Capítulo 2

MODELO DE REQUISITOS

2.1 Introducción

Un modelo, en el desarrollo de software, define cómo solucionar los problemas que aparecen en el desarrollo de una aplicación. Para desarrollar el software, existen diferentes metodologías, que definen los distintos tipos de modelos que se pueden encontrar en este proceso. Así, los modelos básicos son: **de Requisitos, de Diseño, de Implementación y de Pruebas** [9]. Asimismo, el desarrollo de software debe ser registrado a lo largo de todo el proceso, dando lugar a distintos documentos (**Documentación**). De todos los modelos indicados, en este capítulo se trataría el de requisitos, como primer modelo a definir en el desarrollo de la aplicación objeto de este trabajo fin de grado.

El **modelo de requisitos** tiene como objetivo delimitar el sistema y capturar la funcionalidad que debe ofrecer desde la perspectiva del usuario (es el contrato entre el desarrollador y el usuario final).

El modelo de requisitos consiste, básicamente, de tres modelos principales:

- **Modelo de comportamiento:** se basa directamente en el modelo de casos de uso y especifica la funcionalidad, desde el punto de vista del usuario. Tiene dos conceptos claves:
  - *Actores:* representan los distintos papeles que los usuarios pueden jugar en el sistema.
  - *Casos de uso:* representa qué pueden hacer los actores con respecto al sistema.
- **Modelo de presentación o de interfaces o de bordes:** especifica cómo interactúa el sistema con actores externos al ejecutar los casos de uso, es decir, especifica cómo se verán las interfaces gráficas y que función tiene cada una de ellas.
- **Modelo de información o del dominio del problema:** conceptualiza el sistema según los objetos que representan las entidades básicas de la aplicación.

El modelo de requisitos que se va a presentar en este capítulo, está basado en el *modelo de comportamiento* y en el *modelo de interfaces*. Asimismo, antes de realizar estos modelos, el desarrollador debe hacer una descripción detallada del problema (contrato con el usuario final).

2.2 Descripción del problema

La descripción del problema es una definición muy inicial de las necesidades que sirve como punto de partida para comprender los requisitos del sistema, es decir, debe ser una descripción de lo que se necesita y no una propuesta de solución.

En este trabajo fin de grado se pretende desarrollar una aplicación Android para pedir cita previa en peluquerías, tomando como ejemplo *Peluquerías Naranja*, una franquicia de peluquerías canarias. Esta aplicación permite, al usuario del dispositivo, reservar hora o pedir cita en la peluquería. La cita se reservará siguiendo una serie de pantallas, en las que la información introducida de forma táctil se almacenará posteriormente en una base de datos online. Asimismo, la aplicación será capaz de gestionar las citas previas disponibles permitiendo al usuario cancelar su cita previa.

La aplicación presentará una ventana principal con cuatro funciones: *Presentación de la peluquería*, en la que se describirá la peluquería brevemente; *Contacto*, donde se detallarán los datos de contacto de la empresa; *Pedir Cita*, donde se podrá pedir cita en la peluquería seleccionada, y *Gestión de citas*, donde se podrán gestionar las citas. Asimismo, también contará con un menú donde se podrá acceder a la configuración de la aplicación, ayuda, información y salir.

Una vez se ha pedido una cita con la aplicación, ésta queda almacenada en una base de datos online que la peluquería puede consultar y gestionar. La aplicación del terminal del usuario también tendrá acceso a dicha base de datos online, para poder gestionar las citas que ha realizado el usuario final.

2.3 Modelo de comportamiento

El **modelo de comportamiento** describe las diferentes formas de uso de un sistema, en el que cada uno de esas formas se conoce como caso de uso. Cada caso de uso se compone de una secuencia de eventos iniciada por el usuario. Para comprender los casos de uso del sistema, es necesario saber cómo los usuarios lo van a usar o actor (el actor no corresponde directamente con un usuario). Como se muestra en la figura 2.1, el actor y el caso de uso representan los dos elementos básicos de este modelo.

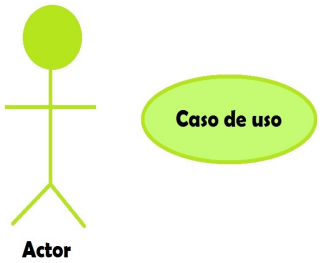

Figura 2.1: Representación de las entidades básicas para el modelo de comportamiento

2.3.1 Actores

Los **actores** [10] corresponden con el papel que un usuario puede jugar dentro de la aplicación (son entidades distintas a los usuarios). Los actores modelan cualquier entidad externa al sistema, además no están restringidos a ser personas físicas, pudiendo representar otros sistemas externos al actual. Cada uno de estos actores podrá ejecutar una o más tareas del sistema.

Los actores se identifican antes que los casos de uso, para que estos sean la herramienta principal para encontrar los casos de uso. Al definir todos los actores y los casos de uso se define la funcionalidad completa del sistema.

A la hora de definir los actores, se identifican primero aquellos que son la razón principal del sistema, conocidos como *actores primarios*, que son los que rigen la secuencia lógica de ejecución del sistema. Además existen otros actores que supervisan y mantienen el sistema, conocidos como *actores secundarios*. Estos corresponden, por lo general, a máquinas o sistemas externos.

Para especificar los actores de la aplicación, se pueden diferenciar los siguientes: el actor primario al que se le define como *Usuario*, que es el encargado de introducir los datos necesarios en la aplicación para que ésta le puedan proporcionar el servicio que proporciona, y varios actores secundarios, como son: el *Servidor de la base de datos externa*, que será el responsable de almacenar las citas previas; el *servidor Google*, que será el encargado de proporcionar los mapas, y el *GPS* que será el encargado de proporcionar la posición del usuario. En la figura 2.2 se presenta un diagrama representando al sistema como una caja cerrada y los diferentes actores como entidades externas a él.

2.3.2 Casos de uso

Después de haber definido los actores se define la funcionalidad del sistema a través de los casos de uso [10]. Cada caso de uso constituye un flujo completo de

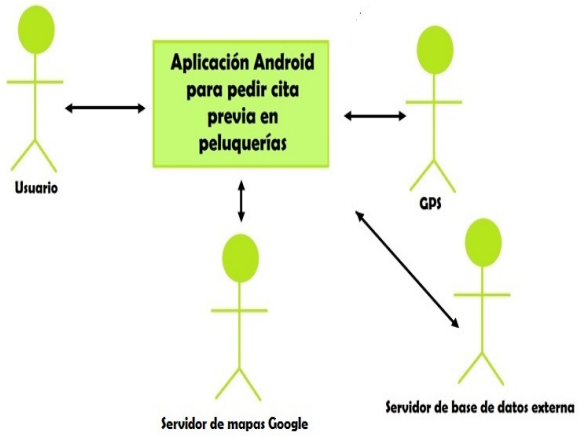

Figura 2.2: Delimitación del sistema para la *Aplicación Android para pedir cita previa en peluquerías*

eventos que muestra la interacción entre los actores y el sistema (es decir, qué van a hacer los actores). El actor primario es el encargado de empezar esta interacción y los casos de uso se muestran como respuesta al evento anterior. La ejecución del caso de uso acaba cuando algún actor genera un evento que requiere un caso de uso nuevo.

Aunque la idea es mantener el modelo de casos de uso lo más sencillo posible, existen ocasiones en los que la funcionalidad debe ser puesta en casos de uso separados o bien realizar una subdivisión del caso de uso. Existen dos enfoques para expresar variantes:

1. Si las diferencias entre los casos de uso son pequeñas, se pueden definir subflujos separados dentro de un mismo caso de uso.
2. Si las diferencias entre los casos de uso son grandes, se deben describir como casos de uso separados. Para estos casos de uso se utilizan principalmente las relaciones de inclusión, extensión y generalización.

Para la aplicación de este trabajo fin de grado la figura 2.3 refleja tres casos de uso principales (*Mostrar Peluquería*, *Pedir Cita* y *Mostrar citas*) y el resto se consideran casos de uso secundarios.

2.3.2.1. Inclusión

La **inclusión** se define como una sección de un caso de uso que es parte obligatoria del caso de uso básico. El caso de uso que se va a insertar depende del caso

de uso anterior. Como se muestra en la figura 2.3, la notación para inclusión es la etiqueta «include».

En la aplicación a desarrollar (figura 2.3), el caso de uso *Manejo de la base de datos externa* está incluido dentro de los casos de uso *Mostrar Peluquerías*, *Mostrar Horarios Disponibles*, *Mostrar citas* y *Gestión citas* porque todos ellos necesitan cargar o modificar datos de la base de datos externa. Asimismo, el caso de uso *Mostrar Mapa* también está incluido en *Mostrar Peluquerías* porque el mapa se muestra después de haber presentado la información de la peluquería seleccionada. Finalmente, los casos de uso *Mostrar peluquerías* y *Mostrar Horarios Disponibles* también están incluidos dentro del caso de uso *Pedir cita*, ya que este caso de uso se encargará de ofrecer al usuario la posibilidad de introducir datos, elegir peluquería y elegir un horario para poder posteriormente pedir cita.

2.3.2.2. Extensión

La **extensión** se utiliza para modelar secuencias de eventos opcionales de casos de uso que, al manejarse de manera independiente, pueden ser insertados o eliminados. Especifica cómo un caso de uso puede insertarse en otro para extender la función del anterior. El caso de uso donde se va a insertar la nueva funcionalidad debe ser independiente del caso de uso insertado. El caso de uso original se ejecuta hasta donde se inserta el nuevo caso de uso. Después de que este nuevo caso de uso haya terminado su función, el curso original de la secuencia continúa como si nada hubiera ocurrido.

Como se muestra en la figura 2.3, la notación para extensión es la etiqueta «extend». En esta figura se puede observar que el caso de uso *Mostrar Mapa* se extiende mediante el caso de uso *Mostrar localización usuario*, ya que una vez elegido el mapa se le da al usuario la libertad de poder ver un mapa con la peluquería más cercana a su posición. Otro ejemplo de extensión se observa en los casos de uso *Mostrar citas* y *Pedir citas* que extienden el caso de uso *Gestión de citas*, ya que en ambos se podrá realizar alguna acción sobre las citas.

2.3.2.3. Generalización

Una relación adicional entre casos de uso es la **generalización** que apoya la reutilización de los casos de uso. Mediante esta relación es necesario describir las partes similares una sola vez, en lugar de repetirlas para todos los casos de uso de comportamiento común. En la generalización hay dos tipos de casos de uso:

- A los casos de uso que se extraen se les llama casos de uso *abstractos*, ya que sirven para describir partes que son comunes a otros casos de uso (no se instancian).

- A los casos de uso que realmente son instanciados se les conoce como casos de uso *concretos*.

Las descripciones de los casos de uso abstractos se incluyen en las descripciones de los casos de uso concretos. Los casos de uso abstractos también pueden ser usados por otros casos de uso abstractos. Normalmente los comportamientos similares entre casos de uso se identifican después de describir los casos de uso, aunque en algunos casos es posible identificarlos antes. En la aplicación a desarrollar, no existen casos de uso que usen esta relación.

En la figura 2.3 se muestra el diagrama completo de los casos de uso para la aplicación a desarrollar. Los casos de uso principales son: *Mostrar Peluquerías, Pedir cita* y *Mostrar citas*. Además de éstos, los casos de uso adicionales son la inclusión o extensión de los casos de uso *Mostrar Horarios Disponibles, Gestión citas, Mostrar Mapa, Mostrar localización usuario* y *Manejo base de datos externa*.

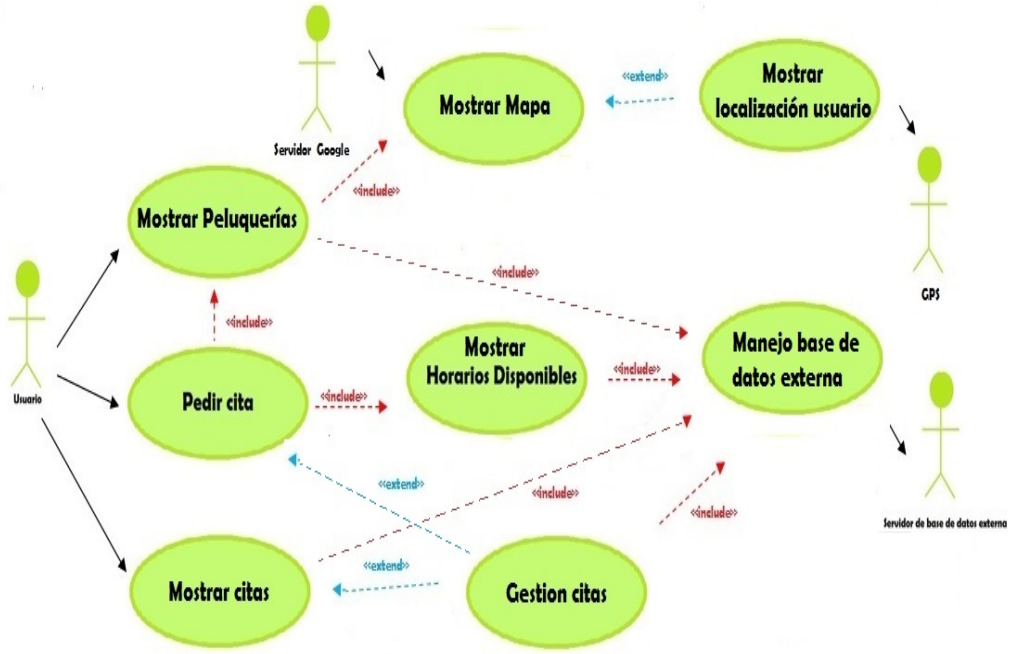

Figura 2.3: Diagrama de casos de uso para la *Aplicación Android para pedir cita previa en peluquerías*

2.4 Modelo de interfaces

El modelo de interfaces describe la presentación de información entre los actores y el sistema. Para ello, se especifica cómo se verán las interfaces de usuario al ejecutar

cada uno de los casos de uso, lo cual ayuda al usuario a visualizarlos según sean mostrados por el sistema. Cuando se diseñan las interfaces de usuario, es esencial tener a los usuarios involucrados, siendo de máxima importancia que las interfaces reflejen la visión lógica del sistema. En la sección 2.5.2 se muestran las interfaces usadas en la herramienta diseñada en este trabajo fin de grado.

2.5 Actores y casos de uso de la Aplicación Android para pedir cita previa en peluquerías

En esta sección se muestra la documentación de los actores y casos de uso, junto con el diseño de las interfaces, que serán usadas como prototipo del sistema.

2.5.1 Actores

Se describen un total de cuatro actores en la aplicación: *Usuario, Servidor de base de datos externa, Servidor Google* y *GPS*.

2.5.1.1. Usuario

Representa al usuario de la aplicación, es de tipo primario y participa en todos los casos de uso de la aplicación: *Mostrar Peluquerías*, *Pedir Cita*, *Mostrar Citas*, *Mostrar Horarios Disponibles*, *Gestión citas*, *Mostrar Mapa*, *Mostrar localización usuario* y *Manejo base de datos externa*

2.5.1.2. Servidor de la base de datos Externa

Representa el servidor externo donde existe una base de datos con las citas y los horarios disponibles, es de tipo secundario y participa en el caso de uso *Manejo de base de datos externa*.

2.5.1.3. Servidor Google

Representa el servidor de mapas de Google que nos proporciona los mapas utilizados en la aplicación, es de tipo secundario y participa en el caso de uso *Mostrar Mapa*.

2.5.1.4. GPS

Representa al servicio GPS para la geo-localización del usuario, es de tipo secundario y participa en el caso de uso *Mostrar localización usuario*.

2.5.2 Casos de Uso

Se describen un total de ocho casos de uso en la aplicación: *Mostrar Peluquerías*, *Pedir cita*, *Mostrar citas*, *Mostrar Horarios Disponibles*, *Gestión citas*, *Mostrar Mapa* ,*Mostrar localización usuario* y *Manejo base de datos externa*.

2.5.2.1. Caso de uso *Mostrar Peluquerías*

- **Actores:** *Usuario*.
- **Propósito:** mostrar al usuario final información de la peluquería elegida, imágenes y la posibilidad de ver un mapa con la ubicación de la misma.
- **Precondiciones:** el usuario debe haber presionado el botón **¿Dónde estamos?**, del menú principal.
- **Flujo principal:** Se presenta al usuario la vista principal, figura 2.4 donde el usuario selecciona la opción **¿Dónde estamos?** En este punto, se inicia este caso de uso representado en la figura 2.5 (E-1).
- **Flujo secundario:** una vez dentro de este caso de uso, figura 2.5, se puede diferenciar dos flujos secundarios. El primer flujo secundario es la carga de los datos de la base de datos externa acción del caso de uso *Manejo base de datos externa* y el segundo flujo secundario es la vista de un mapa con la ubicación de la peluquería gracias al caso de uso *Mostrar Mapa* que se inicia al presionar el botón **Ver en mapa**.
- **Excepciones:**
  - E-1 (Fallo al conectar con el servidor de base de datos externa): debe de conectarse al servidor externo para cargar los datos de las peluquerías.

Figura 2.4: Vista principal propuesta de la aplicación

Figura 2.5: Vista donde se muestra la peluquería elegida

2.5.2.2. Caso de uso *Mostrar Mapa*

- **Actores:** *Usuario* y *Servidor Google*.
- **Propósito:** mostrar al usuario un mapa con la ubicación de la peluquería previamente seleccionada.

- **Precondiciones:** el usuario debe tener una peluquería seleccionada y presionar en el botón **Ver en mapa** que se encuentra en la vista de la figura 2.5.
- **Flujo principal:** se presenta al usuario la vista principal (figura 2.4), donde el usuario selecciona la opción **¿Dónde estamos?** En este punto se inicia el caso de uso *Mostrar peluquerías* y se presenta la vista de la figura 2.5. En esta vista, si el usuario selecciona el botón **Ver en mapa** se inicia el caso de uso *Mostrar mapa* (E-1), representado en la figura 2.6.
- **Flujo secundario:** si el usuario presiona el botón **Peluquería más cercana**, se cambia el mapa y se muestra la peluquería más cercana a la posición del usuario, con la ayuda del caso de uso *Mostrar localización usuario*.
- **Excepciones:**
  - E-1 (Fallo al conectar con el servidor Google): debe de conectarse al servidor Google para cargar el mapa.

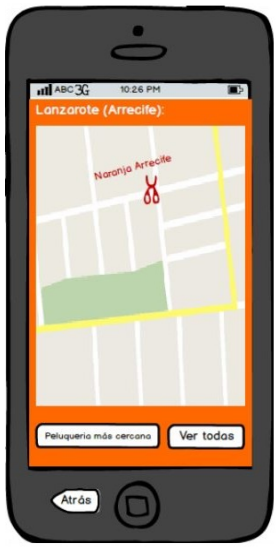

Figura 2.6: Vista del mapa con la peluquería elegida

2.5.2.3. Caso de uso *Mostrar localización usuario*

- **Actores:** *Usuario* y *GPS*.
- **Propósito:** obtener la localización geográfica del usuario.
- **Precondiciones:** el usuario debe presionar en el botón **Peluquería más cercana** que se encuentra dentro del caso de uso *Mostrar mapa* (figura 2.6).

- **Flujo principal:** se presenta al usuario la vista principal (figura 2.4) donde el usuario selecciona la opción **¿Dónde estamos?**, en este paso se inicia el caso de uso *Mostrar peluquerías* (figura 2.5). En esta vista, si el usuario selecciona el botón **Ver en mapa** se inicia el caso de uso *Mostrar mapa*, representado en la figura 2.6. Finalmente, al presionar el botón **Peluquería más cercana**, se inicia este caso de uso, de forma que se pedirá al dispositivo GPS (E-1), la localización del dispositivo del usuario.
- **Flujo secundario:** si el usuario presiona el botón **Ver todas** se muestra un mapa con todas las peluquerías.
- **Excepciones:**
  - E-1 (Fallo al conectar con el GPS): debe de conectar con el GPS para saber la geo-localización del usuario.

2.5.2.4. Caso de uso *Pedir cita*

- **Actores:** *Usuario*.
- **Propósito:** recoger los datos necesarios y pedir cita previa en la peluquería seleccionada.
- **Precondiciones:** ninguna.
- **Flujo principal:** se presenta al usuario la vista principal (figura 2.4). En esta vista, el usuario selecciona el botón **Pedir cita**, iniciando este caso de uso y apareciendo la vista de la figura 2.7, donde el usuario deberá introducir los datos solicitados (E-1).
- **Flujo secundario:** se pueden considerar dos flujos secundarios. Un flujo a través del cual se cargan las peluquerías utilizando el caso de uso *Mostrar peluquerías* y otro que carga los horarios disponibles utilizando el caso de uso *Mostrar Horarios Disponibles*.
- **Excepciones:**
  - E-1 (Fallo al introducir datos): debe introducir correctamente todos los datos (con el formato adecuado).

2.5.2.5. Caso de uso *Mostrar Horarios Disponibles*

- **Actores:** *Usuario* y *Servidor de base de datos externa*.
- **Propósito:** recoger de la base de datos externa y mostrar en pantalla los horarios disponibles en la peluquería seleccionada.

Figura 2.7: Vistas correspondientes a introducir datos para pedir cita

- **Precondiciones:** el usuario debe haber seleccionado una peluquería dentro del caso de uso *Pedir cita* y encontrarse en la ventana donde se introduce el horario.
- **Flujo principal:** se presenta al usuario la vista principal (figura 2.4). En esta vista, el usuario selecciona el botón **Pedir cita**, iniciando el caso de uso *Pedir cita* (vista de la figura 2.7). Como se puede observar en esta figura, en la última vista se debe introducir un horario que ha sido previamente cargado gracias al caso de uso *Mostrar Horarios Disponibles*.
- **Flujo secundario:** el flujo secundario de este caso de uso *Mostrar Horarios Disponibles*, es el encargado de cargar los horarios de la base de datos externa y es posible gracias a la ayuda del caso de uso *Manejo base de datos externa*.
- **Excepciones:** ninguna.

2.5.2.6. Caso de uso *Mostrar citas*

- **Actores:** *Usuario*.
- **Propósito:** mostrar una lista con las citas previas que ha pedido el usuario de la aplicación.

- **Precondiciones:** el usuario debe haber pedido previamente una cita y haber presionado el botón **Mis citas**.
- **Flujo principal:** se presenta al usuario la vista principal (figura 2.4). En esta vista, el usuario selecciona el botón **Mis citas**, iniciando este caso de uso representado en la figura 2.8.
- **Flujo secundario:** existen dos flujos secundarios, uno que carga las citas previamente solicitadas gracias a la ayuda del caso de uso *Manejo base de datos externa* y otro al que se accede presionando en una cita y que permite cancelarla, iniciando el caso de uso *Gestión Citas*.
- **Excepciones:** ninguna.

Figura 2.8: Vista con el listado de citas para la posterior gestión de las mismas

2.5.2.7. Caso de uso *Gestión citas*

- **Actores:** *Usuario*.
- **Propósito:** es el encargado de gestionar las citas, añadir o eliminar citas, de la base de datos externa.
- **Precondiciones:** ninguna.
- **Flujo principal:** se presenta al usuario la vista principal (figura 2.4). En esta vista, el usuario selecciona el botón **Mis citas**, iniciando el caso de uso *Mostrar citas*, representado en la figura 2.8. Al presionar en cualquier cita de la lista, aparece la opción de **Cancelar la cita**, como se puede observar en la

figura 2.9. Al presionar el botón **Cancelar cita**, se inicia este caso de uso que elimina la cita de la base de datos (con la ayuda del caso de uso *Manejo base de datos externa*).

Figura 2.9: Vista de la aplicación que permite cancelar una cita

También se puede acceder a este caso de uso si en la vista principal, figura 2.4, el usuario selecciona el botón **Pedir cita**. Cuando se introducen todos los datos necesarios se puede confirmar la cita tal y como se representa en la figura 2.10. Si se presiona el botón **Confirmar reserva** se inicia este caso de uso para añadir la cita a la base de datos externa.

- **Flujo secundario:** ninguno.
- **Excepciones:** ninguna.

2.5.2.8. Caso de uso *Manejo base de datos externa*

- **Actores:** *Usuario* y *Servidor de base de datos externa.*
- **Propósito:** manejar los datos de la base de datos externa, es decir, escribir, leer y borrar datos en el servidor externo.
- **Precondiciones:** se necesita interactuar con la aplicación ya sea mostrando información de una peluquería, pidiendo una cita previa o mostrando las citas previas.
- **Flujo principal:** este caso de uso tiene varios flujos principales dependiendo del uso que se le de a la aplicación:

Figura 2.10: Vista de la aplicación que permite confirmar una cita

Si se desea mostrar una peluquería: se presenta al usuario la vista principal (figura 2.4). El usuario selecciona el botón **¿Dónde estamos?**, en este paso se inicia el caso de uso *Mostrar Peluquerías*, representado en la figura 2.5. En este punto, se inicia el caso de uso *Manejo base de datos externa* para cargar la lista de peluquerías y la información de la peluquería seleccionada (E-1).

Si se desea pedir cita previa: se presenta al usuario la vista principal (figura 2.4). El usuario selecciona el botón **Pedir cita**, en este paso se inicia el caso de uso *Pedir cita*, cuya finalidad principal es pedir cita previa que se realiza iniciando el caso de uso *Manejo base de datos externa*, para escribir en la base de datos externa (E-1).

Si se desea mostrar las citas previas: se presenta al usuario la vista principal (figura 2.4). El usuario selecciona el botón **Mis citas**, en este paso se inicia el caso de uso *Mostrar citas* cuya finalidad principal es mostrar las citas previas y para ello se debe iniciar el caso de uso *Manejo base de datos externa*, para leer las citas de la base de datos externa (E-1).

- **Flujo secundario:** este caso de uso también puede ser iniciado por los casos de uso *Mostrar Horarios Disponibles* con el objetivo de leer los horarios disponibles en la base de datos externa y *Gestión citas* con la finalidad de borrar una cita de la base de datos externa (E-1).

- **Excepciones:**

  - E-1 (Fallo al conectar con la base de datos externa): debe conectar con el servidor externo para solicitar, borrar o escribir datos en la base de datos externa.

2.6 Clases e interfaces de la vista

Teniendo en cuenta que a la hora de implementar la *Aplicación Android para pedir cita previa en peluquerías* se utilizará la arquitectura, MVP (*Modelo-Vista-Presentador*), que es una variante de la arquitectura MVC, se detallan a continuación, las clases e interfaces más importantes que corresponden con la parte de la vista, explicando cada método y los parámetros de éstos, tal y como representa la figura 2.11.

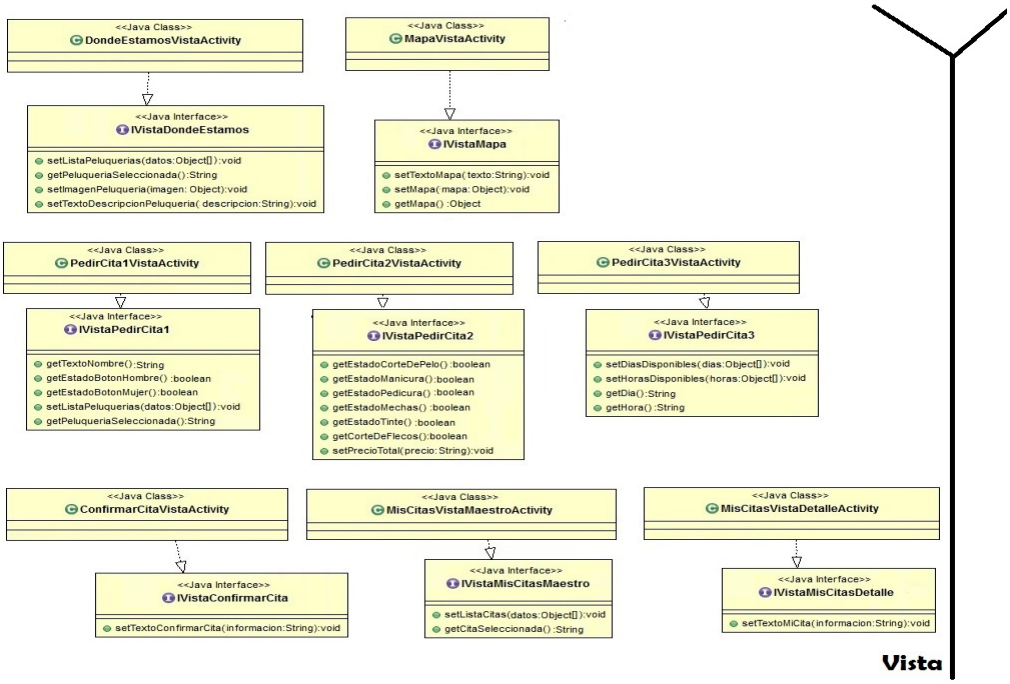

Figura 2.11: Clases de la vista de la aplicación con sus interfaces

2.6.1 Clase DondeEstamosVistaActivity

Vista correspondiente al caso de uso *Mostrar Peluquerías*, en la que el usuario debe seleccionar una peluquería y seguidamente se muestra una foto y una descripción de la peluquería seleccionada, como se observa en la figura 2.5.

Esta clase implementa la interfaz *IVistaDondeEstamos*, la cual aparece representada en el figura 2.11 con los siguientes métodos:

- **setListaPeluquerias(datos: Object[]): void**. Método que rellenará una lista con el listado de peluquerías que entra por parámetros.

- **getPeluqueriaSeleccionada(): String**. Método que obtiene la peluquería que el usuario ha seleccionado en la lista de peluquerías de la vista.
- **setImagenPeluqueria(imagen: Object): void**. Método encargado de cargar la imagen de la peluquería en la vista.
- **setTextoDescripcionPeluqueria(descripcion: String): void**. Método que actualiza la descripción de la peluquería.

2.6.2 Clase MapaVistaActivity

Vista en la cual aparece un mapa con un campo de texto en la parte superior y botones en la parte inferior, como se puede ver en la figura 2.6. La vista se mostrará cuando el usuario quiera ver un mapa con la ubicación de una peluquería.

Esta clase implementa la interfaz *IVistaMapa*, la cual aparece representada en el figura 2.11 con los siguientes métodos:

- **setTextoMapa(texto: String):void**. Método que actualizará el texto de la parte superior del mapa.
- **setMapa(mapa: Object): void**. Método encargado de actualizar el mapa en la vista.
- **getMapa(): Object**. Método que obtiene el mapa la vista.

2.6.3 Clase PedirCita1VistaActivity

Primera vista donde el usuario debe introducir datos necesarios para posteriormente generar una petición de cita previa. Algunos de los datos que se recogen en esta vista son el nombre, el sexo y la peluquería a la que desea acudir.

Esta clase implementa la interfaz *IVistaPedirCita1*, la cual aparece representada en el figura 2.11 con los siguientes métodos:

- **getTextoNombre(): String**. Método que recoge el nombre del usuario.
- **getEstadoBotonHombre(): boolean**. Método que recoge el estado del botón **Hombre** (si está seleccionado el usuario es un hombre).
- **getEstadoBotonMujer(): boolean**. Método que recoge el estado del botón **Mujer** (si está seleccionado el usuario es una mujer).
- **setListaPeluquerias(datos: Object[]): void**. Método que rellenará una lista con el listado de peluquerías que entra por parámetros.
- **getPeluqueriaSeleccionada(): String**. Método que obtiene la peluquería que el usuario ha seleccionado en la lista de peluquerías de la vista.

2.6.4 Clase PedirCita2VistaActivity

Segunda vista donde el usuario debe introducir datos necesarios para posteriormente generar una petición de cita previa. En esta vista se recogen los servicios que desea el usuario contratar en la peluquería.

Esta clase implementa la interfaz *IVistaPedirCita2*, la cual aparece representada en el figura 2.11 con los siguientes métodos:

- **getEstadoCorteDePelo(): boolean**. Método que recoge el estado del servicio corte de pelo. Si esta seleccionado el usuario quiere contratar dicho servicio.
- **getEstadoManicura(): boolean**. Método que recoge el estado del servicio manicura. Si esta seleccionado el usuario quiere contratar dicho servicio.
- **getEstadoPedicura(): boolean**. Método que recoge el estado del servicio pedicura. Si esta seleccionado el usuario quiere contratar dicho servicio.
- **getEstadoMechas(): boolean**. Método que recoge el estado del servicio mechas. Si esta seleccionado el usuario quiere contratar dicho servicio.
- **getEstadoTinte(): boolean**. Método que recoge el estado del servicio tinte. Si esta seleccionado el usuario quiere contratar dicho servicio.
- **getCorteDeFlecos(): boolean**. Método que recoge el estado del servicio corte de flecos. Si esta seleccionado el usuario quiere contratar dicho servicio.
- **setPrecioTotal(precio: String): void**. Método que actualizará el precio total según lo que el usuario haya contratado.

2.6.5 Clase PedirCita3VistaActivity

Vista donde el usuario debe elegir una fecha y una hora en la que quiere acudir a la peluquería.

Esta clase implementa la interfaz *IVistaPedirCita3* la cual aparece representada en el figura 2.11 con los siguientes métodos:

- **setDiasDisponibles(dias: Object[]): void**. Método que actualizará en la vista los días disponibles para pedir cita en la peluquería seleccionada.
- **setHorasDisponibles(horas: Object[]): void**. Método que actualizará en la vista las horas disponibles para pedir cita en la peluquería seleccionada.
- **getDia(): String**. Método que recoge el día que el usuario ha seleccionado en la vista.
- **getHora(): String**. Método que recoge la hora que el usuario ha seleccionado en la vista.

2.6.6 Clase ConfirmarCitaVistaActivity

Vista en la que aparece un texto con todos los datos recopilados de la cita previa, donde el usuario debe confirmar todos los datos que ha introducido y seguidamente se genera una petición de cita previa.

Esta clase implementa la interfaz *IVistaConfirmarCita*, la cual aparece representada en el figura 2.11 con el método:

- **setTextoConfirmarCita(informacion: String): void**. Método que actualizará en la vista el texto que aparece con todos los datos que el usuario introdujo previamente.

2.6.7 Clase MisCitasVistaMaestroActivity

Vista en la que aparece un listado con todas las citas que el usuario ha pedido previamente con dicha aplicación.

Esta clase implementa la interfaz *IVistaMisCitasMaestro*, la cual aparece representada en el figura 2.11 con los siguientes métodos:

- **setListaCitas(datos: Object[]): void**. Método que actualizará en la vista, la lista con todas las citas que el usuario ha pedido previamente.
- **getCitaSeleccionada(): String**. Método que recoge la cita que el usuario ha seleccionado.

2.6.8 Clase MisCitasVistaDetalleActivity

Vista donde aparecen los detalles de la cita seleccionada y un botón para cancelar la reserva en la parte inferior, por si desea cancelar la cita.

Esta clase implementa la interfaz *IVistaMisCitasDetalle*, la cual aparece representada en el figura 2.11 con el método:

- **setTextoMiCita(informacion: String): void**. Método que actualizará en la vista el texto que aparece con los datos de la cita que el usuario ha seleccionado previamente.