Universidad Carlos III de Madrid

# Proyecto Final de Carrera

## Autenticación Biométrica Continua mediante reconocimiento Facial para dispositivos móviles Android

Autor: Alfredo Ruiz Hernández.

Tutor: Lorena González Manzano

Co-tutor: José María de Fuentes

Leganés, Octubre 2015

# Agradecimientos

Quiero agradecer a mis padres todo el apoyo recibido, sin ellos no hubiera sido posible terminar la carrera y ni mucho menos la realización de este trabajo fin de grado, su apoyo ha sido incondicional y siempre han estado para todo lo que he necesitado.

También agradecer a Lorena estar siempre dispuesta para ayudarme en todos los problemas que me han surgido durante el desarrollo de este trabajo final de grado.

# Resumen

La seguridad informática supone actualmente una vía de constante investigación y desarrollo, siendo los métodos de autenticación uno de los aspectos que siempre ha preocupado a los ingenieros de seguridad. En los últimos años el aumento del número de dispositivos móviles ha provocado el surgimiento de nuevas formas de autenticación y es en este punto donde se desarrolla el presente proyecto final de carrera.

En la actualidad podemos comprobar cómo los dispositivos móviles disponen de mecanismos de autenticación biométrica, ya sea por reconocimiento por huella dactilar o mediante reconocimiento facial. Pero estos mecanismos de autenticación actualmente vienen como alternativa al tradicional patrón de desbloqueo o el código alfanumérico que se suele introducir cada vez que se quiere desbloquear el teléfono. Esto supone que no se pueda realizar una protección del terminal a nivel de aplicaciones, es decir, proteger las aplicaciones instaladas en el dispositivo que el usuario elija, además presentan otro problema, y es que, cuando el terminal queda desbloqueado, un atacante puede utilizar el terminal siempre y cuando el dispositivo no entre en suspensión.

Ante esta situación, este proyecto plantea el análisis, diseño y construcción de una aplicación para dispositivos móviles con sistemas operativos Android que:

- Permita al usuario elegir que aplicaciones de las instaladas en su dispositivo desea proteger.
- Realizar autenticación biométrica de rasgos faciales sobre las aplicaciones que el usuario elija.
- Realizar dicha autenticación de forma continua, cada vez que el usuario desee abrir una aplicación protegida aunque el dispositivo no entre en suspensión.

# Índice

# Índice de Figuras

# Índice de Tablas

# CAPÍTULO 1

## 1. Introducción y objetivos

# 1 .1 Introducción

En la actualidad vivimos en la era de la información y la comunicación y como tal, los dispositivos móviles representan una parte fundamental.

Mediante los dispositivos móviles podemos comunicarnos de distintas formas posibles, tenemos acceso a todas las redes sociales mediantes sus respectivas aplicaciones (en adelante, se denominarán apps), y existen en el mercado múltiples aplicaciones de mensajería. Esto supone que en muchos casos, los dispositivos móviles constituyan una parte fundamental de nuestras vidas.

Según las últimas estimaciones se prevé que en 2015 llegaremos a los 7600 millones de móviles superando el total de la población, según el informe Mobility de Ericsson. En Europa, el número de suscripciones de «Smartphone» rondará los 765 millones en 2019, por lo que también superará la población de este continente. **El 65% de todos los teléfonos vendidos en el primer trimestre de 2014 son Smartphone.** [1]

Además, se distinguen dos sistemas operativos muy significativos, Android (de Google) y IOS (de Apple), siendo Android el más representativo pues dispone de una cuota de mercado del 81,5 % en 2014. [2]

Por otro lado el número de aplicaciones que podemos descargar e instalar en nuestro teléfono Android en agosto de 2014 era de 1.300.000. [3].

De todas estas aplicaciones muchas de ellas contienen información privada. Por este motivo junto con la necesidad de privacidad y seguridad de los datos, obligan a establecer medidas de autentificación para asegurar que ninguna otra persona distinta a la autorizada accede a datos ajenos o servicios privados.

La cantidad de datos e información privada que almacenamos en nuestros dispositivos mediante aplicaciones es cada vez mayor, por ello cada vez es se hace necesaria la existencia de aplicaciones de seguridad que protejan la información privada. En concreto, muchas de estas aplicaciones se caracterizan por facilitar la autenticación de los usuarios y hay múltiples formas de conseguirlo. Dentro de la seguridad, se distinguen 3 tipos básicos de autenticación mediante:

- Algo que se sabe, por ejemplo, contraseña, patrón de desbloqueo, pregunta clave, etc.
- Algo que se tiene, por ejemplo, una tarjeta de coordenadas.
- Algo que se es, por ejemplo, el iris del ojo, la huella dactilar, etc.[4]

En el primero de ellos (Algo que se sabe) el usuario debe dar su beneplácito y debe recordar algo, ya sea un patrón de desbloqueo o una contraseña. Es muy frecuente que las personas tengamos que recordar varias contraseñas distintas, lo que provoca que en muchas ocasiones se nos olvide alguna o confundamos una con otra. También es frecuente que las personas utilicen una única contraseña para todas sus cuentas, esto supone que en caso de que alguien pueda obtener la contraseña, tendría acceso a todas ellas.

El segundo método (Algo que se tiene) requiere que el usuario tenga en todo momento el objeto que sirva para autenticarse. Si el objeto se pierde, el usuario no podrá autenticarse. Además si el objeto llega a manos de un atacante este podría suplantar la identidad del usuario.

El tercer método (Algo que es) presenta una gran ventaja frente a los dos métodos anteriores, este método no requiere que el usuario tenga que recordar nada y además tampoco tiene que tener ningún objeto con el que autenticarse. Dado que la autenticación se basa en componentes propios del usuario, como por ejemplo el iris de la cara, no es necesario que recuerde ningún tipo de contraseña ni tampoco es necesario que guarde ningún tipo de objeto de autenticación. Este método tiene como inconveniente que requiere un alto componente tecnológico para su correcto funcionamiento.

La mayoría de aplicaciones que requieren de autenticación, utilizan el primer y el segundo método (algo que se tiene o algo que se sabe). Dada la necesidad de tener contraseña y usuario en todas ellas, conviene proporcionar mecanismos de autenticación que faciliten la autenticación al usuario. En este aspecto, la biometría tercer método) juega un papel fundamental. [5]

Existen aplicaciones que disponen de autenticación mediante técnicas biométricas. Con las nuevas versiones del sistema operativo Android se puede activar la autenticación biométrica para acceder al dispositivo, con ello podemos evitar el recordar e introducir un patrón o contraseña. Pero mediante estas aplicaciones o con el propio mecanismo que dispone Android de autenticación biométrica no se consigue mantener una autenticación continua, es decir, garantizar la autenticación del usuario a lo largo de todo el tiempo de uso. En concreto es de interés el hecho de autenticar al usuario en cada momento que abre una aplicación, ej. Facebook [6], pues de este modo hay seguridad de que sólo el usuario correcto está accediendo y no se ha dejado el móvil, por ejemplo, desbloqueado.

Mediante el proyecto que se desarrolla en este documento, se pretende construir una aplicación de autenticación biométrica de forma continua, que resuelve tanto los problemas que presentaban los métodos de autenticación tradicionales, con respecto a mantener la autenticación de forma continua mientras que el teléfono esté

en activo. Es importante destacar que al igual que otros métodos de autenticación, se requiere que el usuario introduzca, la primera vez que se ejecute la aplicación desarrollada y una única vez, una contraseña que se utilizará como parte del proceso de reconocimiento facial. Esta contraseña se cifra mediante una clave generada a partir de los rasgos faciales del usuario. Es importante considerar que dicha clave no se almacena y depende de los rasgos de cada usuario.

La legislación española ha desarrollado una serie de normativas en relación al tratamiento de datos de carácter personal, incluyendo en ellos los datos biométricos. Esta normativa está basada en la Ley Orgánica de protección de datos y en la cual se trata a los datos biométricos como datos personales a los que se aplicaría la legislación pertinente. Es por tanto necesario comunicar al usuario que se van a utilizar sus datos de carácter personal, que en este caso sería los datos biométricos. [35]

La aplicación desarrollada recibe el nombre de **CADroid.**

## 1.2 Motivación

Según los datos de dispositivos móviles que están reflejados en la sección anterior, se puede asegurar que los móviles constituyen una herramienta de uso cotidiano. En ellos almacenamos información de carácter privado. Así mismo, también lo utilizamos para comunicarnos mediante sistemas de mensajería, aplicaciones de redes sociales o mediante el correo electrónico.

En numerosas ocasiones los dispositivos móviles son un blanco fácil para robos o suplantaciones de identidad. Son fáciles de perder. Este hecho unido a la cantidad de información relevante que almacenamos en el dispositivo nos lleva a la necesidad de disponer de herramientas de seguridad que protejan dicha información.

La mayoría de aplicaciones que disponen de autenticación sólo la realizan la primera vez que el usuario accede a ellas. Por ejemplo, la aplicación de Facebook solo pide al usuario introducir la contraseña la primera vez que instala la aplicación. Si un atacante se hace con el dispositivo de otra persona podría acceder a Facebook sin tener que autenticarse. Esto sucede también con los mecanismos de autenticación que disponen los dispositivos Android para proteger el dispositivo, con este tipo de herramientas se consigue bloquear todo el teléfono, pero si el teléfono es desbloqueado, el dispositivo es vulnerable a ataques. Además, en muchas ocasiones, el usuario solo quiere proteger una aplicación de todas las que disponen, con este tipo de herramientas no es posible elegir a que aplicaciones se le otorga autenticación.

Con este proyecto se pretende construir una aplicación que permita al usuario elegir qué aplicaciones desea proteger y facilitar al usuario el proceso de autenticación continua mediante biometría. Se hace uso de los rasgos faciales de modo que si el usuario elije proteger una aplicación, ésta requerirá de autenticación cada vez que se desee utilizarla. Así, aunque el móvil este desbloqueado y el atacante que intente acceder, no le será posible pues el proceso de autenticación producirá un fallo y negará el acceso a la aplicación.

# 1.3 Objetivo

El principal objetivo perseguido con el desarrollo de este proyecto es el análisis, diseño e implementación de una aplicación para dispositivos Android, que sea capaz de otorgar de autenticación continua cada vez que un usuario accede a una aplicación. Para conseguirlo se pueden distinguirlos los siguientes sub-objetivos:

- Posibilitar autenticación biométrica continua de forma que la protección se encuentre siempre activa.
- Posibilitar al usuario la gestión de qué aplicaciones desea que tengan este tipo de autenticación y así olvidarse de recordar varias contraseñas.
- Establecer las medidas de seguridad adecuadas para que las claves de acceso a las distintas aplicaciones estén protegidas.

# 1.4 Organización del presente documento

A continuación se presenta una breve descripción del contenido de cada una de las partes que forman el documento:

- **Capítulo 1, Introducción y objetivos**: En este capítulo se realiza una introducción de los diferentes métodos de autenticación y en particular sobre la biometría facial. También se presenta el trabajo a desarrollar, se plantean los objetivos a alcanzar y se detalla las motivaciones por las que se realiza el presente proyecto.
- **Capítulo 2, Estado del Arte:** En este capítulo se realiza un análisis de las distintas aplicaciones de autenticación biométrica de rasgos faciales que existen en el mercado
- **Capítulo 3, Análisis:** En este capítulo se presenta una perspectiva general de la aplicación a desarrollar. Se detalla la arquitectura de la aplicación junto con el análisis de las distintas propuestas tecnológicas y su elección final. También se incluyen las especificaciones de los casos de uso y los requisitos de software. Por último se presentan las pruebas de aceptación y las pruebas específicas de reconocimiento facial que se realizarán para garantizar el cumplimiento de los requisitos.
- **Capítulo 4, Diseño**: En este capítulo se muestra la arquitectura definitiva de la aplicación y la infraestructura elegida para desarrollar el reconocimiento facial junto con los diagramas de clases. Se incluyen también los diagramas de secuencia para permitir comprender las distintas interacciones que tienen lugar entre los distintos componentes de la aplicación.
- **Capítulo 5, Implementación del software:** En este capítulo se detallan las decisiones tomadas durante la fase de implementación y el resultado de las pruebas de aceptación y de las pruebas específicas del reconocimiento facial.
- **Capítulo 6, Conclusiones y líneas futuras:** Último capítulo del documento en el cual se expresan las conclusiones obtenidas tras el desarrollo del proyecto y las posibles líneas futuras para extender o mejorar el funcionamiento de la aplicación.
- **Anexo, 1 Gestión de proyecto:** Se detalla la planificación, seguimiento y presupuesto del proyecto.
- **Anexo 2, Manual de Usuario**: Manual de usuario dónde se detalla cómo utilizar la aplicación y sus diferentes funcionalidades.

# CAPÍTULO 2

## 2. Estado del arte

# 2 .1 Panorámica de herramientas existentes

En esta sección se ofrece una visión general de las principales aplicaciones disponibles hoy en día que ofrecen un servicio de seguridad basado en reconocimiento facial.

En este apartado se analizarán las características más importantes dentro del abanico de las distintas soluciones encontradas en el mercado. Existen diversas aplicaciones del Sistema Operativo Android que poseen un servicio de seguridad basado en autenticación biométrica de forma continua. Se han elegido las más importantes en función de los resultados obtenidos y de los comentarios de usuarios.

Se analizarán las siguientes características en cada una de las aplicaciones:

- *Distintas aplicaciones pueden ser protegidas*: Una aplicación posee esta característica cuando se le da la oportunidad al usuario de poder elegir para que aplicaciones (entre las instaladas en el dispositivo) el servicio de seguridad este activo.
- *Compatibilidad con aplicaciones del mismo tipo:* Si puede convivir con otra aplicación del mismo estilo.
- *Es necesario autenticarse para desinstalar/detener la aplicación:* Si la aplicación obliga al usuario a identificarse para desinstalar o detener la aplicación de seguridad. Si un atacante quisiera acceder a una aplicación de las protegidas, sólo tendría que desinstalar o parar (en caso de estar en funcionamiento) la aplicación que proporciona dicha seguridad. Por ello es conveniente que este tipo de aplicaciones obliguen al usuario (en caso de estar activada la seguridad) a autenticarse al intentar desinstalarla o detener su ejecución.

### *FaceLock Pro:*

Te permite proteger el teléfono y las apps con reconocimiento facial. Te permite elegir que aplicaciones individuales se desea proteger. La versión gratuita sólo permite elegir una aplicación simultáneamente.

El reconocimiento facial se puede mejorar, otra mejora posible sería la posible elección del método de autenticación que elijas (solo está disponible el reconocimiento facial y el desbloqueo por patrón). La usabilidad (tarda demasiado en reconocer el rostro y normalmente no lo reconoce) e interfaz de la aplicación se puede mejorar bastante. Existen una versión gratuita y otra por el precio 2.49 euros. Existe versión en castellano. [7]

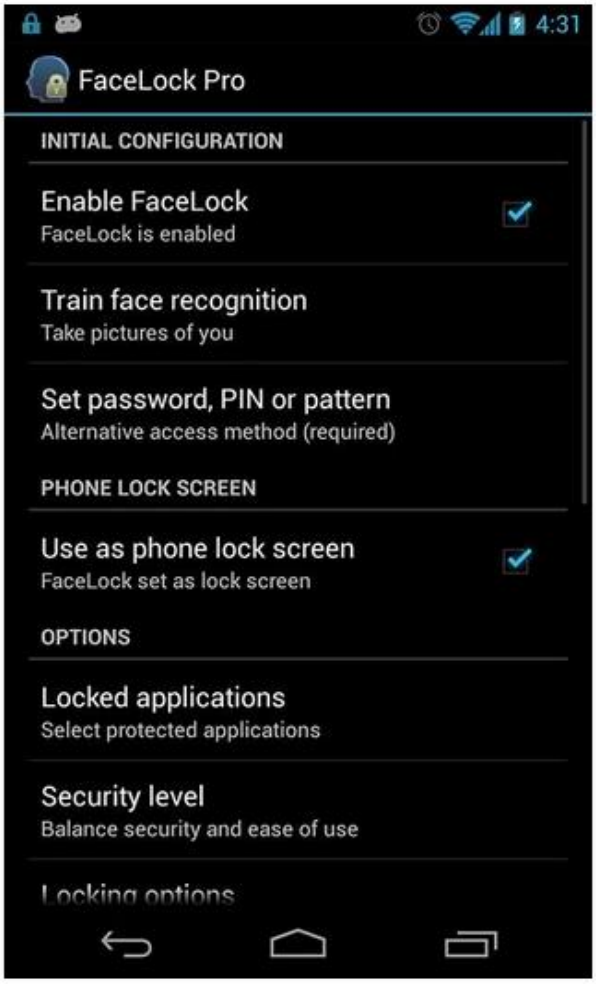



Figura 1. Pantalla inicial de Facelock Pro

### *AppLock by Face:*

No se ha conseguido probar que se pueda otorgar de autenticación facial a las aplicaciones instaladas en el dispositivo. El reconocimiento facial sólo está disponible para bloquear todo el teléfono cuando entra en estado de suspensión. [8]

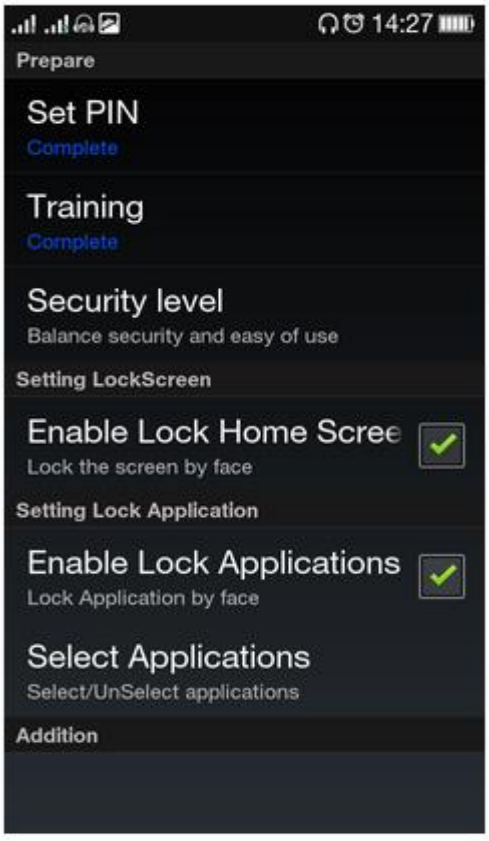



Figura 2. Pantalla de inicio de AppLock by Face

### *Perfect App Lock Pro:*

Esta aplicación te permite proteger cualquier aplicación mediante contraseña. La interfaz es bastante intuitiva. Funciona bien pero incluye funcionalidades absurdas que empeoran el rendimiento de la misma. No permite el reconocimiento facial, si el reconocimiento por huella dactilar.

Existe una funcionalidad que permite realizar una foto automáticamente en caso de que se escriba mal tres veces la contraseña. De esta forma se puede saber quién ha intentado acceder a la aplicación. [9]


Figura 3. Pantalla de inicio de Perfect App Lock Pro

### *FaceLock for Screen*

Permite el bloqueo de las aplicaciones que elijas mediante patrón de desbloqueo. Mala usabilidad, tienes que introducir demasiadas veces el patrón de bloqueo. El reconocimiento facial no suele funcionar. [10]

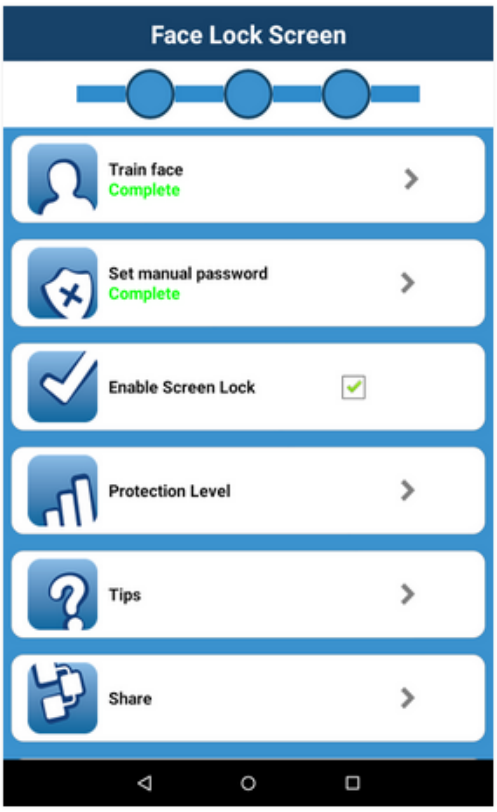



Figura 4. Pantalla de inicio FaceLock for Screen

### *Autenticación biométrica de Android.*

Con las últimas versiones del sistema operativo Android se ha introducido la posibilidad de bloquear el teléfono mediante reconocimiento facial. Este tipo de protección protege todo el dispositivo y no es posible realizar una protección a nivel de aplicaciones. Cabe destacar que esta opción solo está disponible para determinados dispositivos.

Mediante la siguiente tabla se analizan las distintas características comentadas anteriormente. Las columnas se corresponden con las aplicaciones analizadas mientras que las filas se corresponden con las propiedades a analizar. **CADroid** a diferencia de las aplicaciones actuales permite proteger distintas aplicaciones a la vez, es compatible con otras aplicaciones del mismo tipo y necesitas autenticarte para poder desinstalar la aplicación. Las aplicaciones del mercado analizadas no poseen todas estas características juntas y sólo las aplicaciones que son de pago permiten proteger varias aplicaciones a la vez.

| | FaceLock | AppLock by Face | Perfect App Lock (Sin reconocimiento facial) | FaceLock for Screen | Android |
|---|---|---|---|---|---|
| **Distintas aplicaciones pueden ser protegidas** | SI | NO | SI | NO | NO |
| **Se puede proteger a más de una aplicación a la vez** | NO | NO | SI | NO | NO |
| **Es compatible con otras aplicaciones del mismo tipo** | NO | SI | SI | NO | SI |
| **Autenticarse para desinstalar la aplicación** | SI | NO | NO | NO | NO |

Tabla 1 - Herramientas existentes

# CAPÍTULO 3

## 3. Análisis

# 3.1 Perspectiva general del sistema

En esta sección se proporciona una visión general del sistema, indicando las características a cumplir.

El sistema a desarrollar deberá ser capaz de proporcionar autenticación biométrica continua mediante rasgos faciales a cualquier aplicación de las que están instaladas en el dispositivo. Por lo tanto será capaz de identificar la cara de un usuario si previamente se ha registrado en la aplicación. El registro consta de un nombre y una contraseña. La aplicación permitirá al usuario escoger entre las aplicaciones instaladas en el dispositivo a cuál o cuáles de ellas desea otorgar seguridad.

**CADroid** se encargará de detectar que aplicaciones están en funcionamiento en todo momento. En caso de detectar que se desea acceder a una aplicación de las seleccionadas por el usuario, **CADroid** realizará una identificación de rasgos faciales a la persona que desea acceder. Si la persona que desea acceder se ha registrado en **CADroid**, el reconocimiento será satisfactorio y se permitirá el acceso a la aplicación. En caso contrario se facilitará un formulario para que el usuario pueda introducir una contraseña previamente registrada. **CADroid** no almacena ningún tipo de información privada en el dispositivo. Únicamente almacena la contraseña previamente cifrada mediante una clave generada a partir de valores obtenidos de los rasgos faciales del usuario.

**CADroid** tendrá por lo tanto las siguientes características:

1. *Registro del sistema*

En primer lugar el sistema deberá pedir al usuario de la aplicación una contraseña. El usuario podrá ingresar un nombre y una contraseña.

2. *Realizar entrenamiento*

Una vez se ha registrado, el sistema deberá obtener una serie de imágenes del usuario y calcular los valores asociados a esas imágenes. A su vez, deberá de ser capaz de calcular a partir de los datos obtenidos de las imágenes una clave que sea univoca y que servirá para cifrar la contraseña. Este proceso se conoce como entrenamiento.

3. *Elegir aplicaciones a proteger*

El usuario del sistema podrá elegir entre las aplicaciones instaladas en el teléfono a cuál de ellas desea proporcionar el servicio de seguridad. Una vez elegidas las aplicaciones que se desean proteger, se podrá activar el servicio de seguridad.

### 4. Servicio de seguridad

Se trata de un proceso que funcionará de forma independiente a la propia aplicación. Se encargará de detectar que aplicaciones del dispositivo están en funcionamiento, comparará cuál de ellas están seleccionadas previamente por el usuario, y en caso de detectar que una aplicación de las seleccionadas está en funcionamiento será el encargado de activar el reconocimiento facial.

Este servicio podrá ser activado o desactivado por el usuario siempre y cuando haya realizado el entrenamiento previamente.

### 5. Identificación de usuario

Una vez se han activado el servicio, cada vez que se quiera acceder a una aplicación de las seleccionadas, el sistema deberá de ser capaz de realizar una identificación del usuario.

**CADroid** deberá almacenar la contraseña cifrada con una clave generada a partir de los datos de las imágenes del usuario. Esta clave debe generarse de tal forma que únicamente pueda construirse a partir de las imágenes del usuario, consiguiendo una clave única asociada a un usuario. Es aquí donde reside la fortaleza del sistema.

# 3.2 Arquitectura del sistema

En esta sección se abordará la arquitectura de alto nivel del sistema. Como se ha explicado anteriormente **CADroid** deberá de ser capaz de cumplir las características definidas en el punto 3.1.

La parte fundamental del sistema es la encargada de realizar la identificación del usuario que según la Figura 5 se corresponde con el componente MIE (Módulo de identificación y entrenamiento). Es necesario que esta parte del sistema se pueda modificar sin tener que cambiar otras partes que afecten al funcionamiento general de la aplicación. Esto sucede para que cada vez que se quieran ajustar los parámetros de identificación, se puedan modificar sin tener que alterar el funcionamiento del resto de componentes la aplicación.

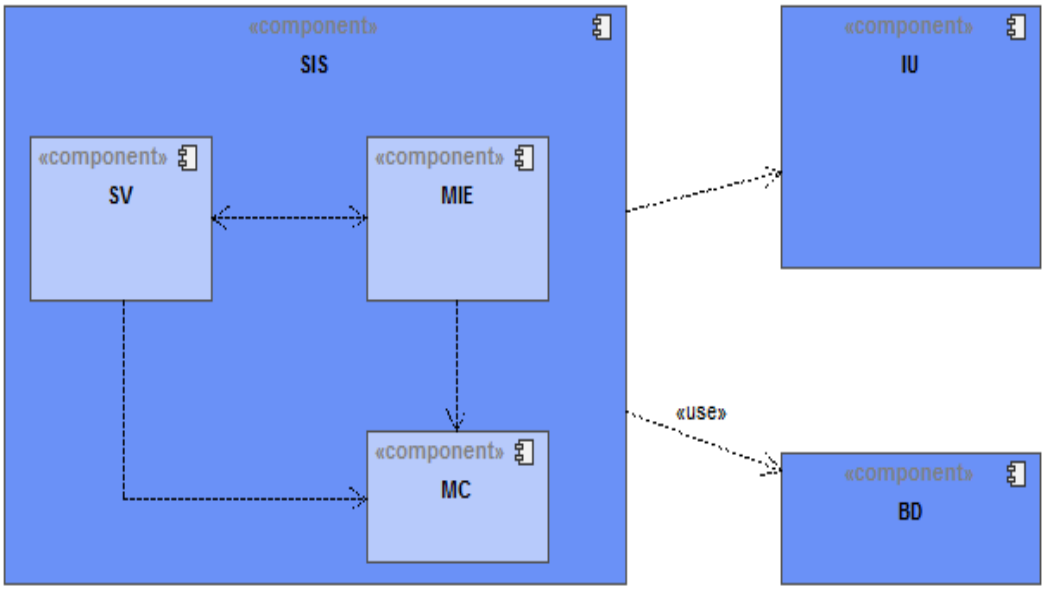



Figura 5 - Arquitectura preliminar del sistema

Por ello, se ha separado la lógica de la parte del registro del usuario y la gestión de aplicaciones, de la parte encargada de realizar el entrenamiento de las imágenes y la posterior identificación. Como se puede observar en la Figura 5, estas dos partes corresponden con el módulo de identificación y entrenamiento (MIE) y del módulo central respectivamente (MC). El modulo central se encarga de realizar el registro del usuario, realiza la gestión de las aplicaciones a proteger y se encarga de activar el servicio de seguridad (SV). Ambos módulos realizan tareas en la base de datos (BD).

Otro elemento fundamental del sistema es el servicio de seguridad (SV), se trata de un proceso que se ejecutará interrumpidamente siempre y cuando este activo. Este servicio es el encargado de comprobar que aplicaciones están siendo ejecutadas por el usuario en cada momento. También se encarga de comunicar a los demás módulos que aplicaciones se están ejecutando y si se trata de alguna de las aplicaciones de las seleccionadas a proteger.

La interacción de la aplicación con el usuario se realiza mediante las pantallas que proporcionará el componente IU.

# 3.3 Estudio tecnológico

En esta sección se detalla el estudio de las posibles tecnologías que se pueden aplicar para el desarrollo de los distintos componentes que forman la aplicación. En las siguientes subsecciones se analizarán las tecnologías impuestas junto con las tecnologías aplicables a los distintos componentes definidos en la arquitectura preliminar.

## 3.3.1 Tecnologías impuestas

La única imposición detectada para el sistema que se desarrollará en este proyecto fin de carrera es que la aplicación debe funcionar para el Sistema Operativo Android. Eso significa que la aplicación ha de desarrollarse mediantes las tecnologías que son compatibles con Android.

Para desarrollar aplicaciones en Android existen diferentes lenguajes de programación, destacan los siguientes:

- **Java**: Es el lenguaje que los creadores de Android (Empresa Google) han escogido para dar soporte a aquellas personas que deseen realizar aplicaciones de forma "nativa" en la plataforma Android. En su página web developer.android.com se puede encontrar toda la información necesaria sobre cada API de Android. La plataforma de desarrollo es eclipse y es gratuita.[11]
- **VisualBasic C# y .NET**: Creado por Microsoft, utiliza como plataforma de desarrollo el Visual Studio. Para poder integrarlo con Android es necesario descargar e instalar el SDK. No es gratuito.[12]
- **App Inventor, In Design CS6**: No requieren experiencias de programación, se programa mediante interfaces gráficas y de manera intuitiva. Para cumplir con el objetivo del proyecto estas dos tecnologías no cumplen con las exigencias mínimas.[13]

### 3.3.2 Tecnologías aplicables al componente Módulo de identificación y entrenamiento

En este apartado se muestra el estudio de las tecnologías que se tomarán en consideración dentro de las aplicables para el desarrollo del componente Módulo de identificación y entrenamiento, en él también se describirán las características principales de las distintas tecnologías consideradas.

El componente Módulo de identificación y entrenamiento tiene cómo tareas principales la detención de rostros, obtención de valores asociados a esos rostros (proceso conocido como entrenamiento) y la posterior identificación de rostros. Estas tareas se realizan mediante la ayuda de una interfaz gráfica.

Para realizar dichas funcionalidades existen varias tecnologías, entre las más importantes podemos encontrar las siguientes:

- **OpenCV**: Es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de la visión como el reconocimiento facial. Se puede integrar con Android. [14]
- **Android** (java): Existe una clase dentro del paquete android.media del API 21 de la biblioteca que proporciona google. Esta clase llamada FaceDetector.Face tiene varios métodos que permiten la obtención de datos asociados a las caras de las personas.[15]
- **Moodstocks**: Es una plataforma de reconocimiento de imágenes dedicada a los desarrolladores de aplicaciones móviles. Permite a la aplicación del desarrollador reconocer imágenes en tiempo real. Se necesita conexión a internet.[16]
- **Snapdragon SDK**: Se trata de un conjunto de herramientas de desarrollo para Android diseñado para que sea fácil integrar el procesamiento y reconocimiento facial en la aplicación a desarrollar. [17]

### 3.3.3 Tecnologías aplicables al componente Módulo Central

En este apartado se muestra el estudio de las tecnologías que se tomarán en consideración dentro de las aplicables para el desarrollo del componente *Módulo Central*.

Este módulo se encarga de lanzar las distintas interfaces gráficas, realiza operaciones en la base de datos (inserción de usuarios, comprobación de contraseña...) y es el encargado de activar el servicio de seguridad siempre y cuando lo active el usuario mediante la interfaz gráfica. Para el correcto funcionamiento de la aplicación, el Modulo Central debe comunicarse con el servicio de seguridad para coordinar el acceso a las distintas aplicaciones protegidas.

Para desarrollar este módulo se podrían utilizar cualquiera las tecnologías descritas en el punto 3.3.1.

### 3.3.4 Tecnologías aplicables al componente Servicio de Seguridad.

En este apartado se muestra el estudio de las tecnologías que se tomarán en consideración dentro de las aplicables para el desarrollo del componente Servicio de Seguridad.

El *Servicio de Seguridad* es un proceso de alta prioridad que se encarga de monitorizar las aplicaciones que están en ejecución y comprobar si alguna de ellas es una de las aplicaciones a proteger.

Para desarrollar este módulo se podrían utilizar cualquiera las tecnologías descritas en el punto 3.3.1.

### 3.3.5 Tecnologías aplicables al componente Base de datos

En este apartado se muestra el estudio de las tecnologías que se tomarán en consideración dentro de las aplicables para el desarrollo del componente *Base de datos,* en él también se describirán las características principales de las distintas tecnologías consideradas.

Este componente se encarga de la gestión de la base de datos de la aplicación, para ello es necesario elegir entre distintas soluciones que existen en el mercado. A continuación se detallan los distintos gestores de base de datos más importantes:

- **CouchDB**: Gestor de base de datos de código abierto, desarrollado por Damien Katz [25], cuya finalidad principal es prestar servicio a aplicaciones web (licencia Apache License y liberada la versión 1.2 en Abril de 2012). Se trata de una base de datos NoSQL, es decir, no usan el lenguaje de consultas estructurado o SQL como principal lenguaje de consultas. Es un gestor de base de datos externa por lo que se requiere conexión directa.[18]
- **FireBird**: Sistema de administración de base de datos relacional de código abierto, que utiliza el lenguaje de consultas SQL, liberado por Borland en el año 2000 y escrito en C y C++. Es un gestor de base de datos externa por lo que se requiere conexión directa.[19]
- **Mysql**: Sistema de gestión de base de datos relacional multiusuario y multihilo. Desarrollada inicialmente por Sun Microsystems, actualmente su propietario es Oracle Corporation. Software libre y tipo de licencia GNU GPL para aquellos proyectos que se adapten a este tipo de licencia (permite usar, compartir y modificar el software), con la opción de poder adquirir una de carácter más privativo previo pago (parte del código está bajo copyright). Gestor de base de datos externa por lo que se requiere conexión directa.[20]
- **SQLite**: Es un ligero motor de bases de datos de código abierto, que se caracteriza por mantener el almacenamiento de información persistente de forma sencilla. No requiere el soporte de un servidor por lo que no necesita conexión directa.[21]

### 3.3.6 Tecnologías aplicables al componente Interfaz gráfica

En este apartado se muestra el estudio de las tecnologías que se tomarán en consideración dentro de las aplicables para el desarrollo del componente *Interfaz gráfica.*

Este componente está compuesto por las distintas interfaces gráficas que permitirán la interacción del usuario con el sistema. Este componente debe soportar el manejo de la cámara del dispositivo con el objetivo de proporcionar cada una de las imágenes al módulo de identificación y entrenamiento.

Para desarrollar las distintas pantallas existen las librerías de **openCV** para el manejo de la cámara y el API que proporciona **Android** para la creación de interfaces gráficas.

# 3.4 Selección de tecnologías no impuestas

En esta sección se mostrarán las tecnologías no impuestas seleccionadas para el desarrollo del sistema junto con el razonamiento que ha llevado a esas decisiones.

Se decide por trabajar con **java** dado que la plataforma de desarrollo (eclipse) es gratuita, es el lenguaje que Google (la empresa creadora de Android) ha escogido para dar soporte a los desarrolladores y porque permite la integración de distintas librerías externas. Esta será la **tecnología principal** con la que se desarrollará la aplicación, pero existen otras tecnologías que se utilizarán para realizar determinadas tareas específicas en cada módulo.

### *Módulo de Identificación y entrenamiento*

**Se descartan las soluciones de Moodstocks y Snapdragon**, la primera de ellas requiere de conexión a internet y dado que las imágenes se procesan en la nube no se tiene el control del funcionamiento del reconocimiento facial. Snapdragon contiene herramientas muy útiles para desarrollar el módulo de identificación y entrenamiento pero al ser un API fabricado por Qualcomm sólo funciona para los dispositivos que tengan procesadores de la serie Snapdragon S4 8960. Este hecho supone que la aplicación únicamente funcione para un conjunto reducido de dispositivos por lo tanto también se descarta esta solución.

**Se eligen las soluciones que ofrecen OpenCV y el API de Android**. OpenCV al ser una librería de código abierto se dispone de mucha información y además disponen de una cantidad de herramientas muy potentes que facilitan la identificación y el reconocimiento de rostros faciales. El API de Android también posee herramientas para la identificación y reconocimiento de rostros pero es menos eficiente y robusta que la solución de OpenCV. Aun así contiene funciones que aportan información muy relevante sobre los rasgos faciales. Por lo tanto se utilizará una solución conjunta entre las herramientas que ofrecen OpenCV y el API de Android.

### *Módulo central*

Para la el lanzamiento de las distintas interfaces se utilizará el API que proporciona Android, en concreto se utilizaran las clases de los paquetes *android.app, android.content, android.os y android.widget.*

Para la comunicación con el servicio de seguridad, el API de Android proporciona una serie de clases dentro de los paquetes *android.os y android.content.* Las clases *Handler*, *ServiceConnected, Message* y *Messenger* serán las encargadas de realizar esta comunicación.

Por ultimo para realizar las gestiones en la base de datos se utilizará el paquete *android.database* del API de Android.

### *Servicio de seguridad*

Para este módulo al igual que el módulo central no es necesario utilizar una tecnología complementaria a la que proporciona el API de Android. Por lo tanto para la realización de las tareas definidas anteriormente se utilizarán las clases de los paquetes *android.os, android.content y android.app.*

### *Base de datos*

**Se elige la opción de SQLite.** Dado que no requiere de soporte de un servidor, no ejecuta un proceso para administrar la información si no que implementa un conjunto de librerías encargadas de la gestión. Tampoco necesita configuración, liberando al programador de todo tipo de configuraciones. Además **SQLite** crea un archivo para el esquema completo de una base de datos, de esta manera los datos de las aplicaciones no pueden ser accedidos por contextos externos. Por todas estar razones y además por ser de código abierto se ha elegido la opción de **SQLite.**

### *Interfaz gráfica*

Para el desarrollo de este módulo se ha elegido para el manejo de la cámara la librería de **openCV** y para la creación de las distintas pantallas el API que proporciona **Android**.

# 3.5 Arquitectura definitiva de alto nivel

## 3.5.1 Arquitectura definitiva del sistema

Partiendo del diseño inicial presentado en la sección 3.2 y teniendo en consideración las tecnologías impuestas y las decisiones tomadas respecto a las tecnologías a utilizar se ha modificado el diagrama de componentes de la Figura 6 para reflejar la arquitectura definitiva del sistema.

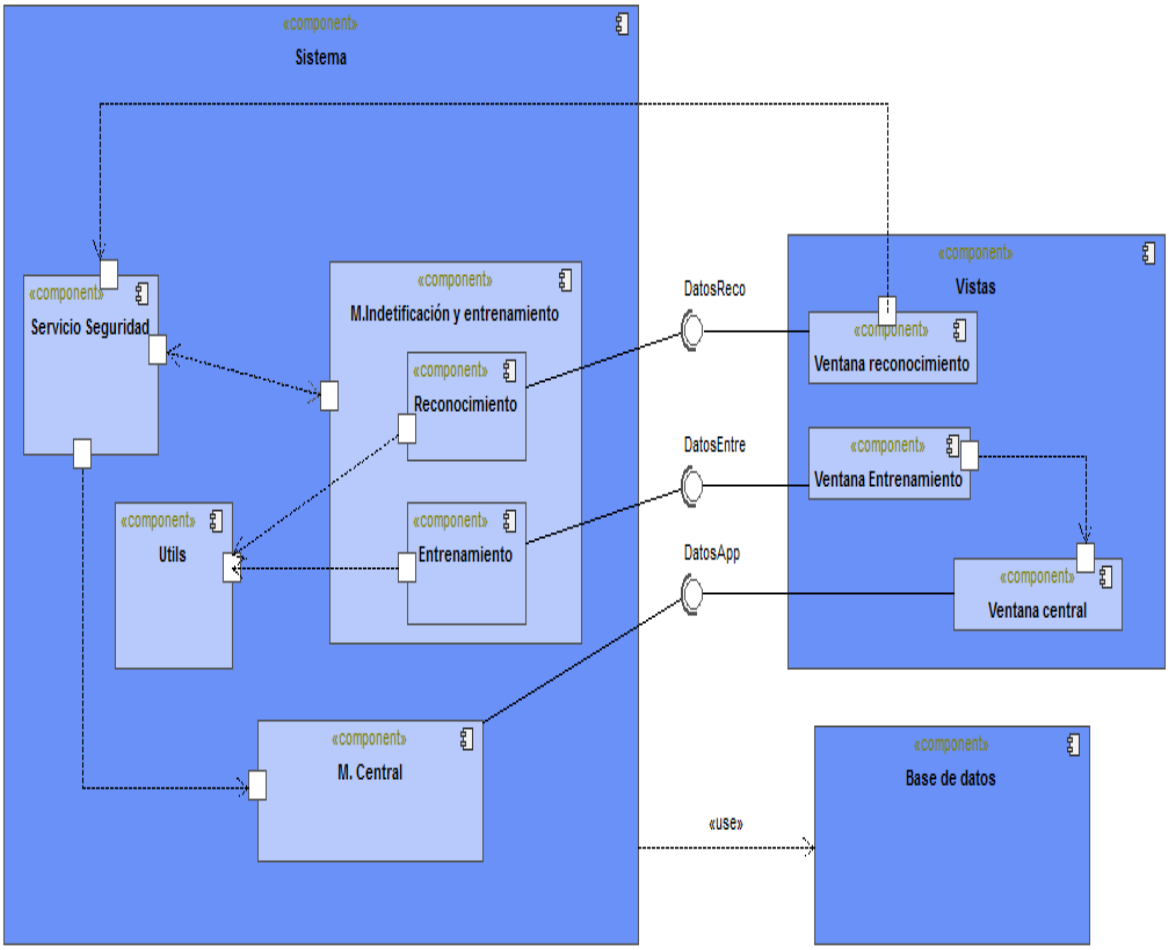



Figura 6. Arquitectura definitiva del sistema

La Figura 6 muestra el diagrama con la arquitectura definitiva.

Como se puede observar se han creado nuevos componentes dentro de los ya existentes y han surgido nuevas relaciones. El *Módulo de identificación y entrenamiento* lo forman tres componentes, el componente encargado de realizar el reconocimiento denominado *Reconocimiento*, el componente encargado de realizar el entrenamiento llamado *Entrenamiento* y por último el componente *Utils,* encargado

de proporcionar las clases y métodos necesarios para la correcta realización del entrenamiento y la identificación. Ambos componentes (*reconocimiento* y *entrenamiento*) dependen de las funciones y métodos que proporciona el componente *Utilis.*

Estos componentes reciben los datos necesarios para realizar sus tareas a partir de las interfaces creadas por los componentes *Ventana reconocimiento* y *Ventana Entrenamiento.* El módulo central obtiene los datos del usuario a partir de la ventana central, muestra las app instaladas al usuario mediante la ventana de *selectApp* y almacena la selección elegida de aplicaciones a proteger.

Para realizar el entrenamiento, el componente *entrenamiento* necesita de los datos proporcionados por la ventana *entrenamiento,* a partir de estos datos realiza operaciones complejas y almacenará los datos necesarios en la base de datos. El servicio de seguridad sólo puede ser activado desde el módulo central y se encarga de monitorizar las aplicaciones que están en ejecución, en caso de detectar que se está ejecutando una aplicación de las seleccionadas por el usuario, el *servicio de seguridad* activa la *ventana reconocimiento* para que el *módulo de reconocimiento* realice la identificación a partir de los datos recogidos. Posteriormente el *módulo de reconocimiento y entrenamiento* debe notificar los resultados de la identificación al servicio de seguridad.

# 3.6 Casos de uso

En esta sección se muestra el diagrama de casos de uso de la aplicación junto con la definición textual de cada uno de los casos de uso presentes en el diagrama. El estudio de los casos de uso descritos facilitará la extracción de requisitos en fases posteriores del proyecto.

## 3.6.1 Diagrama de casos de uso

La Figura 7 muestra el diagrama que representa los distintos casos de uso identificados para el proyecto.

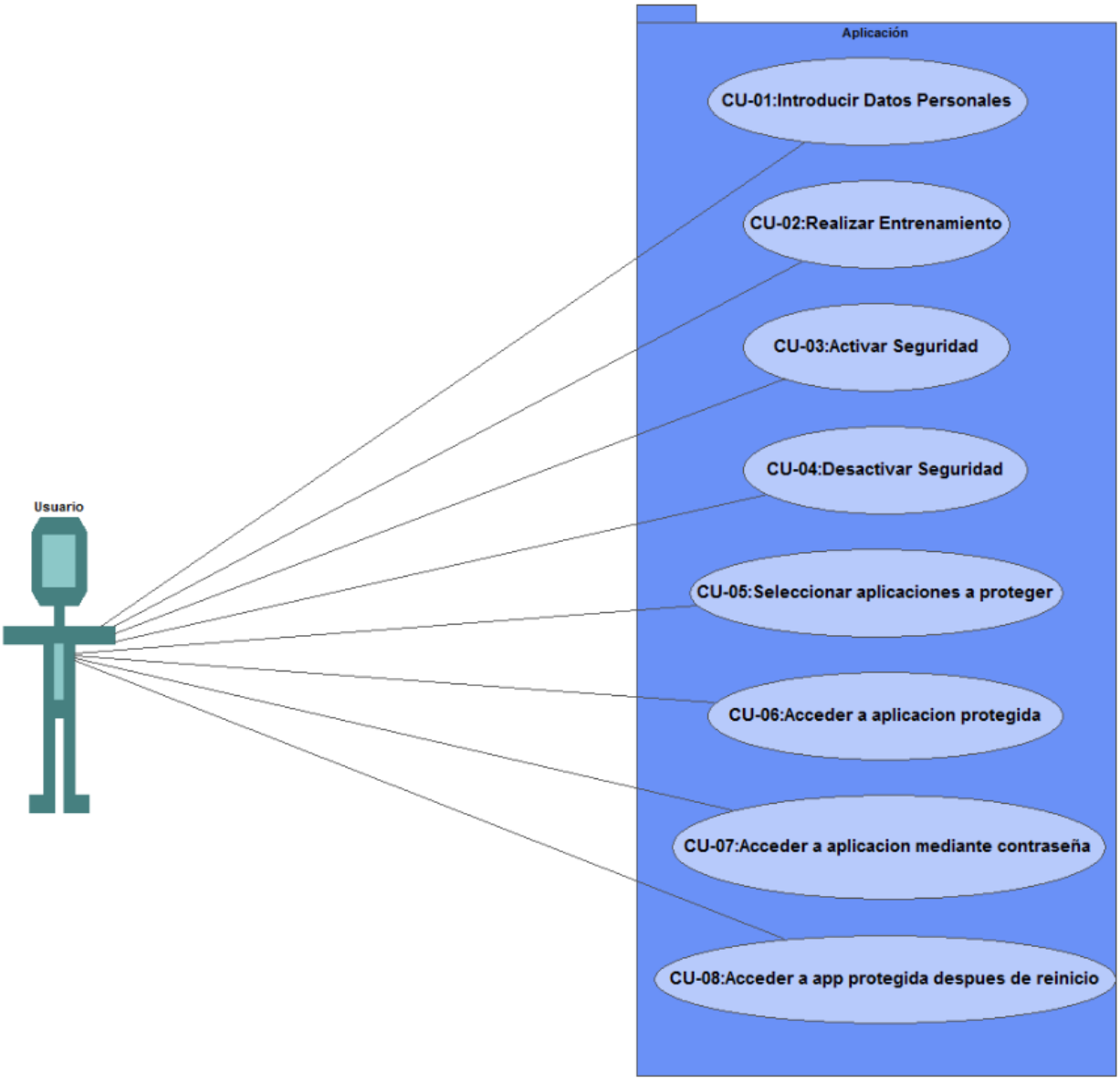



Figura 7. Diagrama de Casos de uso

Se puede observar las distintas funcionalidades que la aplicación ofrece. Cabe destacar que para el tipo de aplicación a desarrollar sólo se ha contemplado la existencia de un usuario registrado de la aplicación. No se contempla la existencia del usuario no registrado.

### 3.6.2 Definición textual de los casos de uso

En este apartado se muestra información detallada de los distintos casos de uso identificados en el diagrama de casos de uso del apartado anterior.

| **Identificador :** CU-01 | |
|---|---|
| **Nombre:** | Registro del sistema |
| **Autor:** | Alfredo Ruiz Hernández |
| **Descripción:** | Introducir nombre y contraseña en el formulario de la aplicación. |
| **Precondiciones:** | • La aplicación esta arrancada.<br>• Se ha introducido un nombre y contraseña.<br>• Se ha pulsado el botón de aceptar |
| **Postcondiciones:** | • Se han almacenado el nombre y la contraseña y el usuario se ha registrado en la aplicación. |
| **Flujo Normal:** | 1. Abrir la aplicación.<br>2. Pulsar botón entrenar.<br>3. Introducir contraseña y nombre |
| **Flujo Alternativo:** | 1. Abrir la aplicación<br>2. Pulsar botón entrenar.<br>3. Introducir contraseña y nombre |

Tabla 2. Caso de uso CU-01

| **Identificador :** CU-02 | |
|---|---|
| **Nombre:** | Realizar entrenamiento. |
| **Autor:** | Alfredo Ruiz Hernández. |
| **Descripción:** | Realizar el entrenamiento de las imágenes del usuario. |
| **Precondiciones:** | • La Aplicación esta arrancada.<br>• Se ha pulsado el botón de entrenamiento.<br>• Se han introducido la contraseña y el nombre.<br>• Se ha pulsado el botón de aceptar.<br>• Se ha pulsado el botón de entrenar durante quince veces con la cámara posterior en posición hacia la cara del usuario. |
| **Postcondiciones:** | • Se ha realizado el entrenamiento. Se han obtenido los datos necesarios del usuario para la posterior identificación. |
| **Flujo Normal:** | 1. Abrir la aplicación.<br>2. Pulsar botón entrenar.<br>3. Introducir contraseña y nombre.<br>4. Pulsar botón aceptar.<br>5. Realizar quince fotografías de sí mismo.<br>6. Pulsar el botón de Entrenar. |
| **Flujo Alternativo:** | 1. Abrir la aplicación.<br>2. Pulsar botón entrenar.<br>3. Introducir contraseña y nombre.<br>4. Pulsar botón aceptar.<br>5. Realizar quince fotografías de sí mismo.<br>6. Pulsar el botón de Entrenar. |

Tabla 3.Caso de uso CU-02

| **Identificador :** CU-03 | |
|---|---|
| **Nombre:** | Activar Servicio de seguridad. |
| **Autor:** | Alfredo Ruiz Hernández. |
| **Descripción:** | Se activa el servicio de seguridad si previamente se ha realizado el entrenamiento. |
| **Precondiciones:** | • La aplicación esta arrancada<br>• Se ha realizado el entrenamiento previamente.<br>• Se ha activado la casilla de activación del servicio de seguridad. |
| **Postcondiciones:** | • El servicio de seguridad está activado. |
| **Flujo Normal:** | 1. Abrir la aplicación.<br>2. Activar la casilla de activación del servicio de seguridad. |
| **Flujo Alternativo:** | 1. Abrir la aplicación.<br>2. Realizar entrenamiento<br>2. Activar la casilla de activación del servicio de seguridad. |

Tabla 4. Caso de uso CU-03

| **Identificador :** CU-04 | |
|---|---|
| **Nombre:** | Desactivar servicio de seguridad |
| **Autor:** | Alfredo Ruiz Hernández. |
| **Descripción:** | Se desactiva el servicio de seguridad si previamente se ha activado |
| **Precondiciones:** | • El servicio de seguridad está activado<br>• Se ha desactivado la casilla de activación del servicio de seguridad |
| **Postcondiciones:** | • El servicio de seguridad esta desactivado. |
| **Flujo Normal:** | 1. Abrir la aplicación<br>2. Desactivar la casilla de activación del servicio de seguridad |
| **Flujo Alternativo:** | 1. Abrir la aplicación.<br>2. Realizar entrenamiento<br>2. Desactivar la casilla de activación del servicio de seguridad. |

Tabla 5. Caso de Uso CU-04

| **Identificador :** CU-05 | |
|---|---|
| **Nombre:** | Seleccionar app a proteger |
| **Autor:** | Alfredo Ruiz Hernández. |
| **Descripción:** | Seleccionar un conjunto de app de las instaladas en el teléfono con el objetivo de otorgarles de autenticación biométrica continua. |
| **Precondiciones:** | • Aplicación arrancada<br>• Se ha pulsado el botón de seleccionar apps.<br>• Se han seleccionado una o varias apps |
| **Postcondiciones:** | • Si el servicio de seguridad está activo. Se activa el reconocimiento facial cada vez que se intenta acceder a cada una de las apps seleccionadas |
| **Flujo Normal:** | 1. Abrir la aplicación.<br>2. Pulsar botón de selección de apps. |

| | |
|---|---|
| | 3. Seleccionar una o varias app de la lista |
| **Flujo Alternativo:** | 1. Abrir la aplicación.<br>2. Pulsar botón de selección de apps.<br>3. Seleccionar una o varias app de la lista. |

Tabla 6. Caso de Uso CU-05

| **Identificador :** CU-06 | |
|---|---|
| **Nombre:** | Acceder a app protegida |
| **Autor:** | Alfredo Ruiz Hernández. |
| **Descripción:** | Acceder a una aplicación de las seleccionadas con el servicio de seguridad activado. |
| **Precondiciones:** | • Aplicación arrancada.<br>• Se ha activado el servicio de seguridad.<br>• Se ha seleccionado una aplicación de la lista.<br>• Se ha intentado arrancar la aplicación seleccionada anteriormente. |
| **Postcondiciones:** | • Se realiza la identificación del usuario y se accede a la aplicación seleccionada. |
| **Flujo Normal:** | 1. Abrir la aplicación.<br>2. Seleccionar una app de la lista.<br>3. Activar servicio de seguridad.<br>4. Abrir app seleccionada anteriormente.<br>5. Mantener el rostro fijo delante del dispositivo. |
| **Flujo Alternativo:** | 1. Abrir la aplicación.<br>2.Activar servicio de Seguridad<br>3. Seleccionar una app de la lista.<br>4. Abrir app seleccionada anteriormente.<br>5. Mantener el rostro fijo delante del dispositivo. |

Tabla 7.Caso de uso CU-06

| **Identificador :** CU-07 | |
|---|---|
| **Nombre:** | Acceder a app protegida |
| **Autor:** | Alfredo Ruiz Hernández. |
| **Descripción:** | Acceder a una aplicación de las seleccionadas con el servicio de seguridad activado. |
| **Precondiciones:** | • Aplicación arrancada.<br>• Se ha activado el servicio de seguridad.<br>• Se ha seleccionado una aplicación de la lista.<br>• Se ha intentado arrancar la aplicación seleccionada anteriormente. |
| **Postcondiciones:** | • Se realiza la identificación del usuario y se accede a la aplicación seleccionada. |
| **Flujo Normal:** | 1. Abrir la aplicación.<br>2. Seleccionar una app de la lista.<br>3. Activar servicio de seguridad.<br>4. Abrir app seleccionada anteriormente.<br>5. Mantener el rostro fijo delante del dispositivo. |
| **Flujo Alternativo:** | 1. Abrir la aplicación.<br>2.Activar servicio de Seguridad<br>3. Seleccionar una app de la lista.<br>4. Abrir app seleccionada anteriormente.<br>5. Mantener el rostro fijo delante del dispositivo. |

Tabla 8.Caso de uso CU-07

| **Identificador :** CU-08 | |
|---|---|
| **Nombre:** | Acceder a app mediante contraseña |
| **Autor:** | Alfredo Ruiz Hernández. |
| **Descripción:** | Acceder a una aplicación de las seleccionadas con el servicio de seguridad activado introduciendo la contraseña, este caso se da cuando se obtiene un resultado negativo durante el reconocimiento. |
| **Precondiciones:** | • Aplicación arrancada.<br>• Se ha activado el servicio de seguridad. |

| | |
|---|---|
| | • Se ha seleccionado una aplicación de la lista.<br>• Se ha intentado arrancar la aplicación seleccionada anteriormente. |
| **Postcondiciones:** | • Se realiza la identificación del usuario y se accede a la aplicación seleccionada. |
| **Flujo Normal:** | 1. Abrir la aplicación.<br>2. Seleccionar una app de la lista.<br>3. Activar servicio de seguridad.<br>4. Abrir app seleccionada anteriormente.<br>5. Mantener el rostro fijo delante del dispositivo. |
| **Flujo Alternativo:** | 1. Abrir la aplicación.<br>2.Activar servicio de Seguridad<br>3. Seleccionar una app de la lista.<br>4. Abrir app seleccionada anteriormente.<br>5. Mantener el rostro fijo delante del dispositivo. |

Tabla 9.Caso de Uso CU-08

# 3.7 Requisitos de software

En esta sección se presentan los requisitos de software identificados que deberán ser cubiertos para cumplir con los objetivos especificados para este proyecto.

Se han de tener en cuenta todas las características que el producto debe proporcionar, enunciadas en el punto 3.1 Perspectiva general del sistema.

El conjunto de requisitos se ha divido dependiendo de su condición. Se establecen dos tipos de requisitos:

- **Requisitos Funcionales**: Específica algo que el sistema entregado debe ser capaz de realizar. Simboliza lo que tiene que realizar el usuario para conseguir un objetivo dentro de la aplicación.
- **Requisitos no Funcionales**: Se corresponden con las restricciones que afectan al sistema. Representan las limitaciones impuestas al usuario.

## 3.7.1 Requisitos Funcionales

| Requisitos del Software | | | | |
|---|---|---|---|---|
| **Tipo: Funcional** | | | | |
| **Id** | **Nombre** | **Descripción** | **Estabilidad** | **Prioridad** |
| **RF-01** | Introducir datos | El usuario podrá registrarse en la aplicación introduciendo su nombre y la contraseña | Alta | Media |
| **RF-02** | Monitorización de apps | La aplicación será capaz de detectar que aplicaciones están funcionando en el dispositivo en todo momento. | Alta | Alta |
| **RF-03** | Autenticación Continua | La aplicación activará el reconocimiento facial cada vez que el usuario quiera acceder a una aplicación de las seleccionadas previamente. | Alta | Alta |
| **RF-04** | Realizar entrenamiento | Durante la fase de entrenamiento la aplicación será capaz obtener una serie de valores a partir de las imágenes de la cara del usuario. Almacenar dichos valores para posteriormente | | |

| | | | | |
|---|---|---|---|---|
| | | compáralos en el proceso de reconocimiento. | | |
| **RF-05** | Detectar entrenamiento realizado | La aplicación detectará si el usuario ha realizado el entrenamiento. | Alta | Media |
| **RF-06** | Generar clave | El sistema será capaz de generar una clave única a partir de los valores asociados a los distintos rasgos faciales del usuario. | Alta | Alta |
| **RF-07** | Reconocer usuario | El sistema será capaz de reconocer la cara del usuario si previamente ha realizado el entrenamiento. | Alta | Alta |
| **RF-08** | Permitir acceso | En caso de reconocer la cara del usuario registrado, el sistema será capaz de permitir el acceso a la aplicación que el usuario quería acceder. | Alta | Alta |
| **RF-09** | Acceso mediante contraseña | En caso de fallo de autenticación, el usuario podrá acceder a la aplicación introduciendo la contraseña. | Alta | Alta |
| **RF-10** | Elegir imágenes | Durante la fase de entrenamiento, el usuario podrá elegir las imágenes que desea realizarse a sí mismo. La aplicación no realiza las imágenes de forma automática. | Alta | Baja |
| **RF-11** | Reinicio del dispositivo | Si el servicio de seguridad está activado, el sistema será capaz de mantenerlo activo en caso de reinicio del dispositivo | Alta | Media |
| **RF-12** | Seleccionar aplicaciones | Se podrá activar el servicio de seguridad para varias aplicaciones a la vez | Alta | Alta |

Tabla 10 - Requisitos Funcionales

### 3.7.2 Requisitos no Funcionales

| Requisitos del Software | | | | |
|---|---|---|---|---|
| **Tipo: No funcional** | | | | |
| **Subtipo: Interfaz** | | | | |
| **Id** | **Nombre** | **Descripción** | **Estabilidad** | **Prioridad** |
| **RNF-01** | Mostar app instaladas | En la sección de selección de aplicaciones a proteger, las aplicaciones instaladas se mostrarán mediante una lista. La lista está compuesta del nombre de la aplicación y la imagen de la misma. | Media | Alta |
| **RNF-02** | Indicador de realización de entrenamiento | Durante el proceso de entrenamiento se indicará al usuario que se están realizando las operaciones oportunas. | Media | Baja |
| **RNF-03** | Mostrar Formulario | La pantalla de registro del sistema será un formulario para introducir el nombre y dos veces la contraseña. | Alta | Alta |
| **RNF-04** | Menú principal | El menú principal de la aplicación está compuesto por una casilla de activación para activar/desactivar el servicio de seguridad y dos botones, uno para acceder a la fase de entrenamiento y otro para acceder a la sección de selección de aplicaciones a proteger. | | |
| **Subtipo: Operacional** | | | | |
| **Id** | **Nombre** | **Descripción** | **Estabilidad** | **Prioridad** |
| **RNF-05** | Habilitar seguridad | No se podrá activar el servicio de seguridad si no ha realizado el entrenamiento previamente. | Alta | Alta |
| **RNF-06** | Requisitos para realizar entrenamiento | Para realizar el entrenamiento, se deberá introducir su nombre y contraseña | Alta | Alta |
| **RNF-07** | Introducir contraseña | Se deberá introducir la contraseña dos veces. | Alta | Baja |
| **RNF-08** | Mínimo caracteres | La contraseña deberá tener un mínimo de 8 caracteres | Alta | Baja |
| **RNF-09** | Contraseña alfanumérica | La contraseña deberá ser alfanumérica. | Alta | Alta |
| **RNF-10** | Misma contraseña | Las dos contraseñas introducidas deben de ser iguales. | Alta | Media |
| **RNF-11** | Nombre no numérico | El nombre introducido no deberá contener dígitos. | Alta | Alta |

| | | | | |
|---|---|---|---|---|
| **RNF-12** | Mínimas imágenes | El usuario deberá hacerse un mínimo de 15 imágenes. | Media | Baja |
| **RNF-13** | Tratamiento de imágenes | No se almacenará en el dispositivo ninguna imagen. | Alta | Alta |
| **RNF-14** | Almacenamiento de datos asociados al usuario | Se almacenará en un XML toda la información relacionada con la cara del usuario. | Alta | Alta |
| **RNF-15** | Cifrado de contraseña | La contraseña del usuario se almacenará cifrada con la clave generada. | Alta | Alta |
| **RNF-16** | Generación clave | La clave generada asociada a la cara del usuario no se almacenará en el dispositivo | Alta | Alta |
| **RNF-17** | Acceso a ajustes | Se requiere de autenticación para acceder a ajustes si el servicio de seguridad está activo. | Alta | Alta |
| **RNF-18** | Acceso al sistema | Se requiere de autenticación para acceder al sistema si el servicio de seguridad está activo. | Alta | Alta |

Tabla 11 - Requisitos no funcionales

# 3.8 Diseño del plan de pruebas

Una vez se ha definido las características de la aplicación que se desarrollará a lo largo de este proyecto fin de carrera se puede proceder a realizar el plan de pruebas de aceptación de la misma. En esta sección se detalla el plan de pruebas de aceptación de la aplicación en donde se muestran las distintas pruebas que se han de superar y los resultados esperados de las mismas.

## 3.8.1 Plan de pruebas de aceptación

| Pruebas de aceptación | | | | |
|---|---|---|---|---|
| **id** | **Elementos probados** | **Descripción** | **Entrada** | **Salida** |
| **PA-01** | RNF-05 | Comprobar que el servicio de seguridad no se puede activar antes de realizar el entrenamiento. | Pulsar el icono del sistema. | No se puede activar el servicio de seguridad |
| **PA-02** | RNF-05, RF-05 | Comprobar que el sistema detecta que el entrenamiento se ha realizado. | Pulsar el icono del sistema.<br>Pulsar el botón del entrenamiento.<br>Realizar el entrenamiento. | Se puede activar la casilla de activación del servicio de seguridad. |
| **PA-03** | RNF-04, RNF-03, RF-09. | Comprobar que al pulsar el botón de entrenamiento aparece el formulario de introducción de nombre y contraseña. | Pulsar el botón de entrenamiento | Nueva pantalla con tres campos y un botón. |
| **PA-04** | RNF-08 | Comprobar que la contraseña introducida tiene que tener un mínimo de 8 caracteres. | Introducir contraseña con menos de 8 caracteres: Introducir *Abc.* | Se muestra un mensaje de error. |
| **PA-05** | RNF-07 | Comprobar que la contraseña hay que introducirla 2 veces. | Introducir una única contraseña. | Se muestra un mensaje de error. |
| **PA-06** | RNF-10 | Comprobar que las contraseñas introducidas tienen que ser iguales. | Introducir en el primer campo de contraseñas: *Abcdefghi*.<br>Introducir en el segundo campo de contraseñas: *aaaaaaaaa.* | Se muestra un mensaje de error. |

| | | | | |
|---|---|---|---|---|
| **PA-07** | RNF-11 | Comprobar que el nombre no puede contener dígitos. | Introducir un nombre que contenga dígitos: *123* | Se muestra un mensaje de error. |
| **PA-08** | RNF-09 | Comprobar que la contraseña debe ser alfanumérica. | Introducir en el campo contraseña: *Abcderrr.* | Se muestra un mensaje de error. |
| **PA-09** | RNF-07,08,09,10,11 | Comprobar que al introducir una contraseña alfanumérica con más de 8 caracteres dos veces y un nombre sin dígitos se accede al entrenamiento. | Introducir en los campos:<br>→Nombre: *Pepito*.<br>→Contraseña 1:*Abcdefjhi12*<br>→Contraseña 2:*Abcdefjhi12.* | Aparece la pantalla de entrenamiento |
| **PA-10** | RNF-12 | Comprobar que el máximo y el mínimo número de imágenes que se debe realizar el usuario son 15. | Realizar más de 15 fotos. | El sistema no lo permite. |
| **PA-11** | RNF-13 | Comprobar que no se almacenan ninguna imagen en el dispositivo. | Mirar en los archivos del teléfono si se almacena algún tipo de imagen. | No se almacena ningún tipo de imágenes. |
| **PA-12** | RNF-01 | Comprobar que en la selección de aplicaciones a proteger aparecen todas las apps instaladas. | Abrir la el apartado de selección de apps y comparar las aplicaciones que aparecen con las aplicaciones que aparecen en la sección de ajustes. | Aparecen todas las app instaladas. |
| **PA-13** | RF-02, RF-03 | Comprobar que la aplicación seleccionada tiene activado el reconocimiento facial si el servicio de seguridad activado. | Elegir la aplicación en la sección de selección de aplicaciones.<br>Activar servicio de seguridad.<br>Salir del sistema.<br>Abrir aplicación seleccionada anteriormente | Se activa el reconocimiento facial. |
| **PA-14** | RF-02, RF-03, RF-12 | Comprobar que se puede activar<br>el servicio de seguridad para varias aplicaciones a la vez | Elegir varias aplicaciones en la sección de selección de | Se activa el reconocimiento facial. |

| | | | | |
|---|---|---|---|---|
| | | | aplicaciones.<br>Activar servicio de seguridad.<br>Salir del sistema.<br>Abrir una aplicación de las seleccionadas. | |
| **PA-15** | RF-08, RF-07 | Comprobar que el sistema realiza el reconocimiento facial de forma satisfactoria al usuario que ha realizado el entrenamiento. | Realizar entrenamiento.<br>Activar servicio de seguridad.<br>Seleccionar *galería* como aplicación a proteger.<br>Abrir la *galería de imágenes.*<br>Esperar mirando hacia la cámara del dispositivo a que se realice el reconocimiento. | Se accede a la galería de ajustes. |
| **PA-16** | RF-08, RF-07, RF-01 | Comprobar que el sistema realiza el reconocimiento facial y no reconoce a un usuario que no ha realizado el entrenamiento. | Realizar entrenamiento.<br>Activar servicio de seguridad.<br>Seleccionar *galería* como aplicación a proteger.<br>Abrir la *galería de imágenes.*<br>Enfocar la cámara delantera del dispositivo a otra persona diferente. | No se accede a la galería de ajustes.<br>Aparece una pantalla de introducción de contraseña. |
| **PA-17** | RF-08, RF-07, RF-01 | Comprobar que al introducir correctamente la contraseña (en caso de fallo de autenticación) se tiene acceso a la aplicación. | Realizar entrenamiento e introducir la siguiente contraseña: *Abc123456*.<br>Activar servicio de seguridad y seleccionar la galería de imágenes como aplicación a proteger. | Se accede a galería de ajustes. |

| | | | | |
|---|---|---|---|---|
| | | | Abrir galería de imágenes y enfocar la cámara delantera del dispositivo a otra persona diferente. En caso de fallo de autenticación introducir la siguiente contraseña: *Abc123456.* | |
| **PA-18** | RF-08, RF-07, RF-01 | Comprobar que al introducir mal la contraseña (en caso de fallo de autenticación) no tiene acceso a la aplicación. | Realizar entrenamiento e introducir la siguiente contraseña: *Abc123456*. Activar servicio de seguridad y seleccionar la galería de imágenes como aplicación a proteger. Abrir galería de imágenes y enfocar la cámara delantera del dispositivo a otra persona diferente. En caso de fallo de autenticación introducir la siguiente contraseña: *Abc111111.* | Se muestra un mensaje de contraseña mal introducida. |
| **PA-19** | RNF-17 | Comprobar que con el servicio de seguridad habilitado se requiere de autenticación al acceder ajustes. | Activar el servicio de seguridad y acceder a ajustes. | Se activa el reconocimiento facial |
| **PA-20** | RNF-18 | Comprobar que con el servicio de seguridad habilitado se requiere autenticación para acceder al sistema. | Activar el servicio de seguridad. Salir del sistema. Abrir el sistema | Se activa el reconocimiento facial. |

| | | | | |
|---|---|---|---|---|
| **PA-21** | RF-11 | Comprobar que si el servicio de seguridad está habilitado, lo sigue estando después del reinicio del dispositivo. | Activar servicio de seguridad.<br>Reiniciar dispositivo.<br>Acceder al sistema. | Se activa el reconocimiento facial. |

Tabla 12.Pruebas de Aceptación

## 3.8.2 Plan de pruebas para el reconocimiento facial

En esta sección se detallará el diseño del plan de pruebas específico para el reconocimiento facial. Con el objetivo de comprobar el funcionamiento y la calidad del reconocimiento facial de la aplicación.

Mediante la tabla 13 se muestra los distintos factores externos en los cuales se han realizado las pruebas de reconocimiento facial. La columna iluminación se refiere a la cantidad de luz que hay en el momento que se realiza el reconocimiento facial, la iluminación puede ser alta o baja. La columna de luz artificial se refiere a si la luz que hay cuando se realiza la prueba es luz artificial o luz natural. La columna distancia al objetivo marca la distancia que hay desde la cara del usuario que realiza la prueba hasta el objetivo de la cámara del dispositivo móvil. La distancia al objetivo puede ser alta o baja.

| Iluminación | Luz Artificial | Distancia al objetivo |
|---|---|---|
| Baja | Si | Alta |
| Baja | Si | Baja |
| Baja | No | Alta |
| Baja | No | Baja |
| Alta | Si | Alta |
| Alta | Si | Baja |
| Alta | No | Alta |
| Alta | No | Baja |

Tabla 13. Factores externos reconocimiento facial

Cada prueba se realiza en cada uno de los entornos anteriormente descrito, por lo tanto cada prueba constará de un total de 8 repeticiones, es decir, se realizará el reconocimiento facial a cada usuario por cada uno de los entornos.

La tabla 14 muestra la descripción de cada columna de la tabla que se utiliza para almacenar los usuarios que han realizado las pruebas de reconocimiento facial. Esta tabla se mostrará rellena en el apartado 5.3 *Resultados de las pruebas de reconocimiento facial.*

| Id Persona | Nombre | Sexo | Rasgos Faciales Destacables | Edad |
|---|---|---|---|---|
| Identificador único de la persona que realiza la prueba | Nombre del usuario | Masculino/Femenino | Rasgos faciales de la persona que realiza la prueba | Edad en años de la persona que realiza la prueba |

Tabla 14. Especificación tabla personas

La tabla 15 muestra la plantilla que se utiliza para mostrar los resultados de las pruebas realizadas. Al igual que la tabla 14, esta tabla aparecerá rellena en el apartado 5.3 de este documento.

| Id Prueba | Persona que Realiza el Entrenamiento | Persona que realiza el Reconocimiento | Repeticiones | Positivos | Tasa de positivos |
|---|---|---|---|---|---|
| Identificador de único de la prueba realizada | Es la persona que realiza el entrenamiento. Aquí se escribe el id de persona. | Es la persona que realiza el reconocimiento. Aquí se escribe el id de persona | Son las veces que se realiza el reconocimiento sobre la misma persona | El número de veces que se ha conseguido superar el reconocimiento facial | Positivos/repeticiones |

Tabla 15. Especificación pruebas de reconocimiento

Como se puede observar en la tabla 15, se diferencia la persona que realiza el reconocimiento de la persona que realiza el entrenamiento. Se ha diseñado de esta manera para diferenciar el reconocimiento sobre la persona usuaria de la aplicación, del reconocimiento de la persona no usuaria de la aplicación, es decir, un posible atacante. Si la persona que realiza el entrenamiento es la misma que realiza el

reconocimiento significa que es el usuario de la aplicación y por lo tanto el reconocimiento debería ser satisfactorio. Si por el contrario la persona que realiza el entrenamiento es diferente a la persona que realiza el reconocimiento, significará que se trata de una persona que desea acceder a la aplicación sin ser el usuario de la aplicación, por lo tanto un posible atacante. En este caso el reconcomiendo debería ser fallido. Es por ello que la tasa de positivos cuando la persona que realiza el entrenamiento es la misma que realiza el reconocimiento debería ser más alta que la tasa de positivos cuando las persona que realiza el entrenamiento es diferente de la persona que realiza el reconocimiento.

Como se ha comentado anteriormente, las repeticiones siempre serán 8 puesto que son los distintos escenarios externos que pueden influir en el reconocimiento facial.

# CAPÍTULO 4

## 4. Diseño

# 4.1 Diseño de Software

En esta sección se mostrará una descripción detallada de los distintos componentes identificados en el capítulo de análisis.

Mediante la Figura 8, se puede observar la arquitectura final del sistema. Se explicará el diseño de los siguientes componentes: *Módulo de identificación y entrenamiento, Servicio de seguridad, Módulo Central, Utilities y Base de datos.*

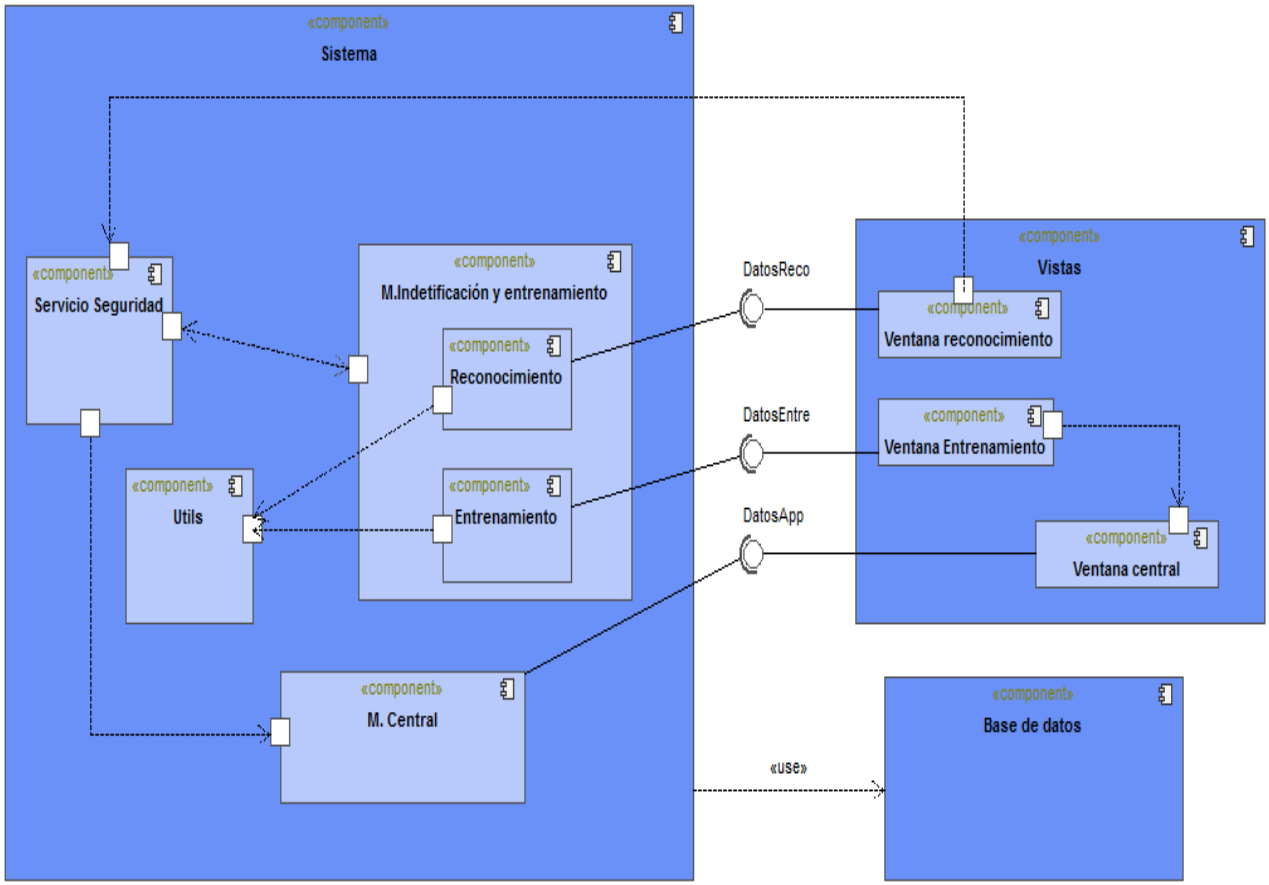



Figura 8. Arquitectura definitiva del sistema

### 4.1.1 Infraestructura utilizada para el entrenamiento

Mediante la Figura 9 se muestra el funcionamiento y los distintos componentes que forman el proceso de entrenamiento.

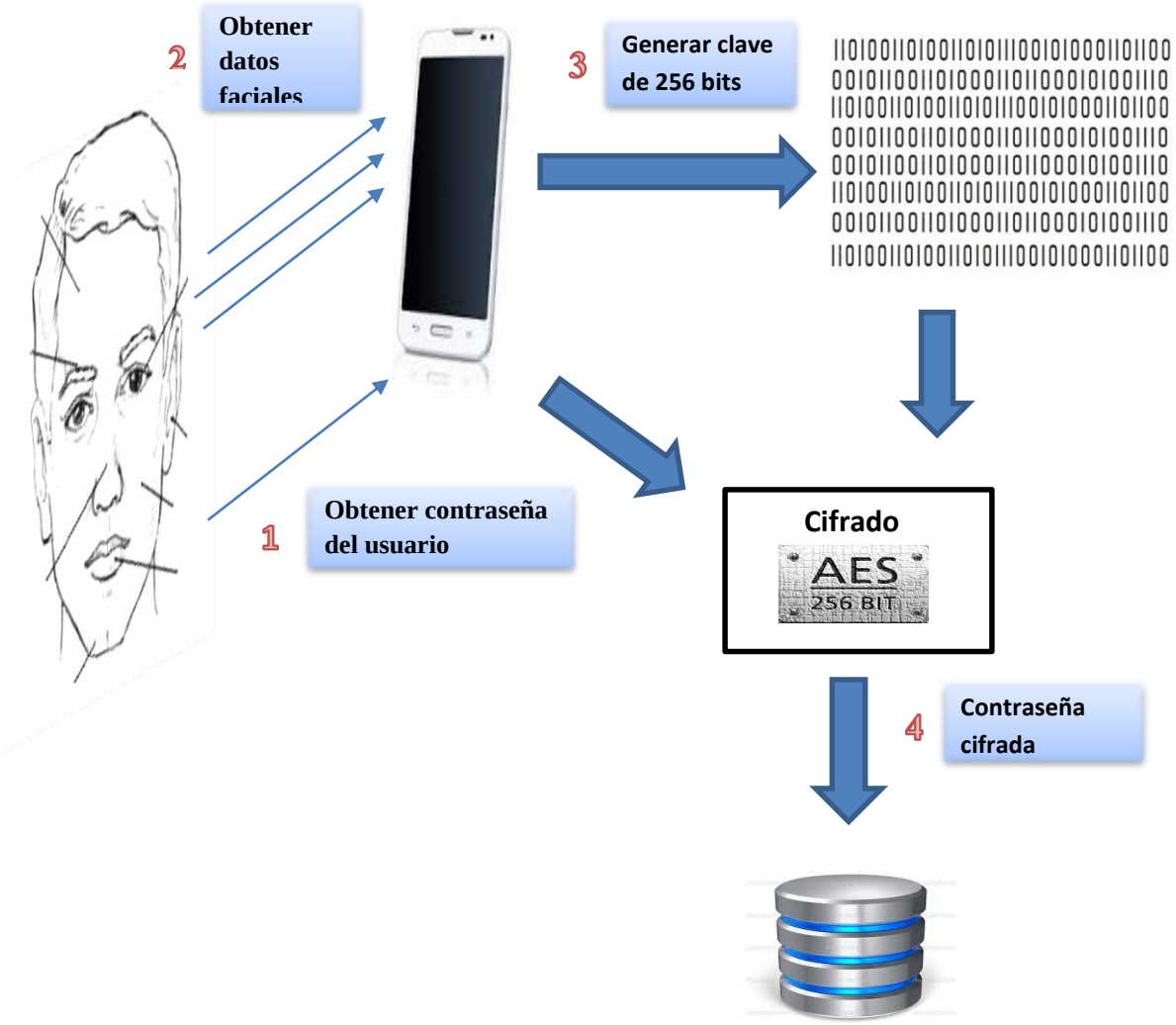



Figura 9. Infraestructura del entrenamiento de imágenes

El entrenamiento al igual que el reconocimiento, está formado por dos partes. Una primera parte en la cual se obtienen una serie de imágenes procedentes de la cara del usuario. A estas imágenes se les aplica una serie de transformaciones y filtros para que todas tengan el mismo formato. Se aplican las siguientes funciones:

- *cvCvtColor(ipl, ipl, CV_BGR2GRAY*): Esta función proviene de la librería de openCV y sirve para transformar la imagen en blanco y negro.
- *cvResize(ipl, ipl, CV_INTER_LINEAR) :* Función que proviene igualmente de la librería de openCV y sirve para redimensionar la imagen.
- *cvEqualizeHist(ipl, ipl):* Esta función sirve para normalizar el brillo y el contraste de la imagen.

Estas imágenes se utilizan para aplicar las funciones de la librería de openCV (*train()* y *predict()*) que nos permiten realizar un entrenamiento previo para el posterior reconocimiento. Cabe destacar que la información relevante proveniente del resultado de aplicar la función que realiza el entrenamiento (*train()*) se almacena en un fichero XML para no tener que realizar el proceso de entrenamiento cada vez que se quiera realizar el reconocimiento. De esta manera solo haría falta cargar la información del fichero.

La segunda parte del entrenamiento consiste en la creación de la clave y cifrado de la contraseña, que es el proceso que se muestra en la figura 9. En primer lugar el usuario introduce sus datos personales en el dispositivo, entre esos datos esta la contraseña. A continuación se obtienen los datos a partir de los distintos rasgos faciales de la cara.

Mediante estos datos se construye la clave de 256 bits. Por ultimo mediante el cifrado AES de 256 bits se cifra la contraseña del usuario con la clave generada y se almacena en la base de datos. [22]

### 4.1.2 Infraestructura utilizada para el reconocimiento

Mediante la Figura 10 se muestra el funcionamiento y los distintos componentes que forman el de reconocimiento.

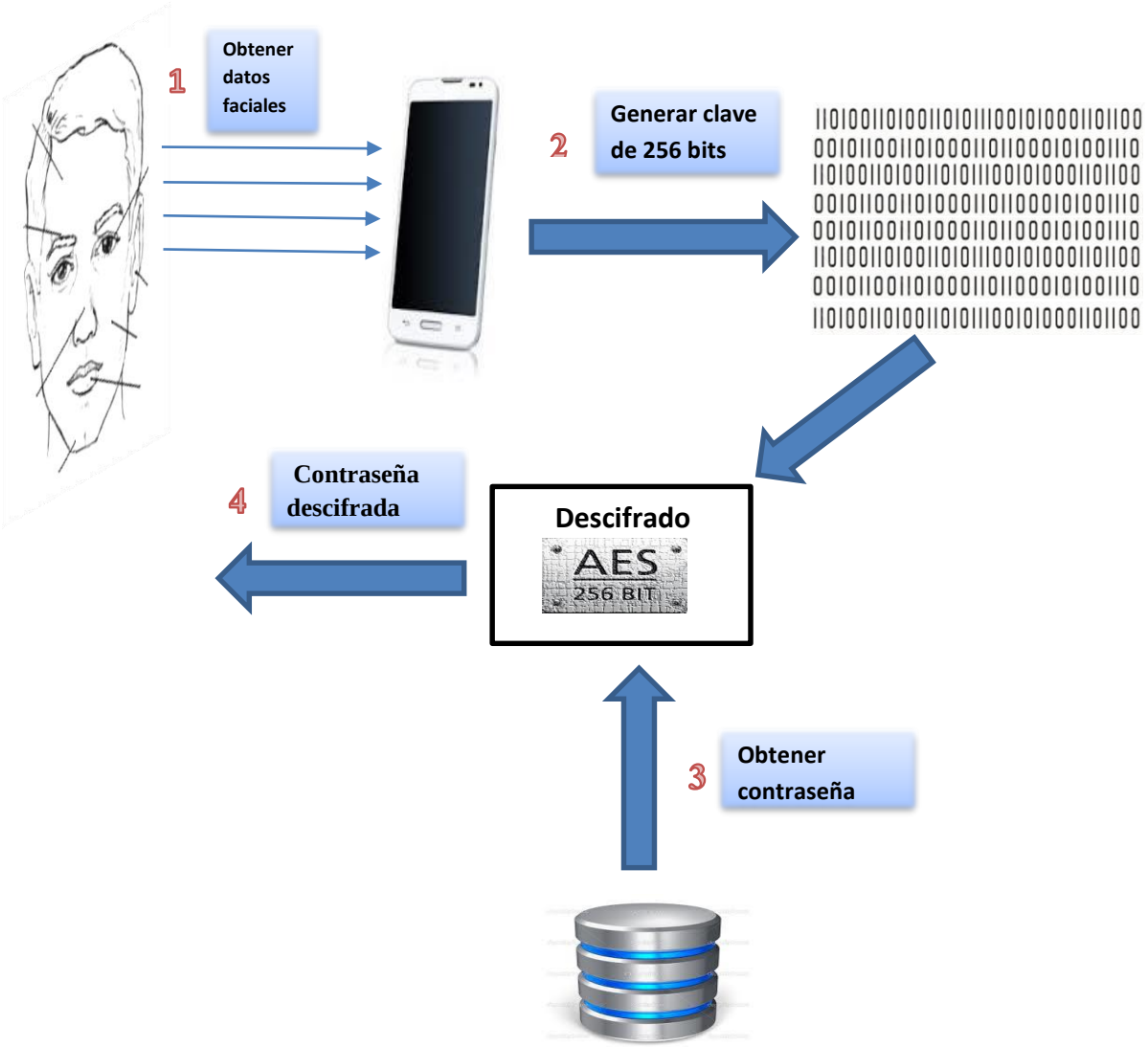



Figura 10. Infraestructura del reconocimiento de imágenes

Mediante esta figura se pretende explicar los distintos componentes y cómo interactúan entre sí a la hora de realizar el reconocimiento facial. La figura muestra la fase del reconocimiento facial en la cual se construye una clave a partir de los rasgos faciales del usuario. Mediante esa clave se intenta descifrar la contraseña del usuario que previamente se ha almacenado en la base de datos. Si se consigue descifrar la

clave, el reconocimiento será un éxito, en caso contrario se considerará reconocimiento fallido.

El reconocimiento facial se compone de dos fases, la primera fase se basa en utilizar las funciones de reconocimiento facial que otorgan las librerías de openCV.

A partir de una serie de iteraciones se comprueba cuantos aciertos se han obtenido del reconocimiento facial, si el número de aciertos está dentro de lo permitido se considera que se ha superado la primera fase del reconocimiento y comenzará la segunda fase del reconocimiento que es lo que se muestra en la figura 10.

La segunda fase del reconocimiento se basa en obtener una serie de valores a través de una serie de imágenes captadas por la cámara del dispositivo. A continuación se genera una clave de 256 bits mediante la composición de los valores procedentes de los rasgos faciales de la cara. Posteriormente se obtiene la contraseña que ha sido previamente cifrada en la fase de entrenamiento. Esta contraseña se obtiene de la base de datos, que aunque en el diagrama de la Figura 10 aparece separado del dispositivo, esta base de datos se encuentra interna en el propio dispositivo y no se trata de un elemento externo.

A partir de la clave de 256 bits generada y la contraseña cifrada se procede a realizar el descifrado de la contraseña mediante el cifrado AES de 256 bits. Si no ocurre ningún fallo durante el proceso de descifrado, significa que la clave generada es la misma que la clave que se generó durante la fase de entrenamiento y por consiguiente se da por válido el reconocimiento ya que la persona que realiza el entrenamiento y el reconocimiento es la misma.

### 4.1.3 Construcción de la Clave Única

En este apartado se pretende explicar el proceso de construcción y los distintos métodos que se han utilizado para la generación de la clave que se utiliza para el cifrado y descifrado de la contraseña. La longitud de la clave se acordó que fuera de 256 bits, ya que se pretende utilizar el cifrado AES de 256 bits, es necesario utilizar una clave de 256 bits. No se conoce un posible ataque mediante las computadoras actuales que puedan romper este tipo de cifrado [25]. Para conseguir una longitud de clave de 32 bytes (256 bits) se utilizan distintos métodos cuyos valores se componen entre sí para formar la clave. Los métodos utilizados provienen tanto del API de Android como de las funciones que otorga la librería de OpenCV. Los métodos utilizados para construir la clave son los siguientes:

### *Distancia de los ojos*

Se utiliza el método *eyesDistance()* que proviene del paquete *Android.media* del API de Android. En concreto pertenece a la clase FaceDetector.Face.

Mediante esta esta función se obtiene un valor decimal que marca la distancia entre los ojos del usuario. Esta distancia es diferente para cada cara de cada usuario, por lo tanto sirve para identificar a un usuario.

### *Ancho de la Cara*

Aunque el ancho de la cara por separado no es un valor que identifique de forma univoca a un usuario, este valor en conjunto con los demás valores si puede servir como valor que identifique una cara, ya que es posible encontrar dos caras con la misma medida del ancho de su cara, pero es más complicado que dos personas tengan la misma distancia entre los ojos y el mismo ancho de la cara y que coincidan también en los demás valores que se explican a continuación. Este valor lo proporciona la Liberia de OpenCV.

### *Alto de la cara*

Se obtiene de la misma forma que el ancho de la cara, y al igual que pasaba con el ancho de la cara, este valor por sí solo no identifica una cara de forma única, pero si en conjunto con los demás valores.

### *Momentos invariantes de HU*

Son considerados como el promedio ponderado de los pixeles de una imagen. Fueron propuestos por primera vez por Ming-Kuei HU en el año 1962. [23] Ming-Kuei HU postuló un conjunto de 7 momentos invariantes a la rotación, traslación y cambios de escala. Estos siete momentos, son siete valores que deben de ser muy parecidos en una misma imagen y diferentes entre imágenes distintas.

Para obtener estos valores de cada imagen se utiliza la librería de OpenCV, en particular la clase *Imgproc*. Esta clase nos proporciona una serie de métodos que son necesarios aplicar a la imagen para obtener los momentos de HU. Los pasos a seguir para obtener los momentos invariantes de HU son los siguientes:

1. **Conversión a escala de grises**. En primer lugar es necesario convertir la imagen a color en una imagen en blanco y negro. Para ello se utiliza el método *cvtColor:*

```
Imgproc.cvtColor(m, mm, Imgproc.COLOR_RGBA2GRAY);
```

2. **Filtrado de la imagen**. Para quitar el ruido es necesario filtrar la imagen. La forma más sencilla es remplazando cada pixel por la medida del

valor de los pixeles del alrededor. Este procedimiento se realiza mediante el método *GaussianBlur:*

```
Imgproc.GaussianBlur(mm, mm, new org.opencv.core.Size(5, 5), 2.2, 2);
```

3. **Detección de bordes**: Se trata de aplicar otro filtro más con la intención de detectar los bordes, es contraria al filtro de GaussianBur, ya que mediante este filtro se consigue detectar los bordes y formas de la imagen. La función de OpenCV que realiza esta tarea es *Canny:*

```
Imgproc.Canny(mm, mm, 80, 100);
```

4. **Detección de contornos:** El siguiente paso es la detección de contornos a partir de la imagen binaria. Un contorno se puede definir como la curva que une todos los puntos continuos, que tiene mismo color o intensidad. En OpenCV existe la función *findContours* que detecta los contornos de una imagen*:*

```
Imgproc.findContours(mm, contours, new Mat(), Imgproc.RETR_LIST,
        Imgproc.CHAIN_APPROX_SIMPLE);
```

Una vez realizado estos pasos previos, ya se puede proceder a obtener los momentos invariantes de Hu. La función de OpenCV encargada de obtener los momentos invariantes de HU a partir de los contornos de la imagen es *moments :*

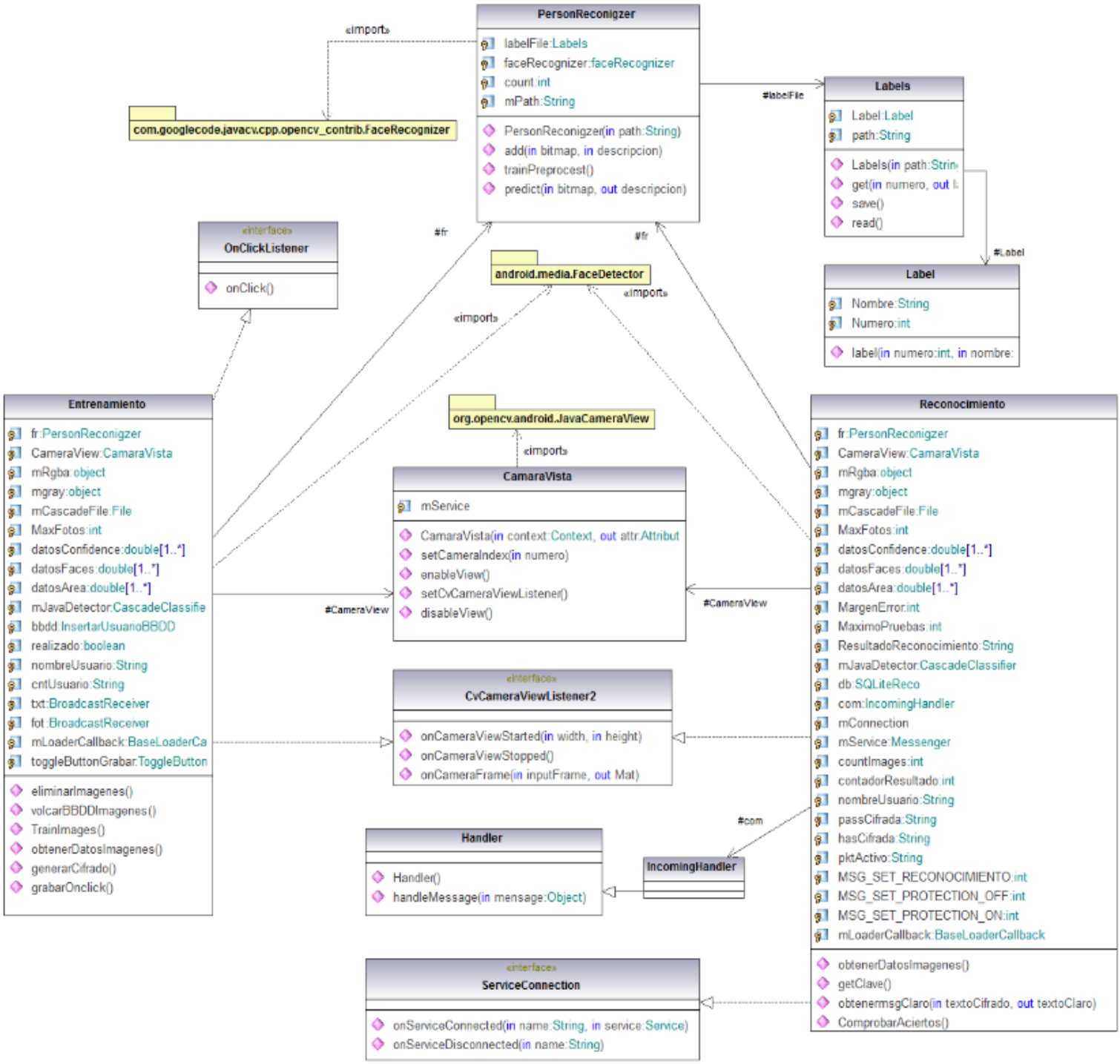
```
Moments mom = new Moments();
mom = Imgproc.moments(contours.get(0));
```

Mediante esta función obtenemos siete valores por cada imagen recibida. Se descartan los valores que sean iguales a 0 y se suman los 7 valores. El valor total se almacena en un array.

Se almacena un valor por cada imagen entrenada. Por lo tanto si el entrenamiento se realiza 15 imágenes del usuario, se almacenan 15 valores del ancho de la cara, la distancia de los ojos, alto de la cara y el total de los 7 momentos invariantes de Hu. Para los 15 valores del ancho de la cara, el alto de la cara, la distancia entre los ojos y los momentos invariantes de HU se obtiene el valor que más veces aparezca. A ese valor se le formatea para que ocupe 8 bytes de tamaño. Y mediante la composición de estos cuatro valores, se obtiene una clave de 32 bytes, 8 bytes por cada propiedad analizada.

### 4.1.4 Componente Módulo de identificación y entrenamiento

La Figura 11 muestra el diagrama de clases del componente Módulo de identificación y entrenamiento. El entrenamiento lo realiza la clase *Entrenamiento,*




mientras que el reconocimiento lo realiza la clase *Reconocimiento.*

Figura 11.Diagrama de clases Módulo entrenamiento y reconocimiento

Ambas clases (*Entrenamiento y Reconocimiento)* implementa la interfaz *cvCameraViewListener2 (*de la librería openCV) encargada de proporcionar los métodos necesarios que gestionan los eventos relacionados con la cámara, por ejemplo, la captura de frames provenientes de la cámara. [27]

Estas dos clases crean instancias de la clase *CameraVista,* que hereda de la clase *JavaCameraView,* perteneciente al paquete *org.opencv.android.JavaCameraView* dentro de la librería de openCV. La clase *CameraVista* es la encargada de la gestión de la cámara, se pueden realizar operaciones como activar o desactivar la cámara frontal del dispositivo.

La clase *Entrenamiento y* la clase *Reconocimiento* importan el paquete *android.media.FaceDetector* del API de Android. Este paquete contiene la clase *FaceDetector* que proporciona métodos que obtienen valores a partir de las imágenes de las caras de los usuarios, como por ejemplo, la distancia entre los ojos. Se utiliza para recopilar información de la cara del usuario para la construcción de la clave única.

La clase *PersonRecognizer* es la que contiene los métodos necesarios para realizar el entrenamiento y realizar el reconocimiento. Esta clase crea una instancia de la clase *Label*, que es la encargada de asociar cada imagen con un nombre y un identificador numérico. Mediante el método *add()* de la clase *PersonRecognizer* se almacenan las imágenes del usuario, que posteriormente se utilizarán cuando la clase *Reconocimiento* llame al método *train().*

Las clases *Entrenamiento* y *Reconocimiento* crean una instancia de la clase *PersonRecognizer* para utilizar los métodos que proporciona.

La clase *Reconocimiento* implementa la interfaz *ServiceConection* para poder comunicarse, mediante una instancia de la clase *IncomingHandle*r, con el servicio de seguridad.

### 4.1.5 Componente Servicio de seguridad

Este componente se encargar de monitorizar que aplicaciones están en funcionamiento. Comprueba si alguna de las aplicaciones que están en funcionamiento es de las seleccionadas por el usuario como aplicación a proteger. En ese caso el servicio de seguridad se encarga de lanzar el reconocimiento.

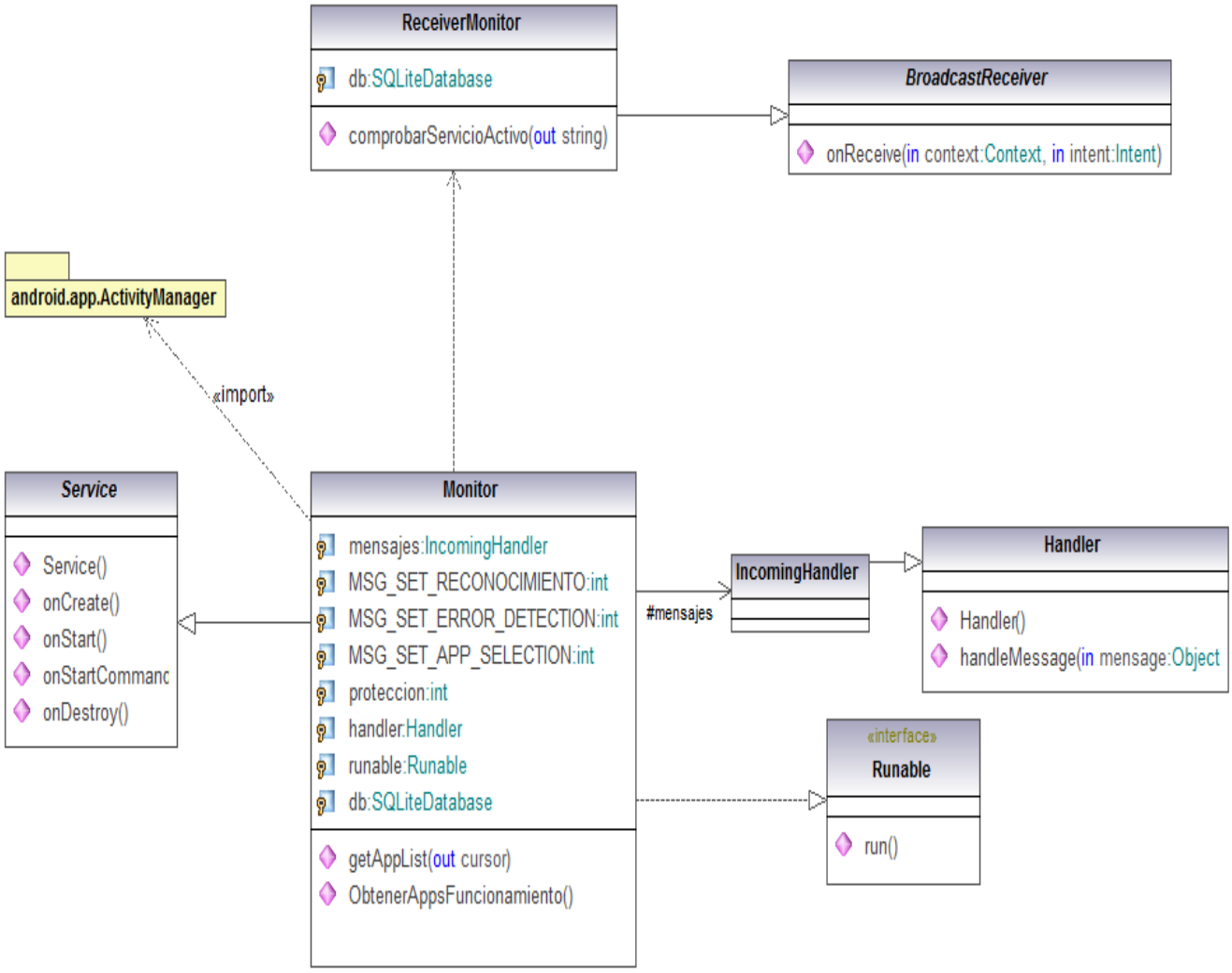



Figura 12 - Diagrama de clases Servicio de seguridad

La figura 12 muestra el diseño detallado del componente Servicio de seguridad.

La clase *Monitor* es la encargada de comprobar que aplicaciones se están ejecutando en el dispositivo. Para ello extiende de la clase *Service.* La clase *Service* está dentro del API de Android, y se trata de un proceso que se ejecuta de forma independiente al resto de la aplicación.

Para que el proceso se ejecute de forma continua es necesario que la clase *Monitor* implemente la interfaz *Runable*. Mediante el método *run()* de esta interfaz se consigue que se ejecute la funcionalidad deseada durante el tiempo que dure el servicio activo. El método *run()* por lo tanto se ejecuta de la misma manera como si se tratase de un bucle infinito. En este método se obtendrán las aplicaciones que están activas mediante el paquete *Android.App.ActivtyManager,* y se comprobará si algunas de las aplicaciones que están en funcionamiento son las mismas que el usuario selecciono como aplicaciones a proteger.

La clase monitor debe parar de realizar las comprobaciones antes mencionadas cuando el reconocimiento facial haya sido satisfactorio. Para ello crea una instancia de la clase *IcomingHandler,* mediante esta clase se tiene acceso a los mensajes recibidos provenientes de otros componentes. Estos mensajes pueden ser de dos tipos:

- *MSG_SET_RECONOCIMIENTO:* Es un número que identifica a un mensaje proveniente de la clase *Reconocimiento*.
- *MSG_SET_ERRO_DETECTION:* Es un número que identifica a un mensaje proveniente de la clase *ErrorDetection.*

Ambos mensajes contendrán un número que indicará si hay que detener la comprobación de aplicaciones en funcionamiento o si hay que activar dichas comprobaciones.

La clase *ReceiverMonitor* es un evento que se ejecuta cada vez que se inicia el dispositivo y comprueba si el servicio de seguridad debería estar activo.

## 4.1.6 Componente Módulo Central

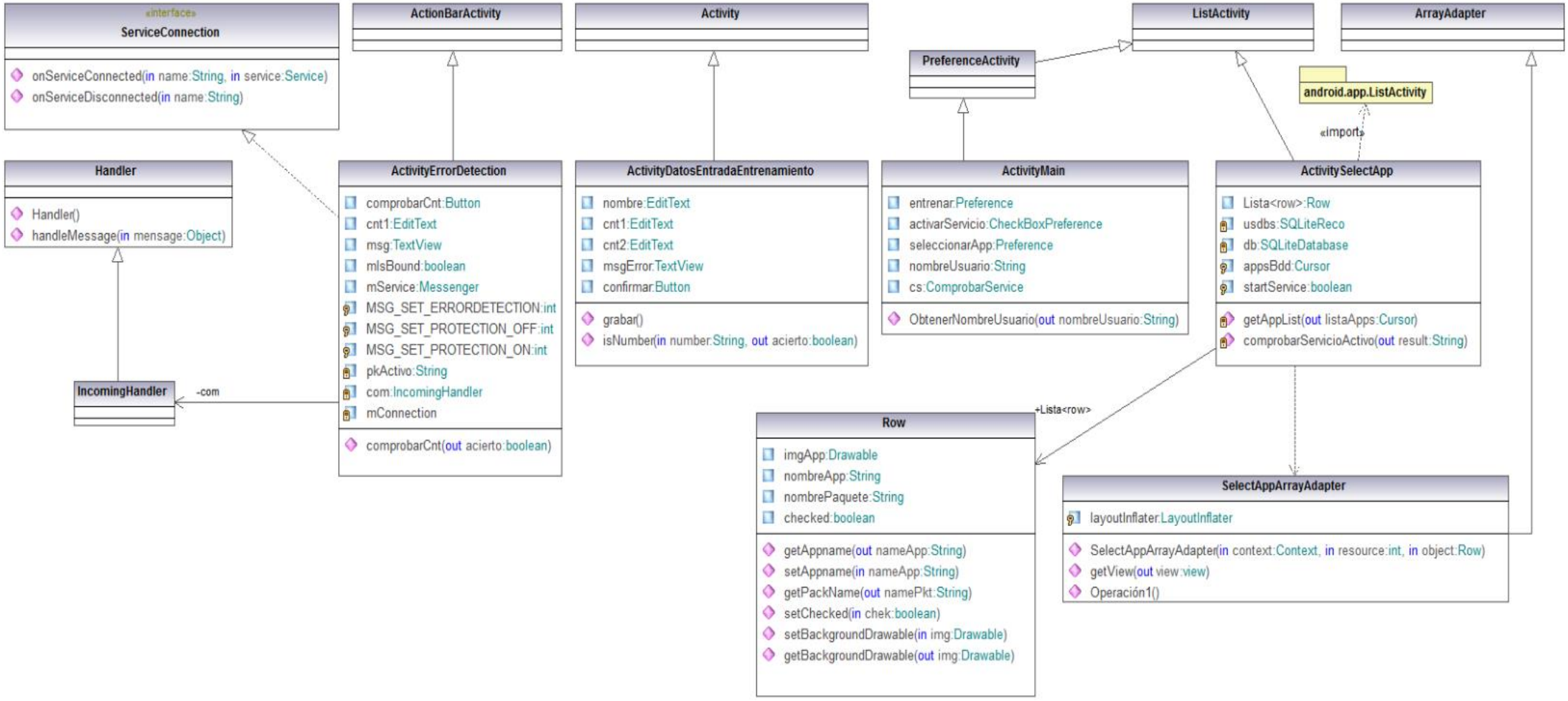



Figura 13 - Diagrama de clases Módulo Central

La Figura 13 muestra el componente *Modulo Central* detallado en cada uno de sus subcomponentes.

Las clases que comienzan con la palabra *Activity* son las encargadas de mostrar la vista al usuario y realizar la funcionalidad requerida en cada caso. La clase *ActivityMain* hereda de la clase *PreferenceActivity,* que es un tipo de actividad orientada a la selección de opciones. Cabe destacar que la clase *Activity* es la que se utiliza en Android para representar una actividad ("pantalla") de la aplicación. Mediante la clase *ActivityMain* el usuario puede realizar el entrenamiento, activar el servicio de seguridad o seleccionar las aplicaciones a proteger.

Si el usuario decide realizar el entrenamiento, antes debe registrarse, y para ello se utiliza la clase *ActivityDatosEntradaEntrenamiento*, mediante el método grabar.

La clase *ActivitySelectApp* es la encargada de mostrar las aplicaciones que están instaladas en el dispositivo. Mediante la clase *SelectAppArrayAdapter* se consigue gestionar el comportamiento de cada elemento de la lista. La clase *Row* es la que contiene los elementos necesarios que tendrá cada fila de la lista, que serán el nombre de la aplicación y una imagen de la misma. Por lo tanto la clase *ActivitySelectApp* para mostrar las aplicaciones, creará una lista de tipo *Row*.

La clase *ActivityErrorDetection* representa la "pantalla" que se muestra en caso de fallo de reconocimiento. Contiene el método comprobarCnt() que se encarga de comprobar si la contraseña que ingresa el usuario es la correcta. Esta actividad debe comunicar al servicio de seguridad si el usuario tiene permiso para acceder a la aplicación protegida, para ello (al igual que la clase reconocimiento) implementa la interfaz *ServiceConnection* y crea una instancia de la clase IncomingHandler.

### 4.1.7 Componente Base de datos

La Figura 14 muestra el componente Base de datos con todas las clases y relaciones que lo componen.

La clase *SQLLiteReco* será donde se definan cada una de las tablas necesarias. Esta clase hereda de la clase *SQLiteOpenHelper* que contiene los métodos necesarios para abrir o cerrar la base datos. Para realizar gestiones en la base de datos, ya sea modificar, eliminar o insertar datos se utiliza la clase *SQLiteDatabase*. Una vez analizado que operaciones son más costosas en términos de complejidad y tiempo sobre la base de datos se decide por crear varias tareas asíncronas (tareas que se ejecutan fuera del hilo principal de la aplicación).

La clase *comprobarService* es una tarea asíncrona cuyo principal cometido es almacenar en la base de datos un valor indicando si el servicio está activo o no.

La clase *InsertarUsuarioBDD* es una tarea asíncrona que se encarga de insertar los datos asociados al usuario, insertar el nombre, la contraseña cifrada, y la función hash() asociada a la contraseña sin cifrar. [24]

La clase *insertarDatosApp* es una tarea asíncrona que se encarga de almacenar las aplicaciones seleccionadas por el usuario. En concreto almacena el nombre del paquete de la aplicación, un valor que indica si la aplicación ha sido seleccionada por el usuario y por último el nombre de la aplicación.

Las tablas creadas son las siguientes:

- *Tabla Usuario*: Nombre (TEXT) PRMARY KEY, contraseña (TEXT) y hash (TEXT).
- *Tabla Service*: Activado (TEXT) PRIMARY KEY.
- *Tabla AppSelected* : nombrePakete (TEXT), isCheked (TEXT), labelName(TEXT) PRIMARY KEY

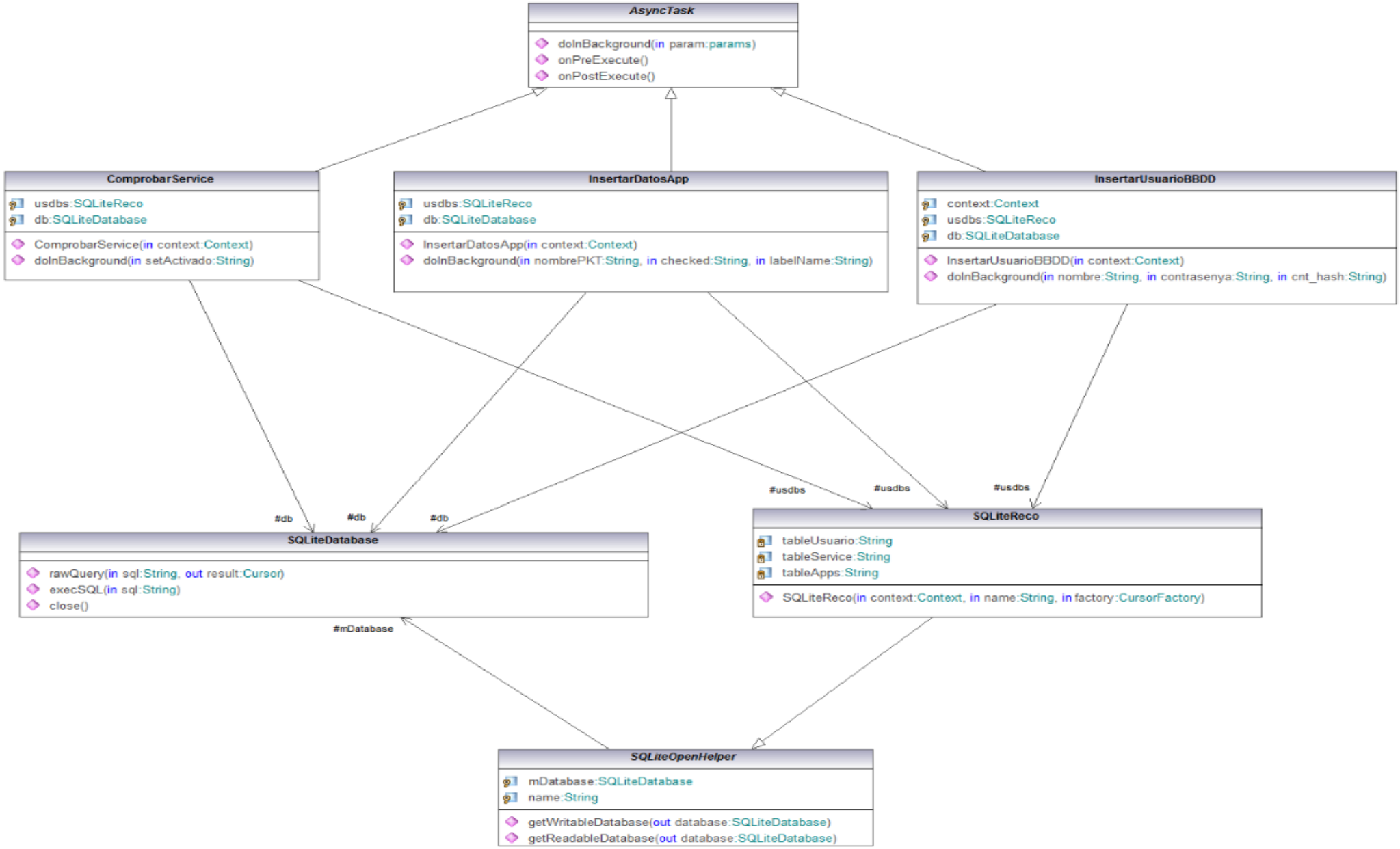



Figura 14 - Diagrama de clases Base de datos

### 4.1.8 Componente Utilities

Este componente que se muestra detallado mediante la Figura 15, proporciona las herramientas necesarias para la construcción de la clave que se genera a partir de los datos obtenidos de las diferentes imágenes del Usuario. También se encarga de facilitar las herramientas necesarias para el cifrado de la contraseña y la construcción de la función hash().

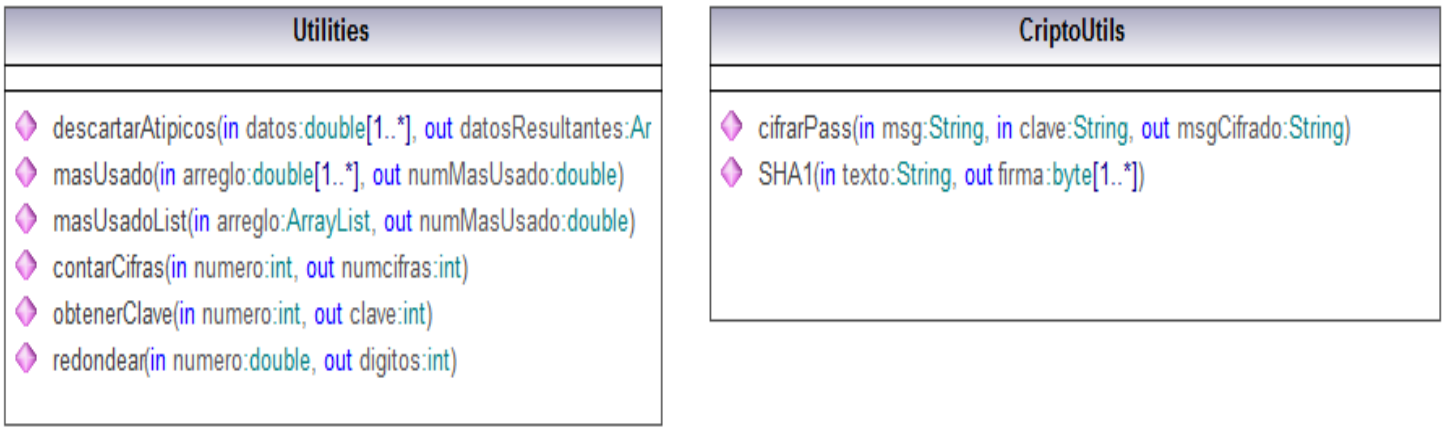



Figura 15 - Diagrama de clases Utilities

Este componente está formado por la clase *Utilites* que contiene los métodos necesarios para la creación de la clave única. Y la clase *CriptoUtils* que proporciona el método *cifrarPass* para cifrar la contraseña, y el método SHA1 para generar la función resumen.

# 4.2 Diagramas de Secuencia

Realizado el diseño del software se procede a mostrar las distintas iteraciones entre las diferentes clases y también entre el usuario y la aplicación.

Mediante una serie de diagramas de secuencia se muestran las distintas iteraciones entre el usuario y la aplicación en la ejecución de los casos de uso que están definidos en el punto 3.6. Cabe destacar que no se mostrarán el flujo alternativo de los distintos casos de uso, dado que en muchos casos el flujo normal y el alternativo suelen ser el mismo o se diferencian en aspectos que no son significativos en las distintas iteraciones.

En algunos casos se han eliminado determinadas iteraciones que no aportaban información, con el objetivo de reducir el tamaño de los diagramas y mejorar el entendimiento de los mismos. Es el caso de las iteraciones con la base de datos que no sean mediante las tareas asíncronas.

## 4.2.1 Introducir nombre y contraseña (CU-01)

Para este caso de uso, se puede observar mediante el diagrama de la Figura 16, que son tres clases las que interaccionan entre si junto con el usuario.

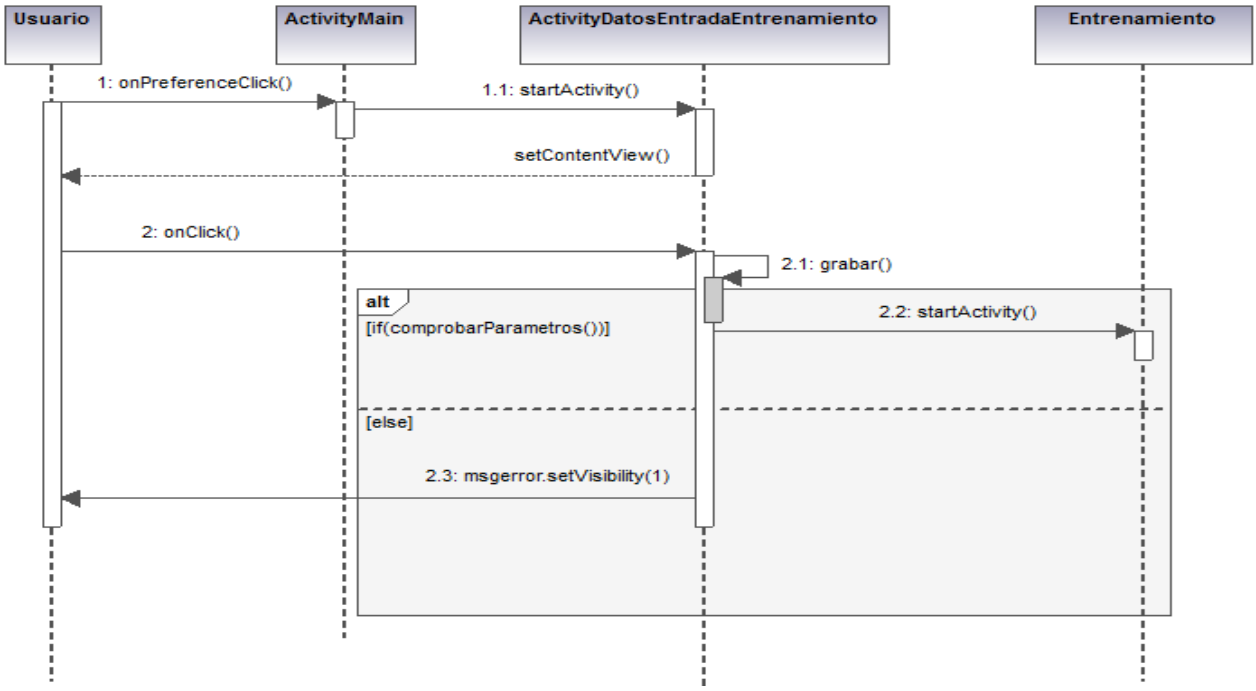



Figura 16 - Diagrama de secuencia CU-01

En primer lugar, el usuario tendrá que pulsar el botón de entrenamiento. En ese momento, la clase *ActivityMain* lanza la actividad *ActivityDatosEntradaEntrenamiento*. Esta actividad muestra al usuario un formulario para que el usuario introduzca su nombre y contraseña dos veces.

Cuando el usuario introduzca los datos y pulse el botón de aceptar, la actividad *ActivityDatosEntradaEntrenamiento* comprobará si todos los datos cumplen los requisitos RNF-07, 08, 09, 10,11. En caso de que sí cumplan los requisitos, se lanzará la clase entrenamiento para comenzar el entrenamiento.

En caso de que no se hayan cumplido los requisitos, se mostrará un mensaje al usuario indicando el error.

### 4.2.2 Realizar entrenamiento (CU-02)

Para este caso de uso, el usuario deberá interaccionar varias veces con los distintos componentes de la aplicación. Dado que el usuario necesita registrarse en la aplicación, es necesario que se produzcan las interacciones entre el usuario y la aplicación que se muestran en el punto 4.2.1 referente al caso de uso CU-01.

La Figura 17, muestra las iteraciones que suceden a partir de que el usuario haya introducido correctamente los datos en la aplicación y se haya lanzado el entrenamiento. La clase entrenamiento abre la cámara frontal del dispositivo y el usuario podrá verse la cara mediante la pantalla del dispositivo. Es entonces cuando la clase entrenamiento inicializa el objeto *fr* de la clase *PersonRecognizer*, pasándole al constructor como parámetro la ruta donde se almacenarán las imágenes del usuario temporalmente. A continuación se inicializa el objeto *mJavaDetector* mediante el fichero *lbpcascade.xml*. Este fichero contiene una base de datos de diferentes caras, se utiliza para poder detectar mediante la cámara que objeto es una cara humana.

Mediante el método *onCameraFrame* de la interfaz *CvCameraViewListener2* se capturan cada uno de los frames que se procesan desde la cámara. Por cada frame procesado se comprueba mediante el objeto *mJavaDetector* si el frame pertenece a una cara humana. Si se detecta que es una cara, se muestra un mensaje al usuario indicando que se ha detectado una cara. Y además se habilita el botón por el cual el usuario se podrá realizar las fotografías a si mismo necesarias para completar la fase de entrenamiento.

Cuando el usuario pulse el botón, la clase entrenamiento llamará al método *add*, al cual se le pasa como parámetro el nombre del usuario, la imagen del mismo y un número asociado a dicha imagen. La clase *PersonRecognizer* se encargará de almacenar las imágenes en una carpeta del dispositivo temporalmente.

Una vez que se ha ejecutado el método add() se procede a obtener los valores necesarios para la construcción de la clave (como se ha construido la clave viene explicado en el punto 4.1.3) Este procedimiento sucederá repetidas veces hasta que el usuario se haya realizado el número máximo de fotos indicado por la variable *MaxFotos* que por defecto se ha definido en 15.

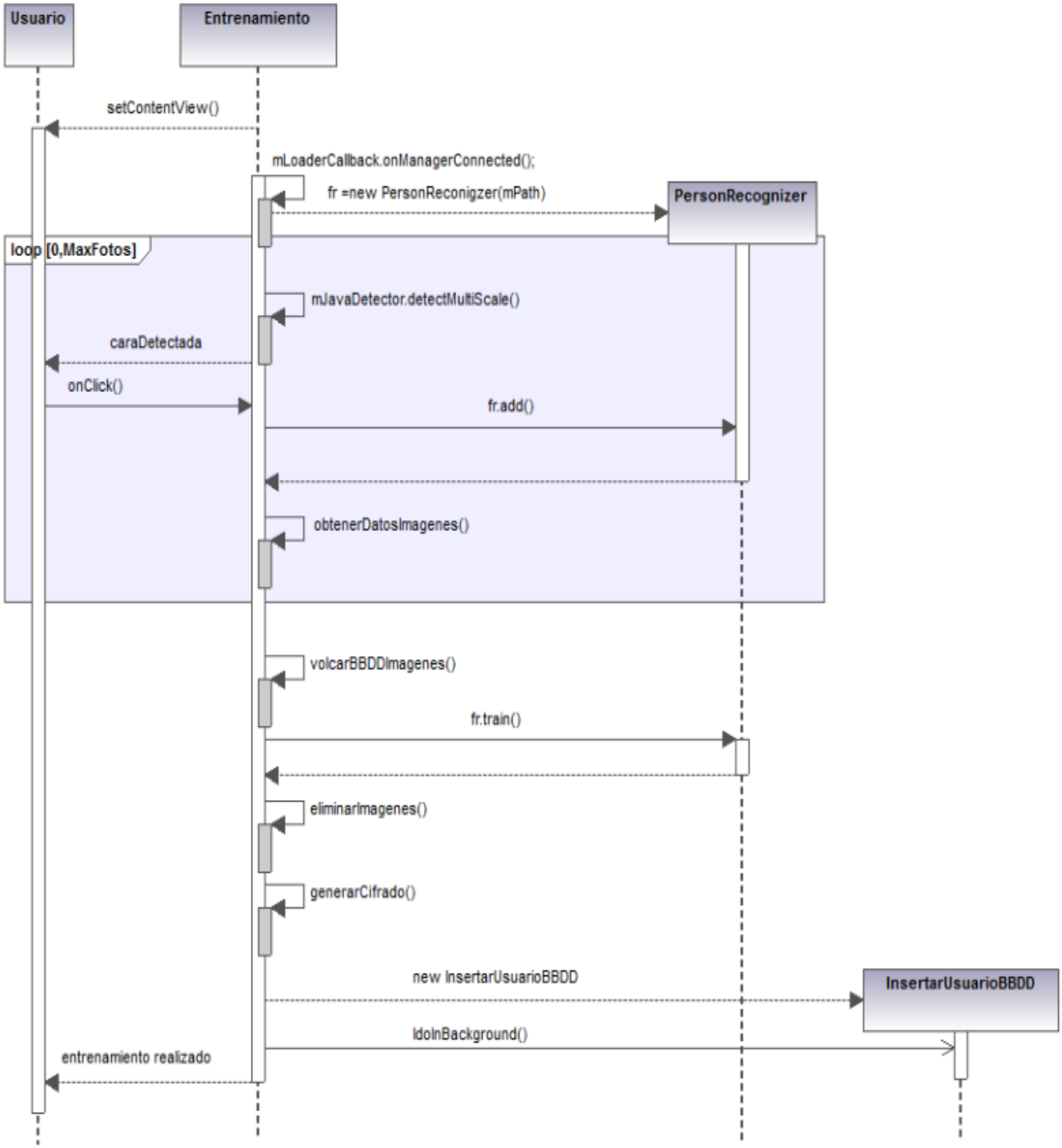



Figura 17 - Diagrama de secuencia CU-02

Cuando el usuario haya alcanzado el límite de fotografías permitidas, se procede a almacenar las imágenes que forman la base de datos de imágenes en la misma carpeta donde están situadas las imágenes del usuario.

El siguiente paso es llamar al método *trainPreprocest*() de la clase *PersonRecognizer*, este método se encarga de obtener las imágenes que se han almacenado previamente, entrenarlas mediante el método *train()* de la clase *faceRecognizer* perteneciente a la librería de *openCV* y almacenar el resultado del entrenamiento en un fichero XML.

Por último se eliminan las imágenes almacenadas en un directorio del dispositivo, se genera el cifrado de la contraseña del usuario mediante la clave generada y se almacena mediante la tarea asíncrona *InsertarDatosUsuario()*, la contraseña cifrada, el nombre del usuario y el resultado de aplicar la función hash() a la contraseña sin cifrar.

Al finalizar todo este procesamiento se muestra un mensaje al usuario indicando que el entrenamiento ha finalizado.

### 4.2.3 Activar/Desactivar Servicio de Seguridad (CU-03,04)

El siguiente diagrama de secuencia de la Figura 18 muestra la ejecución de los casos de uso activar servicio de seguridad y desactivar servicio de seguridad. Ambos casos de uso realizan operaciones similares que se puede mostrar en un mismo diagrama de iteraciones.

En primer lugar, la clase *ActivityMain* comprobará accediendo a la base de datos si existe un usuario registrado. En ese caso la clase *ActivityMain* habilitará el casillero para que el usuario pueda hacer clic en él. En el caso de no existir el nombre del usuario en la base de datos, la clase *ActivityMain* inhabilitará el casillero.

El usuario si está registrado podrá entonces activar el servicio de seguridad, activando el casillero, entonces la clase *ActivityMain* lanzara el servicio de seguridad. En caso de que el usuario desactive la casilla, se parará el servicio de seguridad.

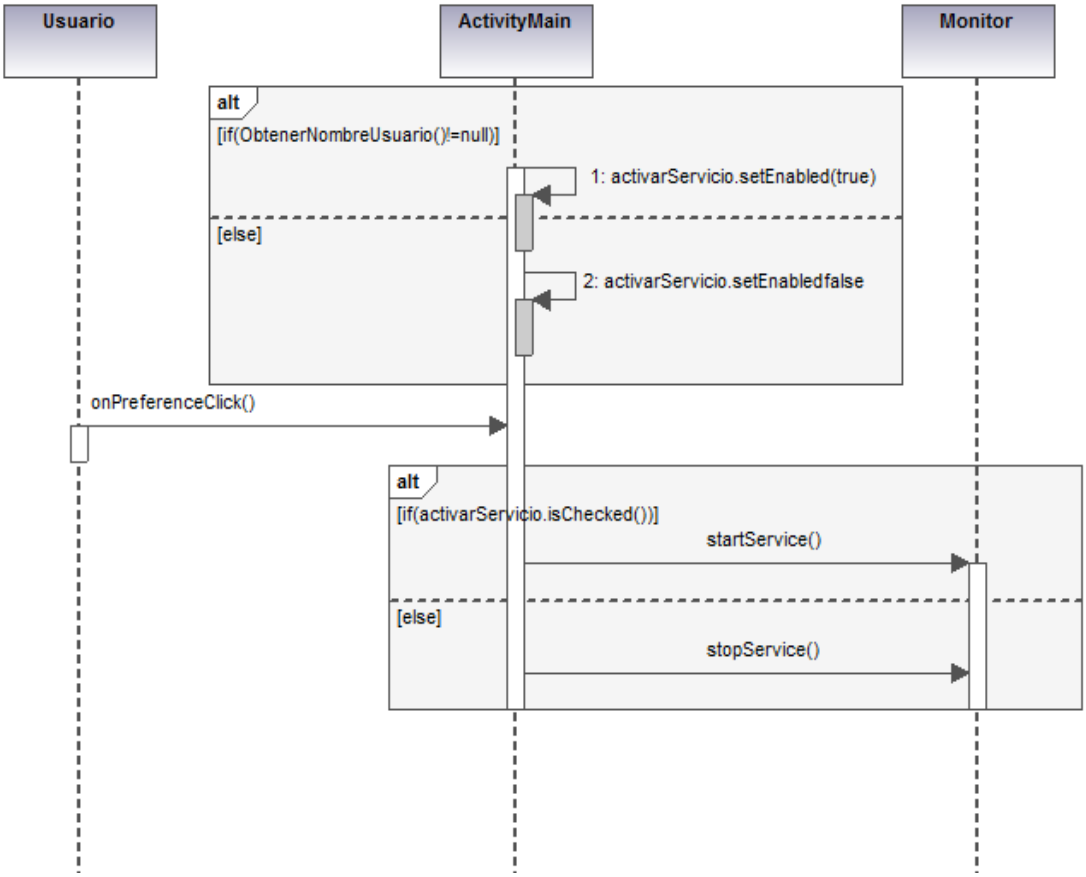



Figura 18 - Diagrama de secuencia CU03, CU-04

### 4.2.4 Seleccionar aplicación a proteger (CU-05)

La finalidad de este diagrama de secuencia es mostrar las distintas iteraciones entre los distintos objetos de la aplicación y el usuario que se suceden al seleccionar un conjunto de aplicaciones a proteger.

La Figura 19 muestra las diferentes acciones que se suceden entre las diferentes clases.

En primer lugar, el usuario debe seleccionar el botón que se encuentra en la clase *ActivityMain*, una vez presionado, la clase *ActivityMain* lanza la clase *ActivitySelectApp*.

A continuación se obtienen el nombre de los paquetes de las aplicaciones que están instaladas en el dispositivo. Se accede a la base de datos y se obtienen (si existen) el nombre de las aplicaciones que el usuario haya seleccionado. La primera vez el usuario acceda a la aplicación obviamente no aparecerán ninguna aplicación seleccionada como aplicación a proteger en la base de datos.

La siguiente tarea es la de crear cada fila por cada aplicación instalada en el dispositivo. Mediante un bucle de tantas iteraciones como aplicaciones instaladas, se crea una instancia de la clase *Row* y por cada instancia que se crea en cada iteración del bucle, se asigna a la fila el nombre de la aplicación y su imagen asociada.

Siguiendo dentro del bucle, se comprueba si la aplicación instalada, de la que se está obteniendo el nombre y la imagen para asignárselas a la fila, está registrada como aplicación a proteger en la base de datos. Si es cierto, se habilita el casillero que contiene la fila. Si no es cierto se deshabilita.

Por tanto, por cada iteración del bucle se crea una fila nueva y se van añadiendo a una lista que contendrá todas las filas. Al final del bucle se crea una instancia de la clase *SelectAppArrayAdapter*, esta clase se encarga de inicializar la vista para mostrar la lista de filas que se ha creado anteriormente. Además, controla los eventos que se ejecutan cuando el usuario desee pulsar sobre el casillero de cada fila. Como parámetro del constructor de la clase *SelectAppArrayAdapter* se le pasa la lista que contiene las filas.

Después de ejecutarse las tareas mencionadas, el usuario podrá visualizar en su pantalla del dispositivo las aplicaciones que tiene instaladas y tendrá la opción de seleccionar una o varias de ellas. Si el usuario selecciona el casillero de una de las aplicaciones de la lista, se generara un evento controlado por la clase *SelectAppArrayAdapter*. En el caso de que el casillero este habilitado se almacenara en la base de datos una “S” en concepto de aplicación con seguridad habilitada o en el caso de que el usuario seleccione el casillero para deshabilitar la seguridad, se almacenará una “N” en concepto de aplicación con seguridad deshabilitada.

Cabe destacar que estas operaciones sobre la base de datos se realizan mediante la tarea asíncrona *InsertarDatosAPP*

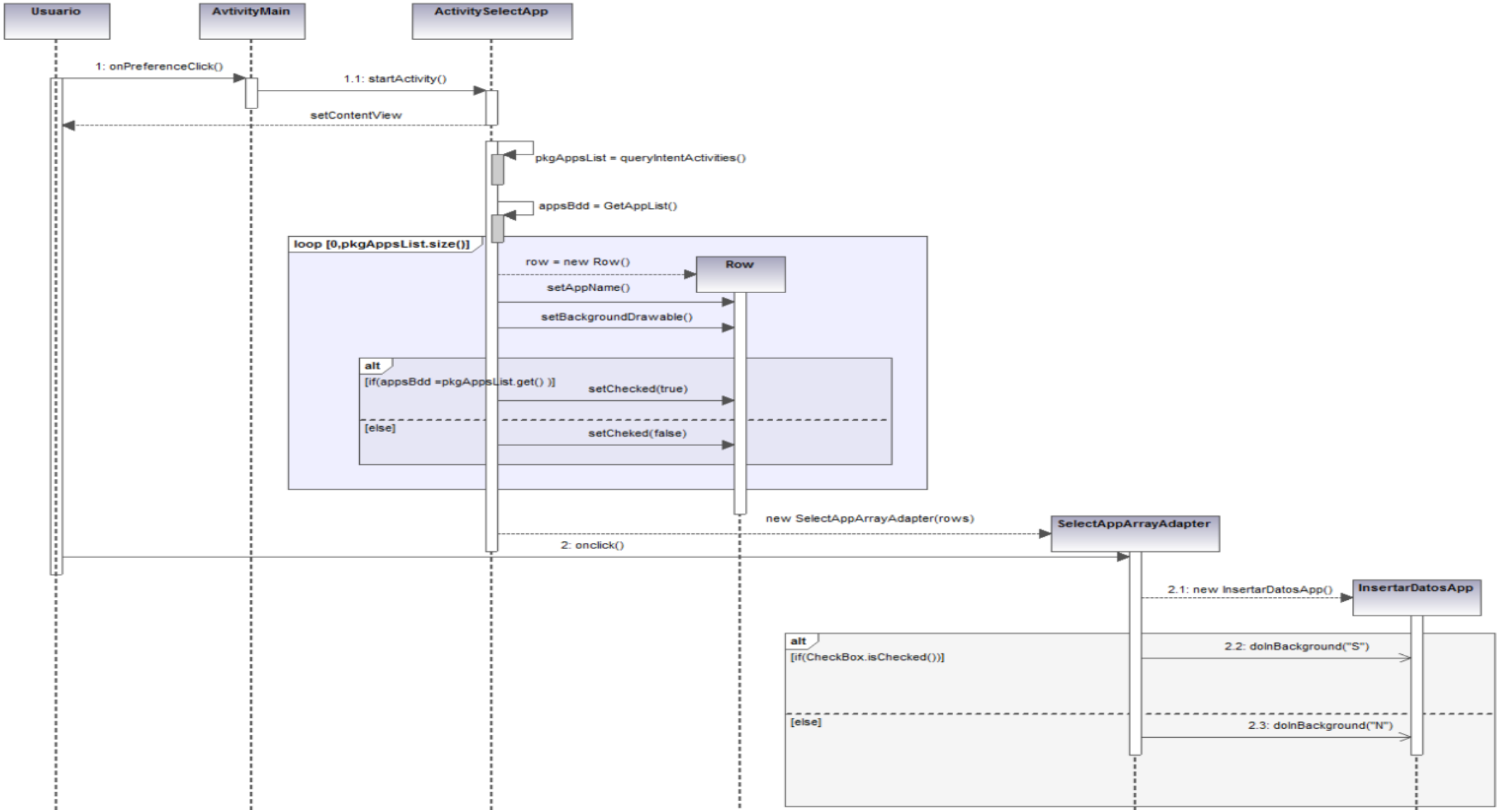



Figura 19 - Diagrama de secuencia CU-05

### 4.2.5 Acceder a App protegida (CU-06)

El Diagrama de secuencia que se muestra mediante la Figura 20 Identifica al caso de uso CU-06. Este caso de uso hace referencia al reconocimiento facial que se realiza en caso de que el usuario desee acceder a una aplicación protegida.

Al igual que con diagrama de secuencia para el caso de uso CU-02, para una mayor comprensión se han eliminado las iteraciones con la base de datos.

Mediante este diagrama se puede observar que son cinco los objetos que interactúan entre sí. Cabe destacar que se parte con la idea que el usuario ha realizado el entrenamiento previamente y ha activado el servicio de seguridad. Por lo tanto, inicialmente tenemos el servicio de seguridad activo comprobando en bucle si alguna de las aplicaciones en ejecución coincide con las que el usuario eligió previamente. En el caso de comprobar que el usuario ha arrancado una aplicación protegida, el servicio de seguridad lanzará la clase *Reconocimiento.* La clase *Reconocimiento* a diferencia de la clase *Entrenamiento* no muestra al usuario las imágenes que la cámara va procesando (lo que la cámara ve). En este caso, el usuario solo podrá visualizar una pantalla en negro. Es en este instante cuando comienza el proceso de reconocimiento.

En primer lugar se crea una instancia de la clase *PersonRecognizer* y al igual que la clase *Entrenamiento* se inicializa el objeto *mJavaDetector* mediante el fichero que contiene la base de datos de las caras.

A continuación comienza un bucle de tantas iteraciones como marque la variable *MaximoPruebas* que para este caso esta puesta a 20. Cada iteración coincide con el procesamiento de cada *frame* que llega de la cámara. Cada imagen que llega desde la cámara se pasa como parámetro de la función *predict* de la clase *PersonRecognizer.* Mediante esta función se consigue obtener a partir de una imagen, un nombre asociado a esa imagen, si el nombre coincide con el del usuario entonces supondrá que se reconoce la cara del usuario. Hasta que no se llegue al máximo de pruebas, se van almacenando los resultados obtenidos. También se irán obteniendo los datos necesarios para la construcción de la clave.

Una vez que se tienen tantos resultados y tantos datos como marque la variable máximo de pruebas, se contabiliza el número de aciertos sobre el total de resultados. En el caso de tener más aciertos que el máximo de pruebas menos un margen de error se considera que ha pasado la primera fase del reconocimiento. Si no, se considera que no ha pasado la primera fase del reconocimiento y por lo tanto el reconocimiento es fallido y se lanzará la pantalla de introducción de contraseña.

La segunda fase consiste en comprobar si mediante la clave generada se puede descifrar la contraseña almacenada en la base de datos. En caso de no poder descifrarla se da por fallido el reconocimiento y se lanza la pantalla de introducción de contraseña (*ActivityErrorDetection*).

Si por el contrario el descifrado obtiene éxito, se considera que el reconocimiento ha sido satisfactorio y se lanzará la aplicación que el usuario quería acceder anteriormente.

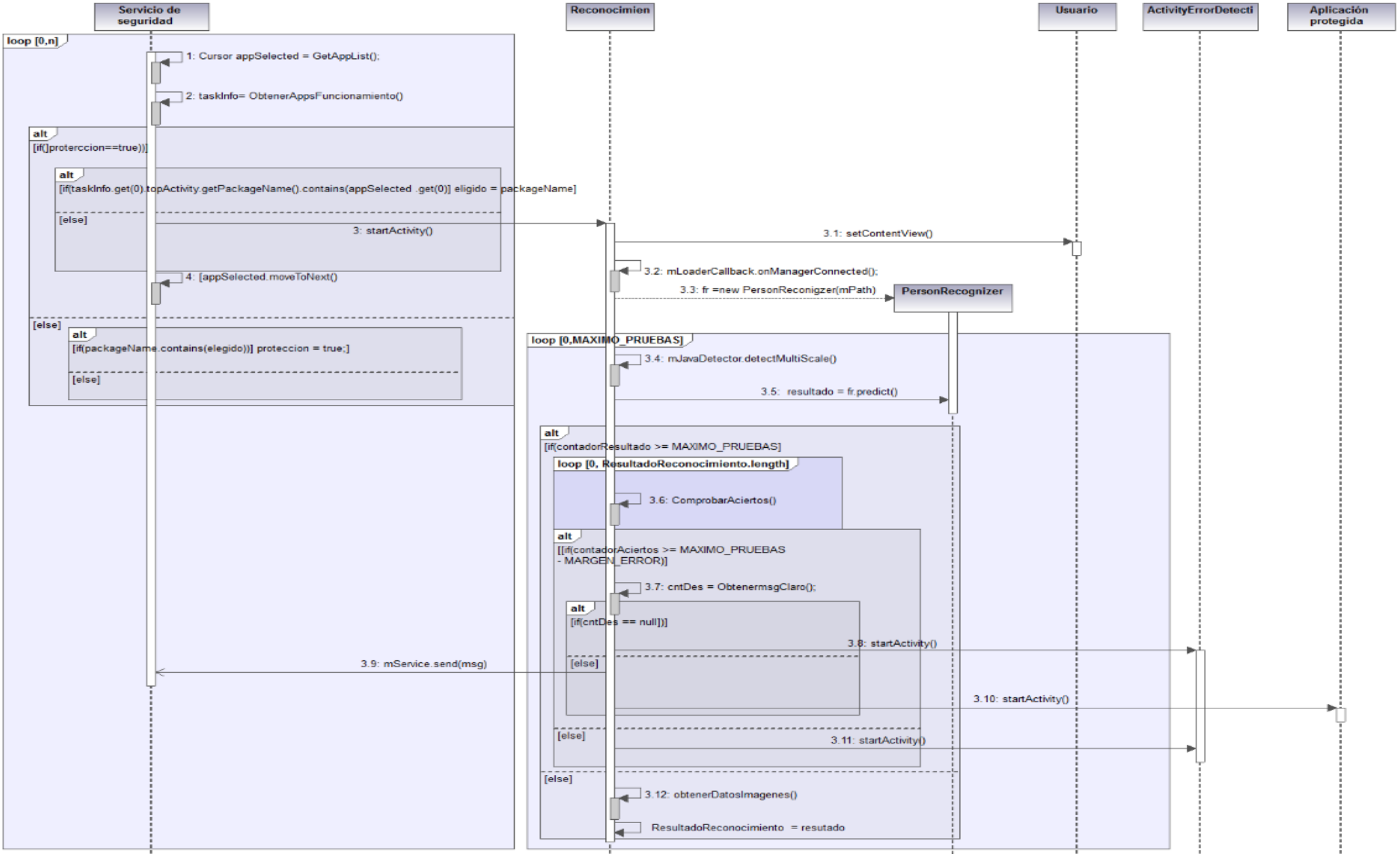



Figura 20 - Diagrama de secuencia CU-06

### 4.2.6 Acceder a App protegida mediante contraseña (CU-07)

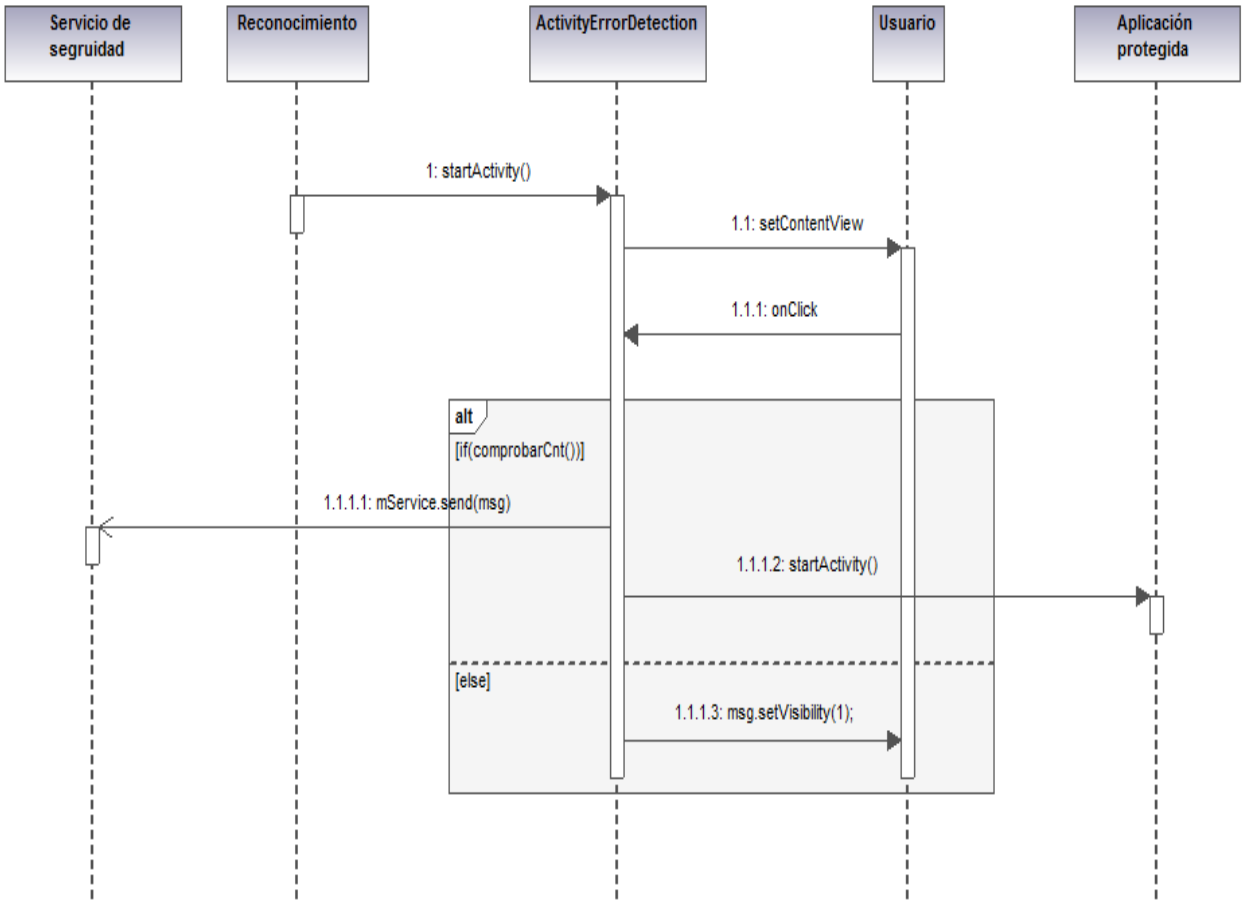



Figura 21 - Diagrama de secuencia CU-07

El diagrama que de la Figura 21, muestra las distintas iteraciones que se realizan en caso de fallo de autenticación y por consiguiente la posterior introducción de contraseña para acceder a la aplicación. En este diagrama las distintas iteraciones suceden a partir de que no se haya reconocido al usuario que realiza el entrenamiento. Según el requisito funcional RF-09, en caso de fallo de autenticación el usuario podrá acceder a la aplicación introduciendo la contraseña.

Cuando no se reconoce al usuario, la clase Reconocimiento lanza *ActivityErrorDetection*, esta clase es la encargada de mostrar la vista en la cual aparecerá un formulario para introducir la contraseña.

Cuando el usuario introduzca la contraseña, la clase *ActivityErrorDetection* comprobará accediendo a la base de datos si el resultado de aplicar la función hash() a la contraseña coincide con el resultado de la función hash aplicada a la contraseña que

se almaceno cuando el usuario se registró al realizar el entrenamiento. En caso de acierto la clase *ActivityErrorDetection* comunicará al servicio debe desactivar la protección de la aplicación que el usuario desea acceder y lanzará dicha aplicación (Aplicación protegida). En caso de fallo al comprobar las funciones hash() de las contraseñas, se comunicará mediante un mensaje de error al usuario que la contraseña introducida es incorrecta.

# CAPÍTULO 5

## 5. Implementación del Software

# 5.1 Decisiones de implementación

En este apartado se muestran las decisiones que se han tomado a la hora de implementar la aplicación tras haber realizado la fase de análisis y diseño. Las decisiones de implementación vienen determinadas por la especificación de requisitos. En algunos casos, la implementación de algunas funcionalidades que se indican mediante los requisitos, ha ocasionado que surjan nuevas funcionalidades que no estaban expresadas en la especificación de requisitos.

A continuación se muestran las decisiones más relevantes e importantes de implementación que se han tomado.

## 5.1.1 Decisiones de implementación del módulo Entrenamiento y Reconocimiento

A continuación se exponen distintos problemas y cuestiones asociadas con la implementación que se consideren de interés:

1. En un primer lugar, como viene definido en la fase de análisis y posteriormente en la fase de diseño, se tenía pensado utilizar un máximo de 15 imágenes del usuario para realizar el entrenamiento. Durante la fase de pruebas se comprobó que la primera parte del reconocimiento no funcionaba correctamente y se decidió por incrementar el número de imágenes que se realiza el usuario durante el entrenamiento. Al realizar nuevamente las pruebas se comprobó que el resultado del reconocimiento no variaba con respecto a las pruebas para 15 imágenes. Entonces se decidió utilizar más imágenes de personas aleatorias para el almacén de imágenes que acompañan a las del usuario. Al realizar las pruebas se comprobó que los resultados mejoraban considerablemente.
2. Otra de las decisiones tomadas fue sobre las variables *MAGEN_ERROR* y *MAXIMO_PRUEBAS,* dado que la variable *MAXIMO_PRUEBAS* marca el número de iteraciones que se deben realizar durante el reconocimiento, se modificó varias veces hasta conseguir una cantidad adecuada de iteraciones que dieran unos resultados aceptables.
3. Otra de las decisiones de implementación que se llevó a cabo fue con respecto al hecho de realizar el entrenamiento. Según especifica el requisito RNF-06, para realizar el entrenamiento se debe introducir el nombre y contraseña. En un primer lugar para cumplir con este requisito se pensó que el usuario sólo se registrara la primera vez que accediera a la aplicación y a partir de ese momento que pudiera realizar el entrenamiento cada vez que quisiera, sin tener que introducir los datos de nuevo. Pero dado que mediante esta forma el usuario no podría cambiar la contraseña a no ser que borrara los datos de la aplicación unido

a que el entrenamiento solo haría falta hacerlo la primera vez, se decidió por obligar al usuario a introducir un nombre y una contraseña cada vez que decidiera realizar el entrenamiento. De esta forma se consigue una mayor sostenibilidad de la aplicación.

4. Durante la implementación se comprobó que el proceso de entrenamiento que realiza numerosas tareas (entrenamiento de imágenes, volcado de imágenes, borrado de imágenes, generación de clave y el posterior cifrado de la contraseña…etc.) era demasiado pesado en términos de tiempo, y dado que se ejecutaba en el hilo principal de la aplicación ocasionaba que diera la impresión que la aplicación quedaba bloqueada. Se decidió por lo tanto realizar dicho procesamiento en una tarea que estuviera fuera del hilo principal mediante la creación de un hilo con la clase Thread, de esta forma durante el procesamiento la aplicación no daba la sensación de estar bloqueada. Para indicar al usuario que se está realizando el procesamiento correspondiente al entrenamiento se mostrará un mensaje que durará en pantalla el tiempo que dure el procesamiento.
5. A la hora de generar la clave se cambió durante varias veces los parámetros por los cuales se obtenían los valores asociados a las imágenes del usuario. Hasta no conseguir una clave que estuviera asociada al usuario y le identificase de forma única se probó con diferentes parámetros y valores.

### 5.1.2 Decisiones de implementación del módulo Central

Las decisiones que se realizaron durante la implementación del módulo central tienen que ver con la forma de distribuir los diferentes elementos que forman las "pantallas" que el usuario ve. Se optó por mostrar al usuario en la pantalla inicial debajo del casillero donde se activa o se desactiva el servicio de seguridad un mensaje que indicara si el servicio está habilitado/deshabilitado y en el caso de estar deshabilitado, se mostrará como habilitarlo. También se decidió por implementar debajo de cada opción del menú principal, la explicación detallada de en qué consiste cada submenú de cara a facilitar al usuario el manejo de la aplicación.

### 5.1.3 Decisiones de implementación para el módulo servicio de seguridad

Al implementar el servicio de seguridad, hubo que decidir cuándo se realizaba la comprobación de saber que aplicaciones están protegidas. En primer lugar se pensó realizar esa comprobación cuando se creara el servicio por primera vez, ir a la base de datos y obtener todas a las aplicaciones que habían sido seleccionadas por el usuario. El problema surgió cuando se comprobó que una vez iniciado el servicio si el usuario deseaba marcar nuevas aplicaciones que antes de iniciar el servicio no estaban marcadas, el servicio de seguridad no "protegía" esas aplicaciones que el usuario había marcado. Esto sucedía porque el servicio de seguridad sólo comprobaba que aplicaciones estaban seleccionadas cuando se iniciaba el servicio.

Dado que comprobar en la base de datos que aplicaciones están marcadas por el usuario en cada iteración del bucle que comprueba que aplicaciones están en ejecución suponía un coste muy alto en términos de computo, se optó por relanzar el servicio cada vez que se accediera a la pantalla de selección de aplicaciones a proteger. De esta forma, dado que el servicio de seguridad comprobaba las aplicaciones que están marcadas como aplicaciones a proteger cada vez que se inicia, se consiguió resolver el problema.

Durante la realización de las pruebas de aceptación se detectó que si el usuario elegía un número muy alto de aplicaciones a proteger, cuando se quería acceder a una de esas aplicaciones seleccionadas, el servicio de seguridad tardaba demasiado en lanzar el reconocimiento. Esto suponía que el usuario pudiera ver el contenido de la aplicación durante el tiempo que el servicio de seguridad tardaba en lanzar el reconocimiento. Este hecho sucedía porque el servicio de seguridad debía comprobar demasiadas aplicaciones. Después de realizar varias pruebas se optó por limitarla selección de aplicaciones de forma que el usuario sólo pudiera seleccionar 10 aplicaciones a proteger a la vez. Cabe destacar que si un atacante quisiera acceder a una de las aplicaciones protegidas sólo tendría que desinstalar la aplicación que se desarrolla en este proyecto. Por ello se tomó la decisión de configurar el servicio de seguridad para que en caso de estar activado lanzara el reconocimiento facial si se deseaba desinstalar la aplicación o detenerla.

# 5.2 Resultado de las pruebas de aceptación

A continuación se muestra los resultados de las pruebas de aceptación del sistema

| id | Elementos probados | Resultado |
|---|---|---|
| **PA-01** | RNF-05 | Superada |
| **PA-02** | RNF-05, RF-05 | Superada |
| **PA-03** | RNF-04, RNF-03, RF-09 | Superada |
| **PA-04** | RNF-08 | Superada |
| **PA-05** | RNF-07 | Superada |
| **PA-06** | RNF-10 | Superada |
| **PA-07** | RNF-11 | Superada |
| **PA-08** | RNF-09 | Superada |
| **PA-09** | RNF-07,08,09,10,11 | Superada |
| **PA-10** | RNF-12 | Superada |
| **PA-11** | RNF-13 | Superada |
| **PA-12** | RNF-01 | Superada |
| **PA-13** | RF-02,RF-03 | Superada |
| **PA-14** | RF-02,RF-03,RF-12 | Superada |
| **PA-15** | RF-08,RF-07 | Superada |
| **PA-16** | RF-08,RF-07,RF-01 | Superada |
| **PA-17** | RF-08,RF-07,RF-01 | Superada |
| **PA-18** | RF-08,RF-07,RF-01 | Superada |
| **PA-19** | RNF-17 | Superada |
| **PA-20** | RNF-18 | Superada |
| **PA-21** | RF-11 | Superada |

Tabla 16. Resultado pruebas de aceptación

# 5.3 Resultado de las pruebas de reconocimiento facial

En la siguiente sección se muestran los resultados obtenidos de las pruebas del reconocimiento facial que se especifican en el punto 3.8.2.

Según se ha definido en el punto 3.8.2 de este documento, a continuación se muestra mediante la tabla las personas que han participado en las pruebas del reconocimiento facial y sus rasgos característicos.

| Id Persona | Nombre | Sexo | Rasgos Faciales Destacables | Edad |
|---|---|---|---|---|
| **P1** | Alfredo | Masculino | Joven, castaño, sin barba | 24 años |
| **P2** | Ana | Femenino | Morena, ojos castaños | 52 años |
| **P3** | Javier | Masculino | Joven, moreno, con barba | 20 años |
| **P4** | Bea | Femenino | Joven, castaños. | 24 años |
| **P5** | Francisco | Masculino | Sin barba, con gafas | 55 años |
| **P6** | Ismael | Masculino | Poca barba, castaño | 27 años |
| **P7** | Jesús | Masculino | Sin barba, moreno | 27 años |

Tabla 17. Personas que realizan las pruebas

En la tabla 17 se muestran las personas acompañadas de sus rasgos faciales más característicos.

| Id Prueba | Persona que Realiza el Entrenamiento | Persona que realiza el Reconocimiento | Repeticiones | Positivos | Tasa de positivos |
|---|---|---|---|---|---|
| **RR-01** | P1 | P1 | 8 | 5 | **62%** |
| **RR-02** | P1 | P4 | 8 | 0 | **0%** |
| **RR-03** | P1 | P2 | 8 | 0 | **0%** |

| | | | | | |
|---|---|---|---|---|---|
| **RR-04** | P1 | P3 | 8 | 1 | **12%** |
| **RR-05** | P1 | P5 | 8 | 0 | **0%** |
| **RR-06** | P6 | P6 | 8 | 4 | **50%** |
| **RR-07** | P7 | P7 | 8 | 5 | **62%** |

Tabla 18. Resultado pruebas de reconocimiento Facial

La tabla 18 muestra los resultados de las pruebas del reconocimiento facial. Como se ha comentado anteriormente en el punto 3.8.2, hay que diferenciar la persona que realiza el entrenamiento de la persona que realiza el reconocimiento.

Unos resultados perfectos serían la obtención de una tasa de positivos de 100% cuando la persona que realiza el entrenamiento es la misma que la persona que realiza el reconocimiento. Esto significaría que el usuario de la aplicación siempre es reconocido por la aplicación.

Cuando la persona que realiza el entrenamiento es diferente a la persona que realiza el reconocimiento, los resultados deseables serían los que tuvieran una tasa de positivos del 0%, ya que significaría que no se producen falsos positivos, es decir, la aplicación no reconocería nunca la cara de a una persona que no es la usuaria de la aplicación.

Según los resultados obtenidos, el reconocimiento facial implementado funciona mucho mejor en entornos con iluminación artificial alta. En entornos con baja iluminación ya sea artificial o natural, el reconocimiento facial funciona peor. Cabe destacar que la implementación se ha realizado de tal forma que se da prioridad a la obtención de tasas de positivos de 0% cuando la persona que realiza el reconocimiento es diferente a la persona que realiza el entrenamiento, es decir, se ha intentado que el reconocimiento facial no tenga falsos positivos. Por otro lado es importante destacar la importancia que tiene la realización del entrenamiento en un entorno bien iluminado y prestando atención a la correcta posición de la cara frente al objetivo de la cámara, por lo tanto cuanto mejor se realice el entrenamiento mejor funcionará el reconocimiento.

# CAPÍTULO 6

## 6. Conclusiones y Líneas Futuras

# 6.1 Conclusiones sobre el proyecto

Durante esta sección se mostrarán las conclusiones obtenidas una vez terminado el proyecto final de carrera, analizando los principales problemas encontrados durante el desarrollo del mismo junto con el resultado final.

## 6.1.1 Resultados obtenidos

Se ha construido una aplicación para el sistema operativo Android, cuyo principal objetivo es el de realizar una autenticación continua basada en el reconocimiento facial cada vez que se accede a unas aplicaciones establecidas. El usuario que utilice la aplicación podrá elegir qué aplicación desea proteger. Sin utilizar esta aplicación, si el usuario deseaba proteger el acceso a alguna aplicación debía restringir el acceso al dispositivo entero. Sin embargo, gracias al desarrollo realizado se evita tener que introducir un patrón o contraseña cada vez que se quisiera acceder a una determinada. Por tanto, se ha obtenido una gran ventaja para la protección de acceso a las aplicaciones tanto en términos de usabilidad como teniendo muy presente la seguridad.

## 6.1.1 Dificultades del proyecto

Prácticamente la mayor dificultad del proyecto vino dado a la hora de implementar el módulo de identificación y entrenamiento. Los primero problemas surgieron a la hora de integrar las librerías de openCV con la aplicación. Dado que había que instalar diferentes librerías, se tardó un tiempo en empezar a utilizarlas.

Otro problema surgió a la hora implementar la detección de rostros al no saber que tecnología era la más apropiada, en un primer momento se optó por realizar la detención mediante los métodos que ofrecen el API de Android, pero dado que la librería de openCV dispone de una batería de funciones mejor implementada, se cambió la forma de detectar los rostros y se optó por hacerlo mediante openCV. La principal dificultad vino por conseguir que el reconocimiento facial tuviera una tasa de aciertos mayor del 50%, para ello se modificó el tratamiento de las imágenes de usuario y de las imágenes que sirven como almacén para realizar el entrenamiento.

A la hora de generar la clave también surgieron diferentes problemas, dado que no se sabía qué tipo de parámetros, funciones o mecanismos matemáticos había que utilizar para obtener una clave que fuera única y se pudiera asociar a la cara del usuario. Además, el hecho de tener que generar una clave que sea identificativa para

cada cara ya supone un gran reto. Es complicado generar patrones identificativos a partir de los rasgos faciales de una persona.

Otro de los problemas surgidos fue con respecto a las actualizaciones del sistema operativo Android. Dado que la aplicación empezó a implementarse con la versión 4.2 de Android, muchas de las funciones utilizadas para el desarrollo de la aplicación quedaron anticuadas con el paso de la versión de 4.2 a 5.0, esto supuso la corrección de algunas funcionalidades para poder adaptarlas a la versión 5.0, por lo tanto se produjo un aumento del tiempo que estaba previsto para la implementación.

## 6.2 Líneas Futuras

En las siguientes líneas se explican las posibles mejoras sobre la aplicación que se puedan realizar en un futuro.

1. **Mejoras en el reconocimiento facial**: Siempre cabe la posibilidad de mejorar el reconocimiento facial de la aplicación. Es posible mejorar el tiempo que tarda en reconocer a personas y mejorar la calidad del reconocimiento.
2. **Posibilidad de realizar el reconocimiento con la pantalla en vertical**. Actualmente la aplicación realiza el reconocimiento facial con la cámara frontal del dispositivo en modo horizontal, como mejora se podría realizar dicho reconocimiento con la pantalla del dispositivo en posición vertical.
3. **Incluir la posibilidad de elegir distintos tipos de autenticación**. Otra posible mejora de la aplicación sería la de ofrecer distintos tipos de autenticación a la hora de proteger aplicaciones. Como por ejemplo que el usuario pueda elegir entre autenticación facial o autenticación por huella dactilar.
4. **Elegir entre el nivel de seguridad**: Se podría añadir la posibilidad de elegir entre el nivel de seguridad que debe tener la autenticación.

# Referencias

[1] Artículo de noticias tecnológicas. Disponible:
*HTTP://WWW.EUROPAPRESS.ES/ECONOMIA/NOTICIA-ECONOMIA-TELECOS-SUSCRIPCIONES-MOVILES SUPERARAN-POBLACION-MUNDIAL-2015-ERICSSON-20140604142937.HTML.*

[2] Artículo de Noticias tecnológicas. Disponible:
*HTTP://WWW.XATAKANDROID.COM/SISTEMA-OPERATIVO/ANDROID-CONSOLIDO-SU-LIDERAZGO-EN-2014-ALCANZANDO-EL-81-5-DE-CUOTA-DE-MERCADO*

[3] Artículo de Noticias tecnológicas. Disponible:
*HTTP://WWW.ELANDROIDELIBRE.COM/2015/01/GOOGLE-PLAY-SUPERA-LA-APPSTORE-EN-CANTIDAD-DE-APLICACIONES-Y-DESARROLLADORES.HTML*

[4] Tipos de Autenticación. Disponible:
*WIKIPEDIA. HTTP://ES.WIKIPEDIA.ORG/WIKI/AUTENTICACI%C3%B3N.*

[5] Definición de biometría. Disponible:
*HTTP://WWW.FACEPHI.COM/ES/CONTENT/TECNOLOGIA/*

[6] Aplicación de redes sociales. Disponible:
*HTTPS://ES-ES.FACEBOOK.COM/*

[7] Facelock Pro. Disponible:

*HTTPS://PLAY.GOOGLE.COM/STORE/APPS/DETAILS?ID=COM.FACELOCK4APPSPRO&HL=ES_49.*

[8] AppLock by Face. Disponible:
*HTTPS://PLAY.GOOGLE.COM/STORE/APPS/DETAILS?ID=COM.VINISOFT.OPENCV.APPLOCK.FACELOCK*

[9] Perfect App Lock Pro. Disponible:
https://play.google.com/store/apps/details?id=com.morrison.applocklite

[10]Facelock for Screen. Disponible:
*HTTPS://PLAY.GOOGLE.COM/STORE/APPS/DETAILS?ID=COM.VE.FACELOCK.FREE*

[11] Lenguaje de programación java. Disponible:
*DEVELOPER.ANDROID.COM*

[12] Lenguaje de programación VisualBasic multiplataforma. Disponible:
*HTTP://XAMARIN.COM/VISUAL-STUDIO?_BT=52153138028&_BK=VISUAL%20BASIC%20ANDROID&_BM=E&GCLID=COFWI47YXSCCFRI6GWODFAMGTW.*

[13]Lenguajes de programación intuitivos. Disponible:
*HTTP://WWW.APPINVENTOR.ORG/*

[14] Opencv:
GARY BRADSKI AND ADRIAN KAEHLER LEARNING OPENCV: COMPUTER VISION WITH THE OPENCV LIBRARY

[15] API de Android.
HTTP://DEVELOPER.ANDROID.COM/REFERENCE/ANDROID/MEDIA/FACEDETECTOR.FACE.HTML

[16] Moodstocks. Disponible:
HTTPS://MOODSTOCKS.COM/

[17] Qualcomm. Disponible:
HTTPS://WWW.QUALCOMM.COM/PRODUCTS/SNAPDRAGON

[18] CouchDB .Página web oficial. Disponible:
HTTP://COUCHDB.APACHE.ORG/

[19] FireBird. Disponible:
HTTP://WWW.FIREBIRDSQL.ORG/

[20] Mysql:
ÁNGEL COBO,PATRICIA GÓMEZ, DANIEL PEÉREZ, ROCIO RONCHAPHP Y MYSQL: TECNOLOGÍA PARA EL DESARROLLO DE APLICACIONES WEB.

[21] SQLite. Disponible:
HTTPS://WWW.SQLITE.ORG/

[22]Cifrado AES 256 bits. Disponible:
HTTP://CSRC.NIST.GOV/PUBLICATIONS/FIPS/FIPS197/FIPS-197.PDF

[23] Momentos Invariantes de HU. DISPONIBLE:
HTTPS://PREZI.COM/XVT49LIFYORT/MOMENTOS-INVARIANTES-DE-HU/

[24] Función hash (). Disponible:
HTTP://WWW.REDESZONE.NET/2010/11/09/CRIPTOGRAFIA-ALGORITMOS-DE-AUTENTICACION-HASH/

[25] Damien Katz. Disponible:
HTTPS://WWW.LINKEDIN.COM/IN/DAMIENKATZ

[26] Ataque por fuerza bruta aes 256 bits.
HTTPS://WWW.BOXCRYPTOR.COM/ES/CIFRADO

[27] Frame. Disponible:
HTTP://WWW.MASTERMAGAZINE.INFO/TERMINO/5072.PHP

[28] Eclipse IDE. Disponible:
HTTPS://DEVELOPER.ANDROID.COM/SDK/INDEX.HTML

[29] Altova Umodel 2015. Disponible:

HTTP://WWW.ALTOVA.COM/SIMPLEDOWNLOAD2C.HTML?GCLID=COU06ECNYCCCFSKYWWODCKKG8G

[30] Windows 8. Disponible:
HTTP://WINDOWS.MICROSOFT.COM/ES-ES/WINDOWS/DOWNLOADS

[31] Notepad ++. Disponible:
HTTPS://NOTEPAD-PLUS-PLUS.ORG/

[32] Microsoft office 2013. Disponible:
HTTP://WWW.MICROSOFTSTORE.COM/STORE/MSEEA/ES_ES/CAT/OFFICE/CATEGORYID.66226700

[33] Android 5. Disponible:
HTTPS://WWW.ANDROID.COM/VERSIONS/LOLLIPOP-5-0/

[34] Google Chrome. Disponible:
HTTPS://WWW.GOOGLE.ES/CHROME/BROWSER/DESKTOP/INDEX.HTML.

[35] Instituto Nacional de Ciberseguridad.
HTTPS://WWW.INCIBE.ES/

# ACRÓNIMOS

| | |
|---|---|
| BD | *Base de datos* |
| API | *Application Programming Interface* |
| GNU GPL | *General Public License* |
| APP | *Application* |
| XML | *Extensible Markup Language* |
| AES | *Advanced Encryption Standard* |
| RSA | *Rivest, Shamir y Adleman* |

# ANEXO 1: GESTIÓN DEL PROYECTO

El siguiente anexo se detalla los diferentes apartados referentes a la gestión del proyecto, en los cuales se incluye el análisis económico del proyecto, los medios técnicos utilizados y la planificación del mismo.

# 1. PLANIFICACIÓN DEL TRABAJO

A continuación se detallan la planificación inicial y el posterior desarrollo real del proyecto.

## 1.1 Planificación Inicial

En la siguiente sección se muestra la planificación inicial del proyecto en la cual se puede observar las distintas tareas en las que se ha descompuesto el proyecto. A su vez también se muestra la estimación de trabajo asociada a cada tarea. Cabe destacar que en la elaboración de la planificación se han tenido en cuenta jornadas laborables de 3 horas diarias. La planificación del proyecto comprende desde el día 6 de Octubre del 2014, al día 14 de mayo de 2015, que hacen un total de 159 días laborables.

A continuación mediante la tabla 19 se muestra cada una de las tareas en las que se divide el proyecto y el tiempo estimado para cada una de ellas.

| Proyecto fin de carrera | 159 días | Fecha inicio 06/10/2014 | Fecha fin 20/05/2015 |
|---|---|---|---|
| **Planificación Inicial** | **3 días** | **06/10/2014** | **08/10/2014** |
| **Estudio del estado del arte** | **7 días** | **09/10/2014** | **17/10/2014** |
| **Análisis** | **16 días** | **20/10/2014** | **10/11/2014** |
| Arquitectura del sistema | 5 días | 20/10/2014 | 24/10/2014 |
| Estudio Tecnológico | 3 días | 27/10/2014 | 29/10/2014 |
| Definición de casos de uso | 4 días | 30/10/2014 | 04/11/2014 |
| Definición de requisitos | 4 días | 05/11/2014 | 10/11/2014 |
| **Diseño** | **24 días** | **11/11/2014** | **12/12/2014** |
| Diseño del software | 10 días | 11/11/2014 | 24/11/2014 |
| Diagramas de secuencia | 14 días | 25/11/2014 | 12/12/2014 |
| **Implementación** | **64 días** | **15/12/2014** | **12/03/2015** |
| **Pruebas** | **10 días** | **13/03/2015** | **26/03/2015** |
| **Documentación** | **35 días** | **27/03/2015** | **14/05/2015** |

Tabla 19. Planificación Inicial

Como se puede observar en la Tabla 19, aproximadamente el 60% del tiempo planificado se distribuye entre la fase de implementación y la fase de documentación, ya que son las dos tareas que más esfuerzo y tiempo requieren.

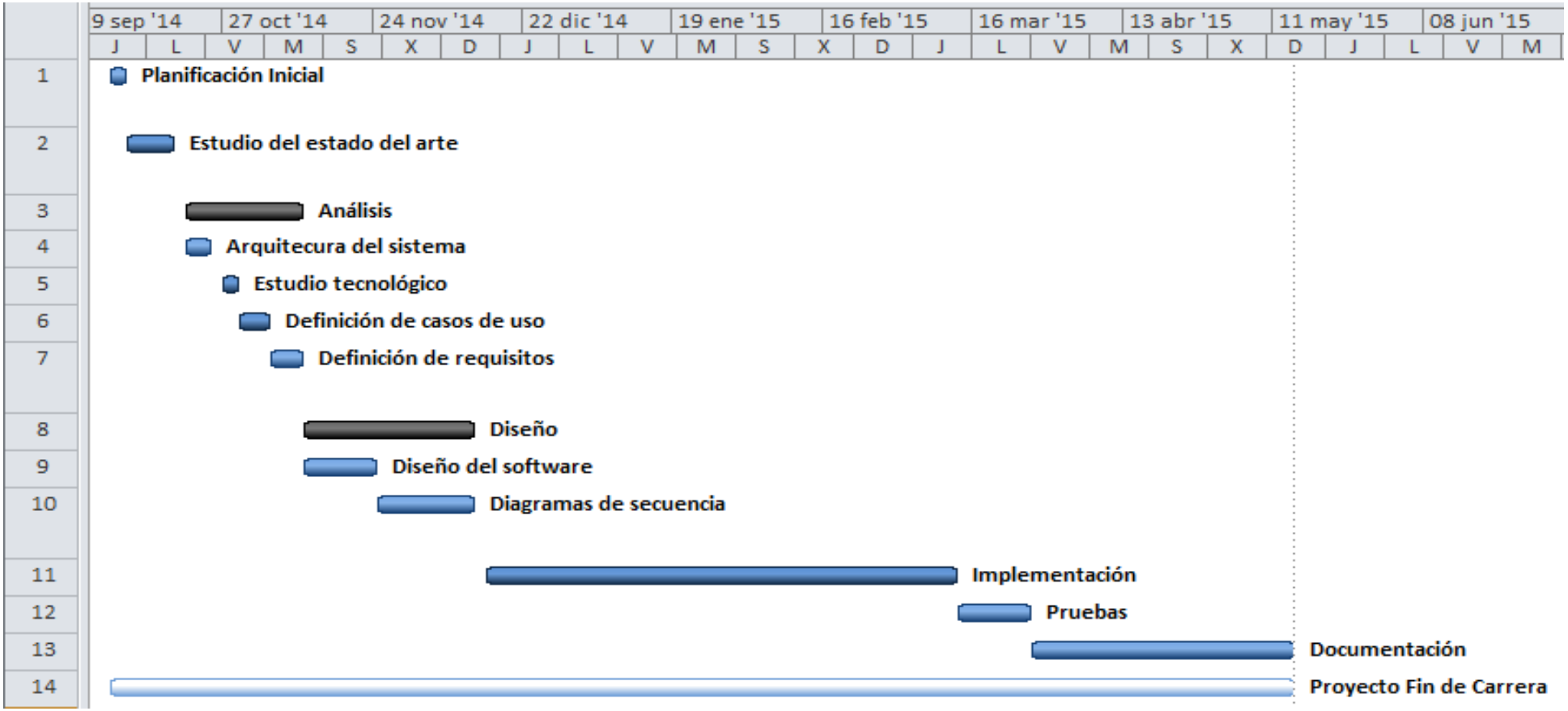



Figura 22. Planificación Inicial del proyecto

## 1.2 Desarrollo real del proyecto

En esta sección se muestra el desarrollo final del proyecto. El desarrollo real del proyecto varía con la planificación inicial del mismo. Esta variación es debida al aumento de tiempo en la realización de algunas de las tareas planificadas, durante la tarea de implementación surgieron problemas que alargaron el tiempo planificado inicialmente. Mediante la planificación inicial se estimaba que el proyecto tuviera la fecha fin del 14 de mayo de 2015, con un total de 159 días laborales. La duración final del proyecto ha sido de 250 días que supone una variación de 91 días con respecto a la planificación inicial.

Mediante la siguiente tabla se puede observar la duración real de cada tarea y la variación en días de cada tarea con respecto a la planificación inicial.

| Proyecto fin de carrera | 250 días | Fecha inicio 06/10/2014 | Fecha fin 18/09/2015 | Variación 91 días (57,2%) |
|---|---|---|---|---|
| **Planificación Inicial** | **3 días** | **06/10/2014** | **08/10/2014** | **0 días** |
| **Estudio del estado del arte** | **7 días** | **09/10/2014** | **17/10/2014** | **0 días** |
| **Análisis** | **16 días** | **20/10/2014** | **10/11/2014** | **0 días** |
| Arquitectura del sistema | 5 días | 20/10/2014 | 24/10/2014 | |
| Estudio Tecnológico | 3 días | 27/10/2014 | 29/10/2014 | |
| Definición de casos de uso | 4 días | 30/10/2014 | 04/11/2014 | |
| Definición de requisitos | 4 días | 05/11/2014 | 10/11/2014 | |
| **Diseño** | **34 días** | **11/11/2014** | **26/12/2014** | **10 días (41%)** |
| Diseño del software | 10 días | 11/11/2014 | 24/11/2014 | 0 días |
| Diagramas de secuencia | 24 días | 25/11/2014 | 26/12/2014 | 10 días (71%) |
| **Implementación** | **104 días** | **30/12/2014** | **22/05/2015** | **40 días (62%)** |
| **Pruebas** | **20 días** | **25/05/2015** | **19/06/2015** | **10 días (50%)** |
| **Documentación** | **65 días** | **06/06/2015** | **18/09/2015** | **30 días (46%)** |

Tabla 20. Desarrollo real del proyecto

En total se observa una desviación de 91 días con respecto a la planificación inicial. Como se puede observar la mayor desviación se encuentra en las tareas de implementación y documentación. Esto es debido a que a la hora de planificar los días de trabajo para la fase de implementación no se tenía experiencia sobre la implementación de una aplicación en Android. Como se puede observar viendo la columna de variación con respecto a la planificación inicial, esta tarea tiene una desviación del 62%. En la tarea de documentación la desviación del 50% es debido en la mayoría de los casos, a las correcciones de la documentación por parte de la tutora que no se tuvieron en cuenta durante la planificación inicial. En la tarea de pruebas se observa una desviación de 10 días, esto es debido principalmente a la poca experiencia en planificación de pruebas y por lo tanto a la necesidad de incrementar el número de pruebas previstas en la planificación inicial.

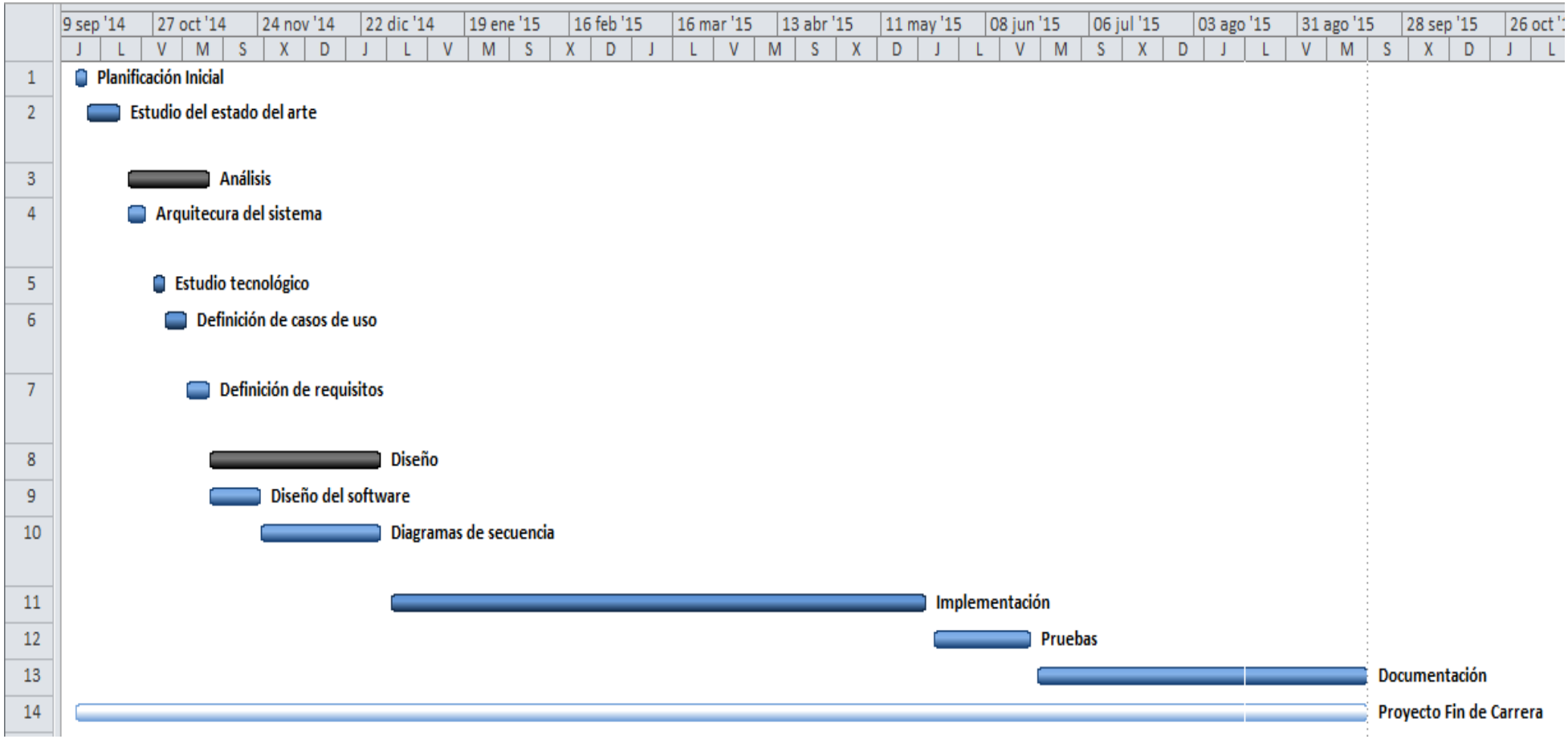



Figura 23. Desarrollo real del Proyecto

# 2. Medios Técnicos empleados para el proyecto

En la siguiente sección se detallan los medios técnicos que se han utilizado para el desarrollo del proyecto. La siguiente tabla muestra los componentes técnicos utilizados junto con una breve descripción.

| Herramienta | Descripción |
|---|---|
| **Eclipse IDE [28]** | Entorno de programación para la implementación de la aplicación |
| **Altova Umodel 2015 [29]** | Herramienta de modelado UML para la creación de los diferentes diagramas y figuras |
| **Windows 8 [30]** | Sistema operativo sobre el que se ha desarrollado el proyecto. |
| **Notepad ++ [31]** | Editor de textos utilizado para la visualización de determinados archivos de código fuente |
| **SQLite [21]** | Motor de base de datos utilizado para el desarrollo de la base de datos de la aplicación |
| **Microsoft office 2013 [32]** | Conjunto de programas para la realización de la documentación del proyecto |
| **Android 5 (Lollipop) [33]** | Sistema operativo de dispositivos móviles sobre el cual se ha realizado la aplicación desarrollada |
| **Google Chrome [34]** | Navegador web utilizado durante el desarrollo del proyecto |

Tabla 21. Herramientas utilizadas en el proyecto

En la siguiente tabla se detalla el equipo utilizado para el desarrollo del proyecto

| Equipo | Descripción |
|---|---|
| **Ordenador de sobremesa :**<br>-Intel Core i7 4GHZ, 16GB RAM | Ordenador sobre el cual se ha realizado todo el Proyecto final. |
| **Dispositivos móviles:**<br>**-**LG L9 (Android 4.2.2)<br>-Samsung Galaxy Alpha (Android 5.0.2) | Dispositivos móviles utilizados para realizar el desarrollo de la aplicación |

Tabla 22. Equipos utilizados en el proyecto

# 3. Análisis Económico del Proyecto

En los siguientes apartados se muestra el análisis económico del proyecto que incluye los diferentes costes que componen el presupuesto final.

## 3.1 Presupuesto

En esta sección se muestra el presupuesto en el cual se detallarán los costes presupuestados para todo el proyecto.

Cabe destacar que los costes que se detallan a continuación vienen reflejados sin el 21% de IVA, exceptuando el coste final del proyecto, en el que se ha añadido el IVA para su cálculo.

### *3.1.1 Gastos de Personal*

Se tiene en cuenta la dedicación de un único ingeniero informático a lo largo de todo el proyecto. La dedicación en horas del ingeniero es de 3 horas diarias en días laborables durante los 159 días de la duración estimada en la planificación inicial.

Se considera un coste de 20 euros brutos la hora trabajada a lo que se ha añadido el importe a pagar a la Seguridad Social, 28,3% para el régimen general de la Seguridad Social.

| Recurso | Salario/Hora | Estimación días | Horas/Día | Horas totales | Coste Bruto | Seg.Social | Coste total |
|---|---|---|---|---|---|---|---|
| **Ingeniero** | 20 € | 159 | 3 horas/días | 477 | 9540 € | 2699,82 € | **12239,82 €** |

Tabla 23. Gastos de Personal

### *3.1.2 Gastos de Equipos*

En la siguiente sección se detallan los gastos relacionados con las herramientas utilizadas durante el desarrollo del proyecto. La tabla 24 muestra los equipos utilizados, su precio de venta al público con el IVA incluido y el periodo de depreciación del mismo. Los equipos utilizados son un ordenador de sobremesa que se empleó durante todo el proyecto y dos dispositivos móviles que se utilizó cada uno durante la mitad del proyecto aproximadamente.

| Descripción | Coste | Dedicación | Periodo de depreciación | Coste imputable |
|---|---|---|---|---|
| Ordenador : **-Intel Core i7 4GHZ, 16GB RAM** | 1253,0 € | 5,3 meses | 36 meses | 184,46 € |
| Teléfono: **-LG L9 (Android 4.2.2)** | 180,0 € | 2,65 meses | 24 meses | 19,87 € |
| Teléfono: **-Samsung Galaxy Alpha** | 340,00 € | 2,65 meses | 24 meses | 37,54 € |
| | | **Coste Total:** | | **241,87 €** |

Tabla 24. Gastos de Equipos

### *3.1.3 Gastos de Software*

En la siguiente sección se detallan los gastos derivados de las licencias del software utilizado durante la realización del proyecto. Al igual que con los gastos de equipo, el precio se muestra con el IVA incluido. Sólo se muestra el software mencionado en la sección 2 de este anexo que no sea gratuito.

| Descripción | Coste | Dedicación | Periodo de depreciación | Coste imputable |
|---|---|---|---|---|
| **Altova Umodel 2015** | 117,0 € | 5,3 meses | 36 meses | 17,22 € |
| **Microsoft Windows 8** | 159,00 € | 5,3 meses | 36 meses | 23,40 € |
| **Microsoft office 2013** | 121,2 € | 5,3 meses | 36 meses | 17,84 € |
| | | **Coste Total:** | | **58,46 €** |

Tabla 25. Gastos de Software

### *3.1.4 Otros gastos*

En esta sección se muestran los gastos asociados a desplazamientos y el gasto derivado del consumo de internet.

| Descripción | Cantidad | Coste unitario | Coste final |
|---|---|---|---|
| **Desplazamientos** | 5 desplazamientos | 10 € | 50 € |
| **Internet** | 5,3 meses | 38,0 € | 201,0 € |
| | **Coste total :** | | **251,0 €** |

### *3.1.5 Coste Total del Proyecto*

A continuación se detallarán los costes totales del proyecto. El coste final del proyecto es la suma todos los gastos vistos anteriormente más los porcentajes de riesgos añadidos para hacer frente a imprevistos. Se ha definido un 15 % de porcentaje de riesgo y un 20% de porcentaje de beneficio.

| Concepto | Coste |
|---|---|
| **Gasto del personal** | 12239,82 € |
| **Gasto de equipos** | 241,87 € |
| **Gasto del software** | 58,46 € |
| **Otros gastos** | 251,0 € |
| **Total gastos sin riesgo** | **12791,15 €** |
| **Riesgo (15%)** | 1279,11€ |
| **Total Gastos sin beneficios** | **14070,26 €** |
| **Beneficios (20%)** | 2814,05 € |
| **Total Gastos sin IVA** | **16884,31 €** |
| **IVA (21%)** | 3545,70 |
| **Total** | **20430,01 €** |

Tabla 26. Coste total Proyecto

# ANEXO 2: MANUAL DE USUARIO

# 1. Introducción

CADroid es una aplicación para dispositivos con sistema operativo Android, cuya función es la de proporcionar seguridad de manera continua a las demás aplicaciones instaladas en el dispositivo mediante la autenticación de rasgos faciales. CADroid permite al usuario elegir entre las aplicaciones que tiene instaladas en su dispositivo a cuál de ellas desea proporcionar autenticación continua de rasgos faciales.

Mediante este manual se pretende que sirva de referencia a los usuarios de CADroid con el objetivo de facilitar su uso y entendimiento.

# 2. Requisitos previos e instalación

CADroid ha sido desarrollada y probada en un dispositivo móvil de marca Samsung con Sistema Operativo Android 5.0, por lo que se recomienda para el correcto funcionamiento de la misma disponer de un dispositivo móvil con sistema Operativo Android 5. 0 Lollipop o superior.

Para la correcta instalación de la aplicación no hace falta ningún tipo de acción adicional por parte del usuario. No se garantiza el correcto funcionamiento de la aplicación para versiones anteriores a la versión del sistema Operativo Android 5.0.

Se detallan a continuación las características mínimas que debe tener el dispositivo móvil:

- Sistema operativo Android. Versión 5.0 Lollipop.
- Cámara fotográfica frontal.
- Memoria asignada: 100 MB.

# 3. Ejecución y funcionamiento

Al pulsar el icono de CADroid se mostrará un menú como el que aparece en la Figura 24. Como se puede observar, la opción de activar el servicio de seguridad esta inhabilitada, esto ocurre porque no se ha realizado aún el entrenamiento. El primer paso es realizar el entrenamiento.


Figura 24. Menú principal de CADroid

## 3.1 Realizar entrenamiento

Realizar un correcto entrenamiento es fundamental para que el reconocimiento facial reconozca al usuario la mayor parte de las veces. Es importante tener en cuenta que el entrenamiento ha de realizarse en un entorno con alta iluminación y controlando la postura de la cara frente al objetivo de la cámara del dispositivo.

Cuando se pulse el apartado de Entrenamiento, aparecerá una pantalla como la aparece en la Figura 25. Como se puede observar existen tres campos en los cuales el usuario deberá introducir su nombre sin dígitos y la contraseña alfanumérica dos veces. Si se ha introducido correctamente las contraseñas y el nombre de usuario, el siguiente paso es pulsar el botón de aceptar.


Figura 25. Pantalla de Entrenamiento de CADroid

La Figura 26, muestra la pantalla de la aplicación después de pulsar el botón de aceptar del menú de entrenamiento (Figura 25). El usuario deberá pulsar en el botón de grabar hasta que el número de fotos almacenadas llegue a 15.

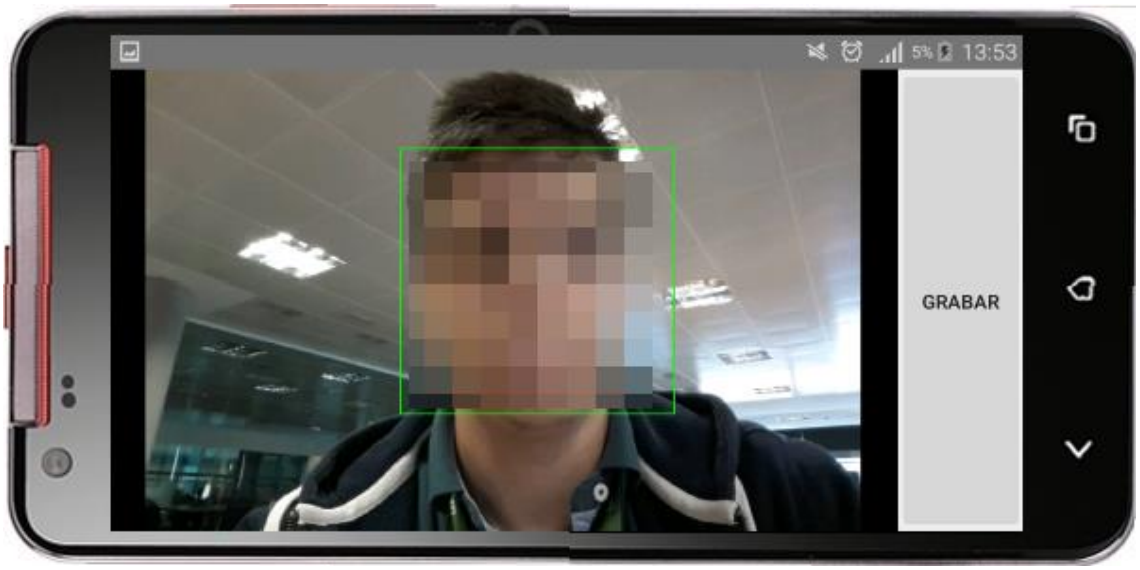



Figura 26. Pantalla de realización de entrenamiento

Cuando se llegue a 15 fotos almacenadas, el botón de grabar cambiará y se mostrará la opción de entrenar. Cuando se pulse dicho botón, CADroid realizará el proceso de entrenamiento mediante las imágenes que el usuario ha introducido. Este proceso puede tardar varios segundos, que se notificará al usuario según aparece en la Figura 27.

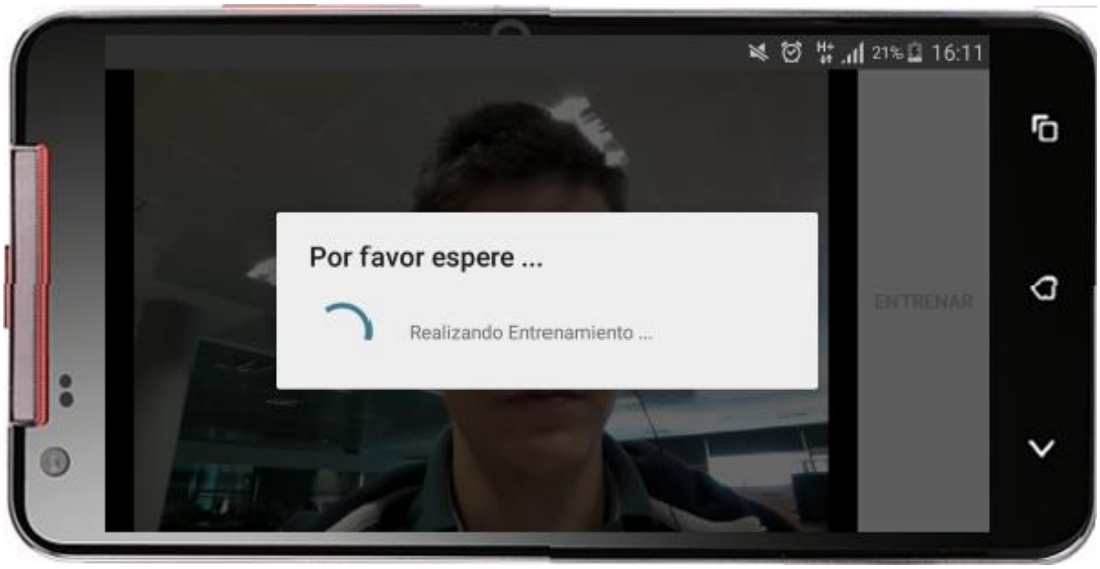



Figura 27. Pantalla de entrenamiento realizado

Cuando CADroid haya terminado de realizar el entrenamiento mostrará la pantalla del menú principal pero ahora con la opción de activar seguridad habilitada.

## 3.2 Elegir Aplicaciones a proteger

Una vez realizado el entrenamiento, ya se puede activar el servicio de seguridad, pero antes debemos seleccionar la aplicación o aplicaciones que deseamos proteger. Para seleccionar las aplicaciones a proteger, el usuario deberá pulsar en la sección “Seleccionar aplicaciones”. Una vez pulsado aparecerá una pantalla como la que aparece en la Figura 28. El usuario podrá elegir las aplicaciones que desea otorgar seguridad solo con pulsar sobre la casilla que aparece al lado del icono de la aplicación.

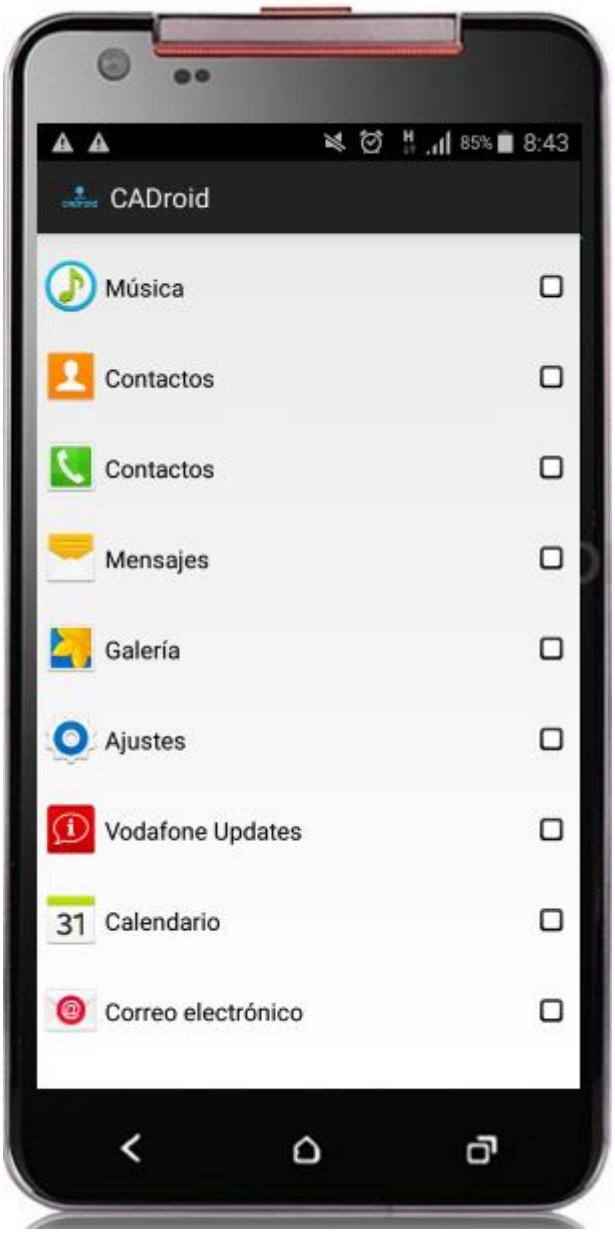



Figura 28. Pantalla de selección de aplicaciones

## 3.3 Realizar Reconocimiento

Si ya se ha realizado el entrenamiento y el usuario ha elegido las aplicaciones que desea proteger, ya puede activar el servicio de seguridad. Para ello solo tiene que pulsar sobre la casilla que indica “Activar servicio de seguridad”. Al pulsar sobre dicha casilla aparecerá una notificación de Android indicando que hay un servicio en segundo plano. El usuario a partir de ahora cada vez que quiera acceder a una aplicación de las seleccionadas a proteger le saltará una pantalla en negro, mientras, CADroid realizará el reconocimiento facial. Si CADroid reconoce al usuario, se mostrará la aplicación que estaba intentando acceder, en caso contrario se mostrará una pantalla como la que se muestra mediante la Figura 29. Para acceder a la aplicación el usuario deberá introducir la contraseña que introdujo durante el entrenamiento y pulsar el botón de aceptar.


Figura 29. Pantalla en caso de fallo de reconocimiento