicación debe llevar el control del descuento del que son beneficiados los clientes nuevos, puesto que los clientes nuevos tienen un año de descuento en todos los pedidos que realizan, además del descuento por volumen de pedido o el descuento de las ofertas vigentes.

Si no hay listado de clientes nuevos, o si el comercial ha tratado a todos, la aplicación debe informar de ello.

Seguidamente, se muestran las actividades que llevan este proceso:

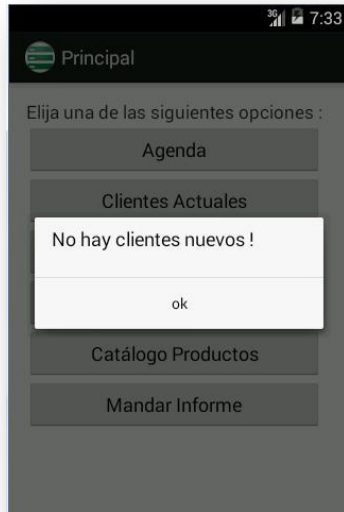


Figura 24: No hay listado de clientes nuevos

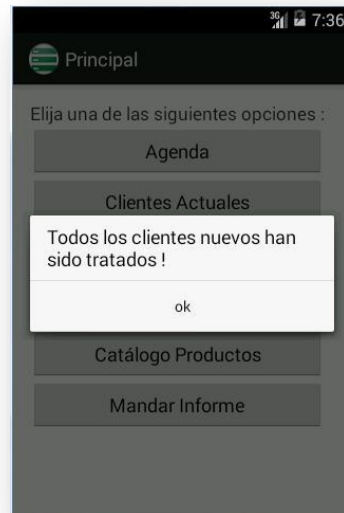


Figura 25: El comercial ha tratado todo el listado de los clientes nuevos

En caso de disponer de un listado de clientes nuevos se verá esta pantalla:



Figura 26: listado de clientes nuevos

Al seleccionar un cliente, se mostrarán los detalles del mismo, y se dará opción a concertar cita con él (cita que más tarde se verá reflejada en la agenda). Esta actividad ya ha sido implementada para el caso de los clientes nuevos. Al tener la vista organizada en fragmentos, éstos se pueden volver a utilizar ya que la funcionalidad que aportan es la misma para este caso. Esto, tal como se ha adelantado en el capítulo 4, es uno de los puntos fuertes que presenta Android. De esta forma se evita duplicar código y se opta por su reutilización.



Figura 27: Detalle cliente nuevo

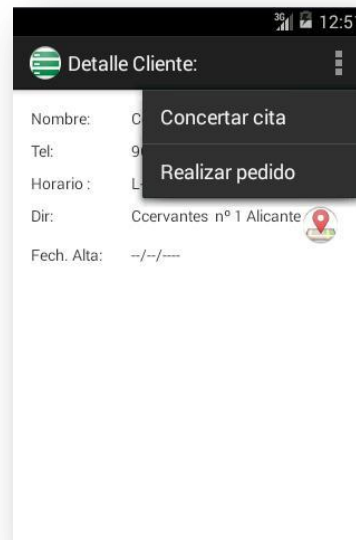


Figura 28: Menú cliente nuevo

Las opciones de concertar cita y realizar pedido presentan la misma interfaz que en el caso de clientes actuales, por lo cual no se volverán a repetir aquí. La única diferencia es que al realizar un pedido, un cliente nuevo se le da de alta con fecha el día de realización del pedido.

Esto influye en dos procesos, el cliente nuevo, al realizar un pedido, pasa a verse en el listado de los clientes actuales del comercial y a la hora de realizar un pedido se lee la fecha de alta de cada cliente para averiguar si tienen menos de un año y así aplicarles el descuento.

5.3.3-Catálogo productos

Con frecuencia la empresa actualiza el catálogo de sus productos, bien porque comercializa nuevos productos o nuevas variedades de los existentes, o porque asocia ofertas por volumen de compra de algunos artículos.

Después de realizar la conexión a la cuenta del usuario, la aplicación sincroniza el catálogo de productos.

Como se ha visto antes, los productos se agrupan por categorías. Al seleccionar una categoría se verán los artículos que le pertenecen y se accederá a su detalle y a las posibles ofertas asociadas. Aquí también, parte de la actividad ya ha sido realizada en el caso de realizar un pedido, por lo cual se volverá a utilizar el mismo código.

A continuación se muestran las actividades involucradas.

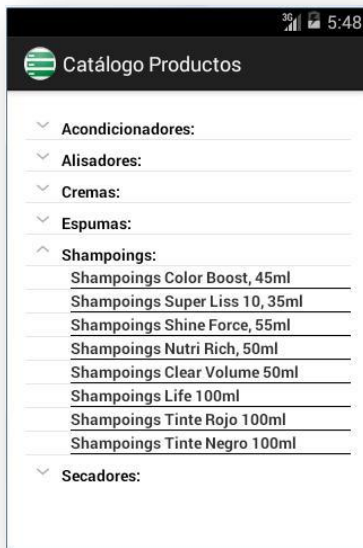


Figura 29: Catálogo de productos

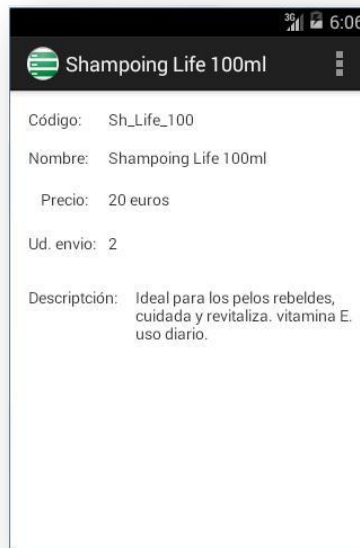


Figura 30: Detalle producto



Figura 31: Ver ofertas de un producto

Al seleccionar la opción ver ofertas, aparecerá el listado de las ofertas que ha tenido ese producto. Al seleccionar una oferta, se mostrarán sus detalles y, dependiendo de si la oferta está cerrada o en vigor, se verá un mensaje u otro:



Figura 32: Listado ofertas Producto

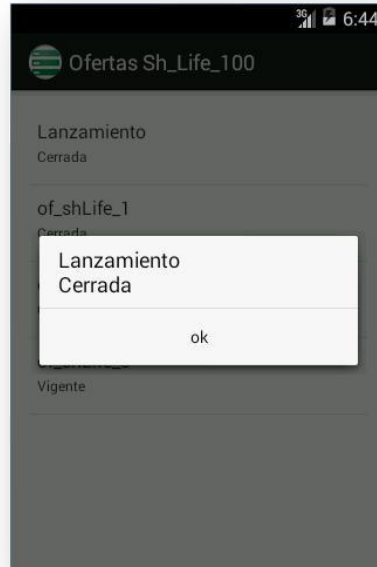


Figura 33: Oferta Cerrada



Figura 34 : Oferta Vigente

5.3.4-Confirmar pedidos

Además de realizar los pedidos, el comercial debe dar seguimiento a éstos llamando a los clientes que los han realizado y confirmando la recepción de la mercancía.

Esta opción es accesible desde la pantalla principal de la aplicación.

Al confirmar un pedido, ya comentamos que hay un proceso que se lanza para actualizar la estimación de stock de los productos pedidos.

Esta actividad mostrará los pedidos realizados y no confirmados. Al confirmar un pedido, se actualiza su estado y se lanza un servicio para actualizar las unidades de las que dispone el cliente.



Figura 35: Listado pedidos sin confirmar

Al seleccionar un pedido, se abrirá un dialogo para poderlo confirmar.

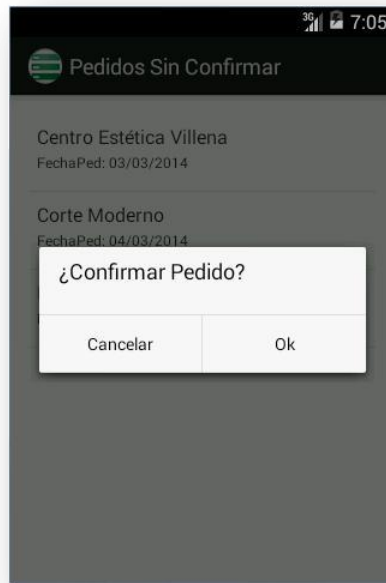


Figura 36: Confirmar un pedido ya realizado

5.3.5-Gestión material

El comercial debe disponer de muestras, de paquetes de bienvenida y catálogos impresos para ofrecerlos a sus clientes.

Por cada cliente nuevo que se le asigna al comercial, la empresa le manda un paquete de bienvenida y un catálogo impreso.

Si la empresa saca nuevos productos o nuevas variedades, el comercial, al concertar cita con sus clientes, debe llevar consigo el nuevo catálogo y las nuevas muestras.

Esta actividad permite gestionar el material del que dispone el comercial actualizando su stock y dándole opción a realizar un pedido.

Esta información se mandará a la empresa en un informe junto con las actividades que ha realizado el comercial.

Al igual que los productos, el material se organiza por categorías (catálogos, paquetes de bienvenida, muestras)



Figura 37: Listado del material

Al seleccionar un producto se mostrará su descripción y se dará opción a actualizar las unidades de las que dispone el comercial o a realizar un pedido.

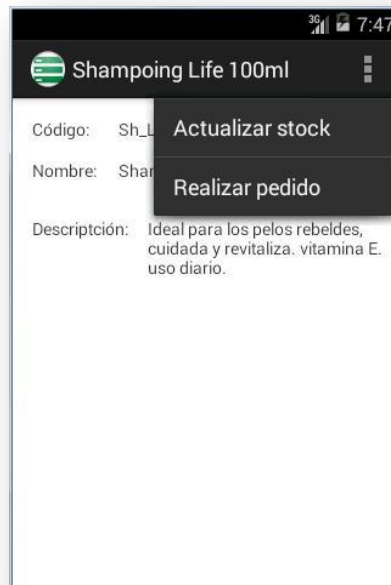


Figura 38: Opciones para gestionar el material

Para la realización de esta actividad, se ha reutilizado gran parte de la actividad de realización de pedido aprovechando así los fragmentos de las vistas ya definidos y reutilizando parte del código ya escrito. Al ser similares, no se mostrarán en este apartado.

5.3.6-Agenda

Esta actividad mostrará al usuario las citas que ha concertado con los clientes y le dará opción a calcular la ruta óptima para visitarlos partiendo de la ubicación actual del comercial.

Para realizar esta actividad, se hará uso del **API de Google Maps** para gestionar los mapas y la geolocalización.

También es necesario considerar que, al necesitar acceso a la geolocalización, hay que conceder a esta actividad los permisos adecuados en el fichero **AndroidManifest**.

La implementación de esta actividad se ha realizado con listas desplegables: las citas quedan ordenadas por día, y a su vez, por horas. Al lanzar la actividad, un proceso recorre las citas que tiene el comercial para, más tarde, alimentar las listas.

A continuación se muestra el aspecto que tiene la interfaz de esta actividad.



Figura 39: Agenda

Como se muestra en la figura, el usuario puede acceder a ver la ruta haciendo click en el icono situado junto con cada entrada. Si se selecciona la entrada, se abre una ventana mostrando los detalles de la cita y ofreciendo la posibilidad de cambiar el estado de la cita (realizada o cancelada).

A continuación se muestran las interfaces de las actividades involucradas:



Figura 40: Detalles de cita

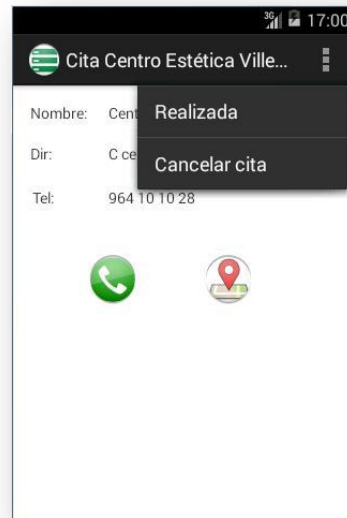


Figura 41: Menú detalles de cita

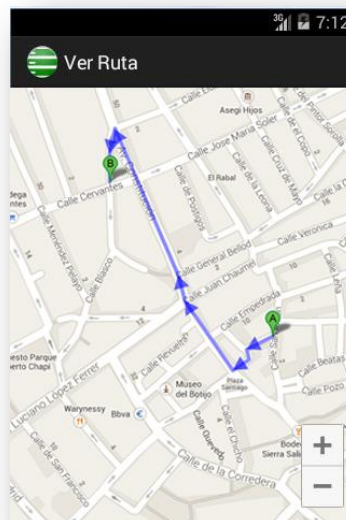


Figura 42: ver ruta

5.3.7- Mandar informe

Al final de cada día laboral, el comercial debe mandar a la empresa los pedidos que ha realizado, el inventario del material del que dispone y si ha realizado las citas con los nuevos clientes que le ha mandado la empresa.

Esta actividad solo consta de un *progressDialog* que va informando del progreso de la tarea de sincronización. Por detrás, hay un servicio que se conecta al web Service de la empresa y manda varios ficheros XML con la información de los pedidos realizados y las citas concertadas y el estado del material del que dispone el comercial hasta el momento.

El aspecto que tiene esta actividad se muestra a continuación.

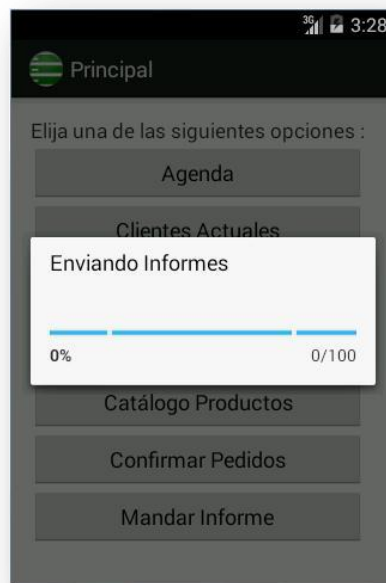


Figura 43: Mandar Informe

Capítulo 6: Mejoras Aplicables

Al iniciar la aplicación y después de identificarse, se ofrece al usuario la interfaz principal desde la cual puede acceder a las diferentes opciones, como la gestión del stock de sus clientes o confirmar los pedidos ya realizados. Este aspecto se podría mejorar automatizándolo y haciendo que cuando el usuario se identifique, corra un servicio que busque los clientes que tienen algún producto en ruptura de stock y le informe de estos clientes.

Del mismo modo, cada vez que se realiza un pedido, el comercial debe volver a ponerse en contacto con los clientes que lo hayan realizado para confirmarlo. Aquí también este proceso se podría hacer de forma que a los tres días laborales de haber realizado un pedido, se le muestra al usuario un recordatorio de los pedidos que tiene que confirmar.

Estos eventos se podrían comunicar al usuario usando los mecanismos que ofrece Android, *Toast* o *Notification*, e informando que se han producido en la barra de estado.

En Android una notificación puede adoptar varias formas, desde un mensaje emergente hasta el parpadeo del LED frontal del dispositivo o una vibración. Se podrían combinar estas utilidades para informar al usuario de estos eventos.

Ya vimos que el hecho de confirmar un pedido hace que se actualicen las unidades disponibles de cada producto pedido. Aquí también se puede aportar una mejora, haciendo que el cálculo de las unidades disponibles de los productos se haga de forma dinámica y no solo basándose sobre la estimación de consumo que comunica cada cliente al comercial.

La implementación de la agenda se ha hecho con listas desplegadas. Una posible mejora sería usar el *API* de *Google Calendar*, de esta forma el comercial podrá crear un calendario que lo puede sincronizar con varios dispositivos.

Vimos que el día del comercial finaliza enviando un comunicado a la empresa informándole de las actividades que ha realizado. Este proceso se podría programar para que se ejecute de forma automática e informar al usuario del resultado de la sincronización.

Y por último, comentar que lo que hace que una aplicación sea atractiva y amigable es su interfaz gráfica, este aspecto queda por mejorar para esta aplicación.

Capítulo 7: Conclusiones

En la actualidad se espera que los teléfonos móviles además de cumplir su labor, sean asistentes personales avisando al usuario de distintos eventos. En este proyecto hemos visto la plataforma Android y las posibilidades que ofrece.

Combinando todas las utilidades que ofrece Android, se puede desarrollar una aplicación muy potente y que ofrece al usuario apoyo y asistencia en sus tareas.

Con este proyecto, se ha descubierto el potencial de la plataforma Android y el SDK del que dispone, que es muy completo y cómodo para desarrollar aplicaciones.

La realización de este proyecto ha reforzado mis conocimientos en la programación orientada a objeto y también en los pasos previos al desarrollo como es la captura de los requerimientos de usuario y el modelado y la estructura del caso de estudio. Además de adentrarme en una tecnología nueva e incipiente

Apéndice

A.1-Definiciones y Acrónimos

- (1) **Kernel de Linux:** o núcleo de Linux, es la parte fundamental del sistema operativo destinada a gestionar los recursos y permite a los diferentes componentes tanto materiales como de aplicaciones de comunicarse entre sí.
- (2) **WebKit:** plataforma libre (licencia BSD y GNU LGPL) destinada a facilitar la integración de un motor de renderizado de páginas Web para las aplicaciones de tipo navegador. Es el núcleo del navegador incluido por defecto en Android.
- (3) **OpenGL:** conjunto normalizado de funciones de cálculo de imágenes en 2D o en 3D.
- (4) **SQLite:** sistema de gestión de bases de datos relacionales SQL, destaca por su sencillez de utilización, es de licencia libre y es perfectamente adaptada a ser utilizada en dispositivos de pocos recursos.
- (5) **Surface Manager:** Librería encargada de dar soporte a los diferentes elementos de navegación de la pantalla y las ventanas dibujadas en la misma.
- (6) **XML:** eXtensible Markup Language, lenguaje de marcas en forma legible, se caracteriza por sus etiquetas. Permite definir la gramática de lenguajes específicos.
- (7) **SOAP:** Simple Object Access Protocol, es un protocolo de (Remote Procedure Call) orientado objeto y se basa en XML. Permite la transmisión de mensajes entre objetos distantes llamando a métodos de un objeto remoto.
- (8) **SDK:** Software Development kit, es un conjunto de herramientas que permite desarrollar aplicaciones.

- (9) **API:** application programming interface, librerías que especifican como un componente de un software debe interactuar con otro. Están destinadas a facilitar la tarea de programación aportando rutinas y funciones.

A.2-Índice de Figuras

- Figura 1: Capas de la plataforma Android
- Figura 2: Diagrama de clases del caso de estudio
- Figura 3: Carpetas de un proyecto Android
- Figura 4: Relación entre actividad, vista y recursos
- Figura 5: Ciclo de vida de una actividad
- Figura 6: Ciclo de vida de los Servicios
- Figura 7: Árbol de actividades
- Figura 8: Interfaz de la actividad Login
- Figura 9: Acceso con datos incorrectos
- Figura 10: interfaz de la actividad principal
- Figura 11: Listado clientes actuales
- Figura 12: Detalle del cliente
- Figura 13: menú detalle del cliente
- Figura 14: Detalle de stock del cliente
- Figura 15: Actualización de los detalles de un producto
- Figura 16: Catálogo productos
- Figura 17: Detalle de una categoría
- Figura 18: Línea pedido
- Figura 19: Interfaz para concertar cita con el cliente
- Figura 20: Histórico de pedidos de un cliente
- Figura 21: Confirmar un pedido pendiente
- Figura 22: Listado citas concertadas con un cliente
- Figura 23: Confirmación de la realización de una cita
- Figura 24: No hay listado de clientes nuevos
- Figura 25: El comercial ha tratado todo el listado de los clientes nuevos

- Figura 26: listado de clientes nuevos
- Figura 27: Detalle cliente nuevo
- Figura 28: Menú cliente nuevo
- Figura 29: Catálogo de productos
- Figura 30: Detalle producto
- Figura 31: Ver ofertas de un producto
- Figura 32: Listado ofertas Producto
- Figura 33: Oferta Cerrada
- Figura 34: Oferta Vigente
- Figura 35: Listado pedidos sin confirmar
- Figura 36: Confirmar un pedido ya realizado
- Figura 37: Listado del material
- Figura 38: Opciones para gestionar el material
- Figura 39: Agenda
- Figura 40: Detalles de cita
- Figura 41: Menú detalles de cita
- Figura 42: ver ruta
- Figura 43: Mandar Informe

A.3-Bibliografía

- (2012) Lauren Darcey, Shane Conder, "Programación Android 4". Editorial Anaya.
- (2011) Frank Ableson, Charlie Collins, Robi Sen, "Android, guía para desarrolladores" Editorial Anaya.
- (2008) DiMarzio, J.F. "Android A Programmer's Guide". McGrawHill.
- (2009) Rogers, Rick. "Android Application Development". O'Reilly Series.
- (2009) Rogers, Reto. "Professional Android Application Development". Wrox.
- <http://developer.android.com/index.html>