CredPrint: Sistema para agilizar la recepción en eventos e instalaciones

Memoria TFG

Trabajo Fin de Grado

Grado en Ingeniería Informática

VNiVERSiDAD D SALAMANCA

Julio 2021

Autor:

Alejandro González González

Tutores:

María Belén Curto Diego

María José Polo Martín

Guillermo Ménguez Álvarez

Pablo Rodríguez

CERTIFICADO DE LOS TUTORES

María Belén Curto Diego, María José Polo Martín personal investigador de la Universidad de Salamanca y Guillermo Menguez Álvarez,Pablo Rodríguez personal del Observatorio Tecnológico HP HACEN CONSTAR: Que el trabajo titulado “CredPrint: Sistema para agilizar la recepción en eventos e instalaciones” ha sido realizado por Alejandro González González, con DNI 72153316Q y constituye la memoria del trabajo realizado para la superación de la asignatura Trabajo de Fin de Grado de la titulación Grado en Ingeniería Informática de esta Universidad.

Y para que así conste a todos los efectos oportunos.
En Salamanca, a 5 de julio de 2021

María Belén Curto Diego

CURTO DIEGO MARIA BELEN - 07855759V
Firmado digitalmente por CURTO DIEGO MARIA BELEN - 07855759V
Fecha: 2021.07.06 18:00:33 +02'00'

María José Polo Martín

POLO MARTIN MARIA JOSE - 07847347T
Firmado digitalmente por POLO MARTIN MARIA JOSE - 07847347T
Fecha: 2021.07.06 15:41:28 +02'00'

Guillermo Ménguez Álvarez

Guillermo Ménguez Álvarez
Digitally signed by Guillermo Ménguez Álvarez
Date: 2021.07.05 15:11:12 +02'00'

Pablo Rodríguez

Resumen

Esta memoria presenta una aplicación multiplataforma orientada a dispositivos móviles para agilizar la recepción de asistentes en eventos e instalaciones.

La aplicación permite a un usuario, que llegue a un evento o una instalación, rellenar un formulario con sus datos, escanear su documentación y enviarla por correo a un encargado de verificar e imprimir las credenciales para permitir el acceso a las instalaciones. Esto permitirá agilizar el proceso de ingreso a eventos o el acceso a instalaciones que requieran un registro para poder ingresar en ellas.

Para la realización de este proyecto se ha usado Android Studio que es un IDE para la creación de aplicaciones para sistemas operativos Android. Este IDE requiere conocimientos del lenguaje de programación Java y el lenguaje de marcas XML.

Además, se ha usado Workpath plataform que es una versión customizada de Android que dispone de un SDK con APIs propias.

Este proyecto se enmarca y se desarrolla dentro del Observatorio Tecnológico HP, iniciativa de colaboración entre HP y la Universidad de Salamanca

Abstract

This document presents a multi-platform app, geared to mobile devices, to speed up reception at events and facilities.

This app allows a user, who arrives at an event or a facilitiy, to fill out a form with their data, scan their documentation and send it by mail to someone in charge of verifying and printing the credentials to allow acces to the facilities. This speed up the process to join an event or facility that requires a documentation to get in it.

To carry out this Project, I used the Android Studio, which is an IDE for creating applications for Android operating systems. This requires use Java programming language and the extensible markup language (XML).

Workpath has also been used, which is a custom version of Android that has an SDK with APIS.

This Project is framed and developed within the HP technological Observatory a collaboration inititive between HP and the University of Salamanca.

Tabla de contenido

Lista de figuras

1. Introducción

La llegada a un evento o a una empresa con medidas de control de acceso siempre requiere hacer largas colas hasta que las personas encargadas de la recepción nos atienden, toman nuestros datos y nos imprimen un pase.

El objetivo de este Trabajo Fin de Grado (TFG) es desarrollar una aplicación que permita agilizar este proceso utilizando las impresoras multifunción de HP, dado que este proyecto se enmarca y se desarrolla dentro del Observatorio Tecnológico HP, iniciativa de colaboración entre HP y la Universidad de Salamanca

El usuario podría acercarse a una impresora, escanear su DNI y rellenar el formulario con sus datos. Esta información se le haría llegar a la persona de recepción encargada de validar la información, quien dispondría de toda la información en el formato adecuado para imprimir las credenciales que acompañan a la tarjeta de acceso a las instalaciones.

Para la realización de este proyecto se utilizará una de las impresoras multifunción de HP que llevan instalado un sistema Android customizado llamado Workpath. Workpath Platform es un sistema operativo basado en Android que se ejecuta en ciertas impresoras multifunción (MultiFunction Printer – MFP)

La plataforma HP Workpath es un componente central de un ecosistema constituido por impresoras, aplicaciones y herramientas de gestión basadas en la nube, diseñado para ayudar a los intermediarios a diferenciar su negocio y aportar valor a sus clientes a través de la integración de aplicaciones. Al conectar las impresoras multifunción HP a repositorios o soluciones de software de terceros en la nube, las aplicaciones Workpath ofrecen oportunidades comerciales en el gran mercado de la integración de contenido en papel a contenido digital, así como nuevas soluciones creativas que utilizan las capacidades únicas de las impresoras HP.

Las aplicaciones de Workpath ayudan a las empresas y a las organizaciones a acelerar la digitalización pues facilitan la conversión del papel en contenido digital. Mejorando el acceso a los contenidos y simplificando el procesamiento de los documentos también permite que los trabajadores de hoy en día tengan mayor movilidad. La racionalización de los flujos de trabajo también reduce las posibilidades de que se produzcan errores en la introducción de datos, especialmente cuando se procesan muchos documentos, permitiendo ahorrar tiempo y costes. En este sentido se reduce la necesidad de equipos

de oficina adicionales, como ordenadores de sobremesa para apoyar el procesamiento de documentos.

1.1 Documentación técnica

La documentación técnica se compone por los siguientes anexos que se describen a continuación.

- Anexo I - Planificación temporal. Incluye la planificación temporal del proyecto.
- Anexo II - Especificación de requisitos. Documento que pertenece a la primera fase del ciclo de vida del software donde se definen los requisitos esenciales del sistema.
- Anexo III - Especificación de análisis. Documento centrado en el dominio del problema y una propuesta de la primera representación técnica del sistema.
- Anexo IV - Especificación de diseño. En este documento se crea una especificación de un artefacto software.
- Anexo V - Manual de programación. Documento en el que se recoge la funcionalidad de las clases del proyecto además de algunas explicaciones de
- los componentes usados.
- Anexo VI - Manual de usuario. Documento en el que se muestra la forma de utilizar el programa para un usuario común.

2. Objetivos

El objetivo principal de este Trabajo Fin de Grado es desarrollar una aplicación que permita agilizar la llegada a un evento o a una empresa con medidas de control de acceso.

A continuación, se expondrán los diferentes objetivos que debe cumplir esta aplicación además de los objetivos personales meramente relacionados con el proyecto.

2.1 Objetivos del sistema

Los objetivos del sistema serán los siguientes:

- Agilizar la llegada a un evento o a una empresa con medidas de control de acceso

La aplicación debe permitir a un usuario introducir sus datos, escanear su documentación y enviar sus datos con la documentación escaneada para que sea verificada. La aplicación será multiplataforma por lo que se desarrollará en Java para un sistema Android.

La aplicación debe ser capaz de escanear documentación haciendo uso del escáner de la impresora y permitir rotar la foto escaneada para que el usuario la vea correctamente. También deberá permitir al usuario introducir los datos necesarios para confirmar sus credenciales.

Una vez están todos los datos correctos la aplicación deberá permitir enviar tanto los datos como los archivos escaneados por mail a una dirección de correo de los técnicos del evento o instalaciones.

Como última funcionalidad la aplicación deberá permitir configurar ciertos datos de la misma, como el email del técnico al que se envían los datos de los usuarios o los términos y condiciones, usando la interfaz web de la impresora.

- Diseñar la aplicación centrada en el usuario

Se diseñará una interfaz de usuario muy sencilla para que sea intuitiva y satisfaga las necesidades del usuario, además también deberá ser capaz de aportarle cierto feedback al usuario que la esté usando de forma que este sepa que hacer en cada momento o si falta algo como solucionarlo.

- Crear un entorno de pruebas

Para facilitar el desarrollo de la aplicación y sus diferentes pruebas se usará el simulador de Workpath. Este es un simulador que simula los sistemas operativos de las MFPs(impresoras multifunción) de HP. Añadir que este simulador viene con varios ejemplos ya instalados que hacen uso de los servicios del Workpath SDK.

2.2 Objetivos técnicos

El desarrollo de esta aplicación requerirá un proceso de investigación y aprendizaje de las diferentes tecnologías usadas en la misma, lo que facilitará un correcto desarrollo de la aplicación.

- Aprendizaje del funcionamiento del sistema operativo Android

Aprender cómo funciona el sistema operativo Android será importante para la correcta realización e implementación de las funcionalidades de la aplicación.

- Aprender a manejar el IDE Android studio

Aprender a lanzar la aplicación en el simulador de Workpath o depurarla además de aprender a importar la librería del Workpath SDK para el uso de sus APIs.

- Aprender a usar JSON

- Aprender a instalar y utilizar el simulador Workpath

- Aprendizaje del uso de ciertas APIs del Workpath SDK

Aprender a usar las APIs de scanner, email y config será indispensable para la correcta realización de este proyecto.

2.3 Objetivos personales

Uno de las motivaciones de elegir este proyecto ha sido poder realizarlo en colaboración con una empresa experta en el sector de la informática como HP. Esto me permitirá estar bajo supervisión de profesionales del desarrollo del software lo que me ayudará a conocer la metodología de trabajo que se usa en el mundo empresarial.

Además, la aplicación que se relizará hace uso de tecnologías muy comunes hoy en día, lo cual de cara al mundo laboral aporta un conocimiento de gran utilidad a futuro. Aprender a desarrollar aplicaciones multiplataforma es un punto destacable respecto a cómo se está moviendo el mundo de la informática.

Supondrá un reto aprender a usar las APIs del Workpath SDK que serán necesarias para el correcto desarrollo de esta aplicación. Esto me servirá para aprender a ser más independiente en el aprendizaje algo muy importante en este sector.

Por lo tanto, se consideran objetivos personales:

- Aprender cuál es el funcionamiento del sector a nivel empresarial
- Aprender a desarrollar un proyecto de manera profesional e independiente
- Aprendizaje de tecnologías relacionadas con el desarrollo de aplicaciones multiplataforma
- Investigación de las tecnologías necesarias para el correcto funcionamiento de la app

3. Ámbito de desarrollo

3.1 Impresoras multifunción

Las impresoras multifunción son impresoras que tiene la capacidad de realizar ciertas tareas de las que antes se tenían que encargar varias máquinas. Estas funcionalidades pueden ser imprimir, escanear, enviar correo y/o fotocopiar. No obstante, dependiendo del modelo que nos encontremos puede incorporar fax, tarjetas de memoria, disco duro, etc.

Un dispositivo multifunción (Multi Function Printer/Product/Peripheral, MFP) puede operar como periférico de entrada/salida de computadora o de modo autónomo, sin necesidad que la computadora esté encendida. Así, las funciones de fotocopiadora y fax son autónomas, mientras el escaneado, generalmente no se puede llevar a cabo sin la conexión a la computadora, aunque sí se puede escanear directamente a una memoria flash, por ejemplo, a través del puerto USB.

Los dispositivos multifunción utilizados en grandes oficinas (Figura 2) o empresas se encuentran habitualmente conectados a la red, como cualquier computadora personal. De este modo, todo el personal puede tener acceso a ellos aprovechando al máximo sus funcionalidades.

Los MFP son cada vez más importantes para las compañías, y es necesario obtener el máximo partido de sus posibilidades. Para ello, es preciso disponer del software apropiado. El software disponible para los dispositivos multifunción se puede agrupar en cinco grandes grupos:

- Administración: toda la administración del equipo se puede manejar dentro del propio dispositivo. El *host* MFP puede incluir cálculo de costes, rutinas para contar el número de impresiones en color y en blanco y negro, etcétera.
- Artes gráficas: aquí se incluyen básicamente aplicaciones que permiten elegir distintos formatos y efectos para la impresión
- Gestión de documentos: este tipo de software gestiona almacenamientos y cargas de documentos de cualquier punto de la red, los escaneos de copias físicas a documentos electrónicos en el sistema, la transformación de texto a pdf. Los MFP de mejor calidad pueden ser integrados dentro de un sistema de gestión de documentos, de modo que el almacenamiento, la distribución y la impresión de grandes volúmenes de documentos puede ser gestionada electrónicamente, a través de los dispositivos multifunción,

utilizando software de reconocimiento óptico de caracteres y otros como los ya mencionados.

- Funciones extra para la impresión: con un MFP y el software adecuado, por correo electrónico se pueden mandar los elementos escaneados, sin necesidad de pasar por una computadora.
- Software *ad-hoc*: es el software que se construye para una empresa u organización particular, y que se ajustará a las necesidades específicas que se encuentren en cada caso.

En el caso de las impresoras de HP añadir que estas cuentan con una interfaz web (Figura 1) que permitirá configurar algunas de las funcionalidades de la impresora o ver el estado de esta.

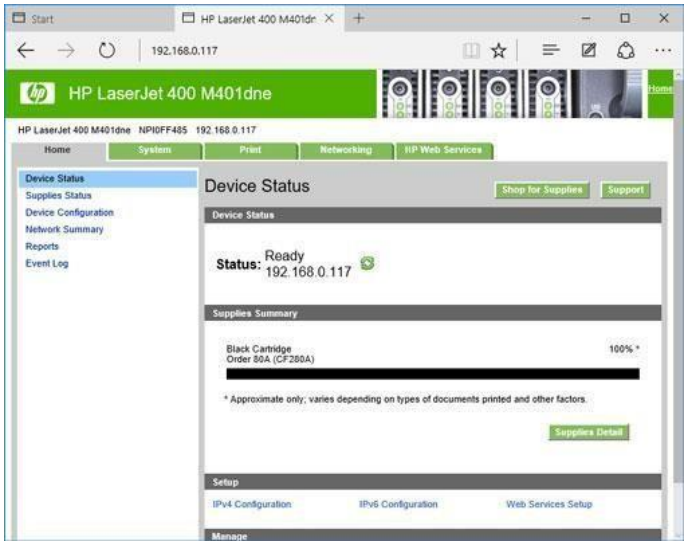

Figura 1 - UI impresoras MFP HP

Figura 2 - Impresora MFP

3.2 Android

Android es un sistema operativo móvil basado en núcleo Linux y otros softwares de código abierto. Fue diseñado para dispositivos móviles con pantalla táctil, como teléfonos inteligentes, tabletas, relojes inteligentes (Wear OS), automóviles con otros sistemas a través de Android Auto, al igual los automóviles con el sistema Android Automotive.

Los componentes principales del sistema operativo de Android:

- **Aplicaciones**: las aplicaciones base incluyen un cliente de correo electrónico, programa de SMS, calendario, mapas, navegador, contactos y otros. Todas las aplicaciones están escritas en lenguaje de programación Java.
- **Marco de trabajo de aplicaciones**: los desarrolladores tienen acceso completo a las mismas APIs del entorno de trabajo usados por las aplicaciones base. La arquitectura está diseñada para simplificar la reutilización de componentes; cualquier aplicación puede publicar sus capacidades y cualquier otra aplicación puede luego hacer uso de esas capacidades (sujeto a reglas de seguridad del *framework*). Este mismo mecanismo permite que los componentes sean reemplazados por el usuario

- **Bibliotecas**: Android incluye un conjunto de bibliotecas de C/C++ usadas por varios componentes del sistema. Estas características se exponen a los desarrolladores a través del marco de trabajo de aplicaciones de Android. Algunas son: System C library (implementación biblioteca C estándar), bibliotecas de medios, bibliotecas de gráficos, 3D y SQLite, entre otras.
- **Runtime de Android**: Android incluye un set de bibliotecas base que proporcionan la mayor parte de las funciones disponibles en las bibliotecas base del lenguaje Java. Cada aplicación Android ejecuta su propio proceso, con su propia instancia de la máquina virtual Dalvik. Dalvik ha sido escrito de forma que un dispositivo puede ejecutar múltiples máquinas virtuales de forma eficiente. Dalvik ejecutaba hasta la versión 5.0 archivos en el formato de Dalvik (.dex), el cual está optimizado para memoria mínima. La Máquina Virtual está basada en registros y utiliza clases compiladas por Java que han sido transformadas al formato dex por la herramienta incluida *dx*. Desde la versión 5.0 utiliza el ART, que compila totalmente al momento de instalación de la aplicación.
- **Núcleo Linux**: Android depende de Linux para los servicios base del sistema como seguridad, gestión de memoria, gestión de procesos, pila de red y modelo de controladores. El núcleo también actúa como una capa de abstracción entre el hardware y el resto de la pila de software.

Las aplicaciones se desarrollan habitualmente en el lenguaje Java con Android Software Development Kit (Android SDK), pero están disponibles otras herramientas de desarrollo, incluyendo un kit de Desarrollo Nativo para aplicaciones o extensiones en C o C++, Google App Inventor, un entorno visual para programadores novatos y varios marcos de aplicaciones basadas en la web multiteléfono. También es posible usar las bibliotecas Qt gracias al proyecto *Necesitas SDK*.

El desarrollo de aplicaciones para Android no requiere aprender lenguajes complejos de programación. Todo lo que se necesita es un conocimiento aceptable de Java y estar en posesión del kit de desarrollo de *software* o SDK provisto por Google el cual se puede descargar gratuitamente.[71]

Todas las aplicaciones están comprimidas en formato APK, que se pueden instalar sin dificultad desde cualquier explorador de archivos en la mayoría de dispositivos.

3.3 Workpath platform

Workpath platform es un Sistema Operativo basado en Android que se ejecuta en ciertas impresoras multifunción. No tiene tienda de Google (Google Play), ni los Google Play Services.

Tampoco es el SO principal de los dispositivos. En estos dispositivos hay dos sistemas operativos, el nativo de las impresoras y Workpath. Para acceder a la app que se ejecuta en Workpath, habrá un icono en el SO nativo que lanza la app Android.

Una aplicación Workpath se representa como un botón en la pantalla de inicio integrada (nativa). Cuando se presiona ese botón, el dispositivo cambia a la plataforma Workpath e inicia la aplicación. En este punto, la aplicación tiene las mismas capacidades de interfaz de usuario que las aplicaciones de Android que se ejecutan en otras plataformas.

Cuando se cierra la aplicación (navegando hacia atrás o presionando el botón de inicio), el dispositivo vuelve a su pantalla de inicio original. Por diseño, no se espera que las aplicaciones borren los datos de la sesión del usuario en estas transiciones. Esto permite a los usuarios salir y volver a ingresar a la aplicación sin perder el estadode esta.

Es importante comprender estas transiciones para poder implementar aplicacionesde manera adecuada.

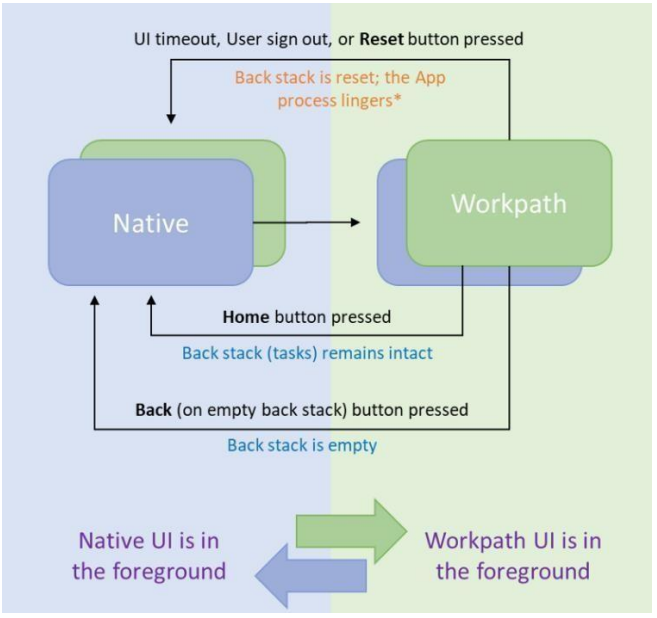

Figura 3 - Funcionamiento emulador Workpath

La Figura 3 muestra el funcionamiento de una aplicación ejecutada en un sistema Workpath. Una vez se pulsa el botón se pasa del sistema nativo al sistema Workpath que será el encargado de manejar las acciones necesarias de la aplicación cuando se usen los botones. Para volver al sistema nativo se hará siguiendo las transiciones que se muestran en la figura pulsando los botones de home, back o reset que cualquier sistema Android tiene.

3.4 Emulador Workpath

Para la realización de las pruebas de este proyecto se ha usado un emulador de un sistema Android customizado llamado Workptah. Este emulador servirá para simular las funciones de las impresoras multifunción de HP.

3.4.1 Workpath SDK

Para poder hacer uso de las funciones simuladas se usará el Workpath SDK que es un kitde desarrollo de software que HP nos brinda mediante el cual podremos hacer uso de sus APIs para así usar las funcionalidades de la impresora.

Para esto lo primero será importar la librería a nuestro proyecto una vez hecho esto ya podremos empezar a hacer uso de las APIs que nos brinda esta librería.

Como se ve en la Figura 4 Una vez hemos importado la librería nuestra aplicación hará uso a través de esta de los dispositivos de la impresora como el escáner, el servicio de correos etc.

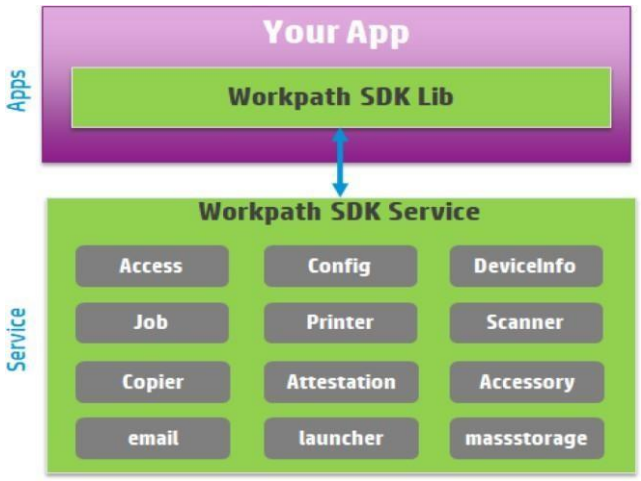

Figura 4 - Esquema Workpath SDK

El emulador nos proporciona un panel de control mediante el cual configurar alguno de los servicios del SDK. Como se ve en la Figura 5 se muestra el panel de control del emulador y las opciones que nos brinda este para poder configurar.

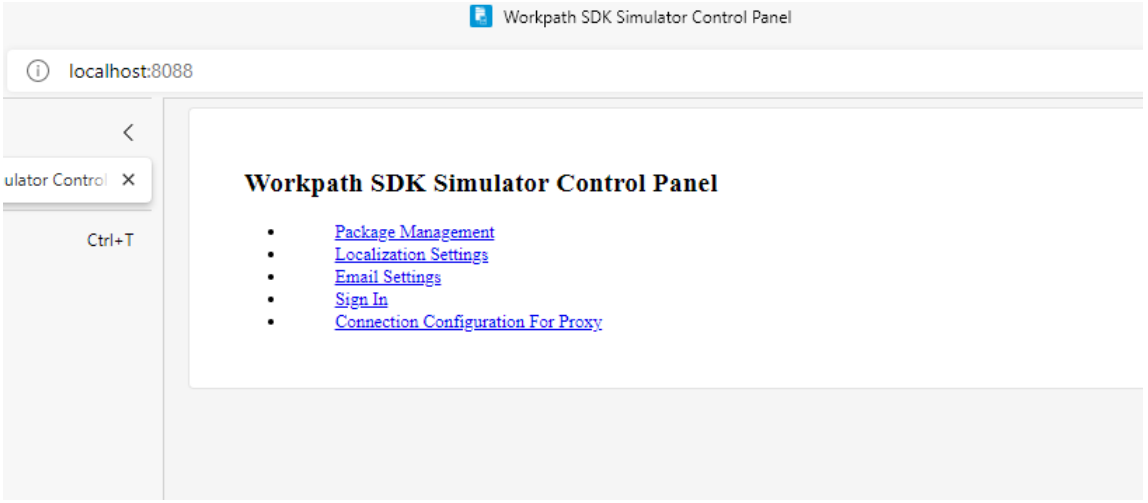

Figura 5 - Panel control emulador Workpath

4. Técnicas y herramientas

4.1 Herramientas para el desarrollo de la aplicación

Android Studio

Android Studio es el entorno de desarrollo integrado oficial para la plataforma Android. Reemplazó a Eclipse como el IDE oficial para el desarrollo de aplicaciones para Android.

Está basado en el software IntelliJ IDEA de JetBrains y ha sido publicado de forma gratuita a través de la Licencia Apache 2.0. Está disponible para las plataformas GNU/Linux, macOS, Microsoft Windows y Google Chrome OS. Ha sido diseñado específicamente para el desarrollo de Android.

Kotlin es el lenguaje preferido de Google para el desarrollo de aplicaciones de Android. Aun así, Android Studio admite otros lenguajes de programación, como Java y C ++.

Workpath simulator

Es un emulador del sistema Android que usan las MFPs de HP, para la realización deeste proyecto se ha usado la versión 1.4 que proporciona HP en su página web.

4.2 Herramientas para diseño y gestión

Visual Paradigm

Visual paradigm es una herramienta CASE para que los equipos de desarrollo modelen el sistema de información y gestionen los procesos de desarrollo. Admite estándares de modelado como UML, SoaML o XMI. [11] Esta herramienta nos permite modelar nuestro proyecto a partir de múltiples diagramas.

REM

Es un software desarrollado por el departamento de lenguajes y sistemas informáticos de la Universidad de Sevilla. Es un software de Gestión de requisitos. Éste está un poco desactualizado ya que es una tesis doctoral, pero cumple su función y es gratuito. Además, su manejo es sencillo y como se ha usado en la carrera, lo he elegido.

4.3 Lenguajes y sistemas informáticos

Java

Es un lenguaje de programación de alto nivel desarrollado por Sun microsystems. Es un lenguaje orientado a objetos, además, es un lenguaje que se puede ejecutar en cualquierplataforma, tiene la característica de ser al mismo tiempo compilado e interpretado. El compilador es el encargado de convertir el código fuente de un programa en un códigointermedio llamado *bytecode* que es independiente de la plataforma en que se trabaje y que es ejecutado por el intérprete de Java que forma parte de la Máquina Virtual de Java.

XML

Es un metalenguaje que permite definir lenguajes de marcas desarrollado por el *World Wide Web Consortium* (W3C) utilizado para almacenar datos en forma legible. Proviene del lenguaje SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML) para estructurar documentos grandes. A diferencia de otros lenguajes, XML da soporte a bases de datos, siendo útil cuando varias aplicaciones deben comunicarse entre sí o integrar información.

JSON

JSON (JavaScript Object Notation - Notación de Objetos de JavaScript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del lenguaje de Programación JavaScript. JSON es un formato de texto completamente independiente del lenguaje, pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituido por dos estructuras:

- Objeto: Colección de pares nombre/valor, donde nombre es una cadena y valor puede ser una cadena, número, otro objeto, un array, true, false, null. El objeto siempre estará limitado por "{" y "}".

  {"nombre","Alejandro"}

- Array: Un conjunto de valores separados por comas. El array es delimitado por "[" y "]" y en ocasiones también es llamado lista o vector.

  ["naranja","rojo","azul","blanco"]

Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras.

5. Aspectos relevantes

En este apartado se detallarán las partes más relevantes del desarrollo del proyecto. Comenzar exponiendo el diseño de la interfaz de la aplicación y porqué se optó por ese diseño. Luego se expondrán los servicios utilizados del Workpath SDK, además de como configurarlas a través del panel de control que el simulador ofrece y explicar el funcionamiento de cada uno de ellos. Por último, se hablará de la metodología elegida para el desarrollo del proyecto y detallar el diseño de arquitectura y los componentes que lo conforman.

5.1 Diseño de la interfaz

En lo que se refiere al diseño de la interfaz se optó por un diseño muy sencillo, para así facilitar al usuario su uso. La interfaz cuenta con unos campos a rellenar y 3 botones que solo se habilitarán cuando sea necesario, además de la opción de desplegar el documento y rotarlo una vez escaneado como se ve en la Figura 6.

UI antes de escanear:

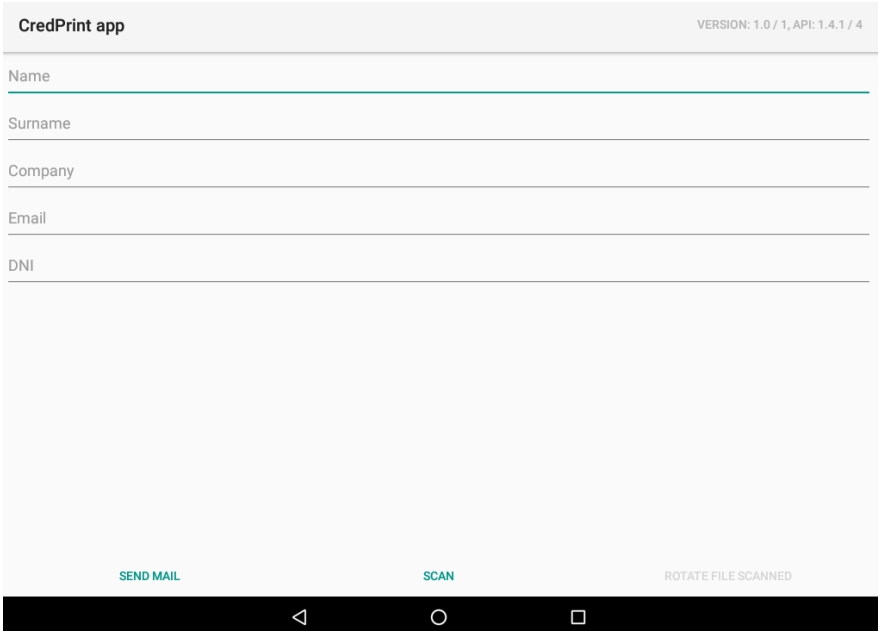

Figura 6 - UI Principal App

UI una vez se ha escaneado el archivo (Figura 7):

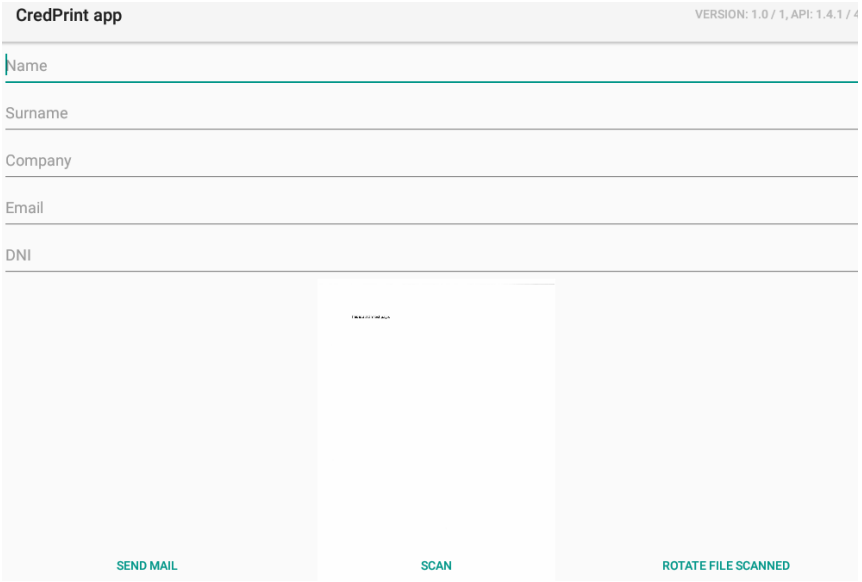

Figura 7 - UI Archivo escaneado

Se opta por una interfaz así de sencilla puesto que el objetivo es agilizar la recepción en instalaciones, dado que añadir complejidad a las funciones que el usuario pudiera realizar, sólo ralentizaría el proceso.

5.2 Metodología Scrum en el proyecto

La metodología seleccionada para la realización de este proyecto es Scrum que pertenece al ámbito de metodologías ágiles. Esta es una metodología de desarrollo rápido de software.

Utilizando Scrum como referencia se definieron los siguientes puntos (explicado Anexo I).

Roles dentro del proyecto

- Equipo de desarrollo: Este rol se decidió asignárselo al autor del TFG Alejandro González González.
- Scrum master: este rol se le asigno en conjunto a los 4 tutores del TFG.
- Product owner: Este rol también se le asigno a los tutores del TFG

Sprint backlog

Al principio de cada sprint se listan el número de tareas con las que se realizarán durante el sprint y a continuación se crea el "backlog del sprint". Esta es una lista de tareas a realizar durante el sprint. En la Figura 8 se puede ver el sprint backlog generado durante la realización de este proyecto.

ID	Nombre de las tareas	Tiempo estimado(dias)	Tiempo real (dias)	Fecha de inicio	Fecha de finalizacio	Estado
1	Sprint 1.Inicio del proyecto			09-mar	22-mar	
1.1	Estudio de los objetivos del sistema	1	0.5			Finalizado
1.2	Estudio de las tecnologias seleccionadas	2	1.5			Finalizado
1.3	Planificacion y distribucion de los futuros sprints	2	1			Finalizado
1.4	Entregable:Instalacion de todas las tecnologias					Entregado
2	Sprint 2.Instalacion del entorno y realizacion de pruebas			23-mar	05-abr	
2.1	Instalacion de workpath sdk	1	1			Finalizado
2.2	Pruebas del entorno workpath sdk	1	1			Finalizado
2.3	Instalacion y pruebas del IDE Android studio	1	1			Finalizado
2.4	Seguimiento del curso de Advanced android app devel	15	7			No finalizado
2.5	Entregable:Finalizacion del curso de Android					No entregado
3	Sprint 3. Implementacion UI sistema			06-abr	19-abr	
3.1	Seguimiento del curso de Advanced android app devel	15	8			Finalizado
3.2	Desarrollo de la UI del sistema	2	2			Finalizado
3.3	Entregable:Primera version con una UI funcional					Entregado
4	Sprint 4.Desarrollo de la funcion de scaneo			20-abr	03-may	
4.1	Estudio de la Api de scaneo del workpath sdk	4	5			Finalizado
4.2	Implementacion de la funcionde escaneo	5	5			Finalizado
4.3	Entregable:Segunda version con funcionalidad escaneo					Entregado
5	Sprint 5.Desarrollo de la funcion de send email			04-may	17-may	
5.1	Estudio de la Api de email del workpath sdk	4	3			Finalizado
5.2	Implementacion de la funcion de envio de email	4	3			Finalizado
5.3	Entregable:Tercera version con funcionalidad de email					Entregado
6	Sprint 6. Desarrollo de la funcion de config			18-may	31-may	
6.1	Estudio de la Api de config del workpath sdk	4	3			Finalizado
6.2	Implementacion de la funcion config	4	4			Finalizado
6.3	Entregable:Tercera version con funcionalidad de config					Entregado
7	Sprint 7.Desarrollo funcionalidades extra			01-jun	14-jun	
7.1	Implementacion funcionalidad mostrar documento	1	1			Finalizado
7.2	Implementación funcionalidad rotar documento	1	0.5			Finalizado
7.3	Aumentar feedback del sistema	1	1			Finalizado
7.4	Documentacion Memoria	15	7			No finalizado
7.5	Entregable:Version final de la app					Entregado
8	Sprint 8. Elaboracion de la documentacion			15-jun	28-jun	
8.1	Documentacion Memoria	15	8			FInalizado
8.2	Realizacion de Anexos	2	2			Finalizado
8.3	Entregable:Documentacion completa					Entregado

Figura 8 - Sprint backlog

Para la realización del sprint backlog se fueron poniendo las tareas necesarias a realizar para conseguir completar el entregable de la semana.

En las primeras semanas se hizo un gran esfuerzo en aprender a usar las tecnologías y sistemas necesarios para conseguir finalizar el proyecto, además de una correcta instalación de éstas para la futura realización de pruebas.

En las siguientes semanas se pasó al desarrollo de la aplicación, se diseñó la interfaz y se fueron implementando las funcionalidades necesarias para el correcto funcionamiento de esta una vez se habían estudiado.

Las últimas semanas se dedicaron a la redacción de esta memoria y la documentación pertinente que se ha de realizar para un sistema software.

5.3 El proceso de desarrollo del software

El proceso de desarrollo del software define un conjunto estructurado de actividades para desarrollar un sistema de software de alta calidad y proporciona un marco de trabajo desde el cual poder establecer un plan detallado para el desarrollo del software.

El proceso de desarrollo del software se ha dividido en 4 fases con finalidades dispares.

Para la realización de este proyecto se ha seguido la estructura del proceso unificado: inicio (Anexos I y Anexo II), elaboración (Anexo III), construcción (Anexo IV y Anexo V) y transición (Anexo VI).

5.3.1 Estudio de las tecnologías

Dado que este proyecto se realizaba en colaboración con HP se planteó desde un principio para que el desarrollo de este proyecto se llevara a cabo con el uso de sus impresoras.

Como la aplicación tenía que ser multiplataforma y se iba a usar un sistema customizado de Android, fue imprescindible aprender los conceptos importantes del funcionamientode este sistema operativo, como, por ejemplo, programar para una mayor eficiencia de la aplicación, además de aprender algunos conceptos que introduce este sistema operativo.

5.3.1.1 Life cycle Android

Para poder entender el funcionamiento de una aplicación en Android es necesario que entendamos todo su ciclo de vida, es decir, como actúan sus procesos desde que la instalamos y la abrimos hasta que la cerramos.

En el caso de Android una aplicación pasa por varios procesos, es muy importante entender que cada pantalla que tenga una app pasa por estos procesos que vamos a explicar:

1. OnCreate: Creación de la app, cada vez que iniciamos una aplicación o creamos una actividad (pantalla) nueva. En este proceso la aplicación aún no tiene UI (User Interface), solo las funciones que va a usar para cargar los datos necesarios para el funcionamiento de la app.
2. OnStart: En este punto la app inicia los componentes gráficos que vemos por pantalla y se vuelve visible. Además, prepara la UI para que el usuario pueda interaccionar con ella.
3. OnResume: Este proceso se mantiene activo mientras el usuario haga uso de la UI, pulsar botones, escribir texto, seleccionar elementos…
4. OnPause: A este estado se llega siempre desde el estado onResume y podemos volver a onResume a través de este estado. Este estado se origina generalmente cuando la aplicación pasa a segundo plano o abrimos otra actividad, si es el primer caso la aplicación se recuperará, sin embargo, si abrimos otra actividad completamente pasará de este estado a pararse.
5. OnStop: Este estado del lifecycle de una aplicación ocurre siempre después de que la pausáramos previamente. La aplicación ya no está en segundo plano ya que deja de ser visible para el usuario, aunque puede tener procesos ejecutándose en el sistema. Hay dos opciones: llamar a onDestroy y cerrarse porcompleto o volver a cargar la actividad sin cargar todo desde cero. Android cuando una app lleve cierto tiempo en segundo plano o parada, pasará a onDestroy automáticamente.
6. OnDestroy: Llegados aquí ya no hay vuelta atrás, la aplicación detecta que la actividad se ha cerrado y por lo tanto destruye todos los procesos e información que no han sido almacenados.

5.3.1.2 Estudio Workpath emulator

Una vez entendido como funciona el sistema Android fue necesario entender las peculiaridades del Workpath y cómo usar los servicios que ofrece su SDK para esto se hizo uso de los ejemplos que HP nos permite descargar, de forma que se descargaron los ejemplos, se ejecutaron y se depuraron para ver el funcionamiento de sus servicios.

5.3.2 Modelos del proceso software

A continuación, se expondrán algunos de los modelos usados para la realización de este sistema.

5.3.2.1 Modelo de casos de uso

Un caso de uso es una técnica de modelado usada para describir lo que debería hacer un sistema nuevo o lo que hace un sistema que ya existe.

A continuación, en la Figura 9 se muestra un paquete de casos de uso que se modeló durante la especificación de requisitos, para más información consultar el Anexo II.

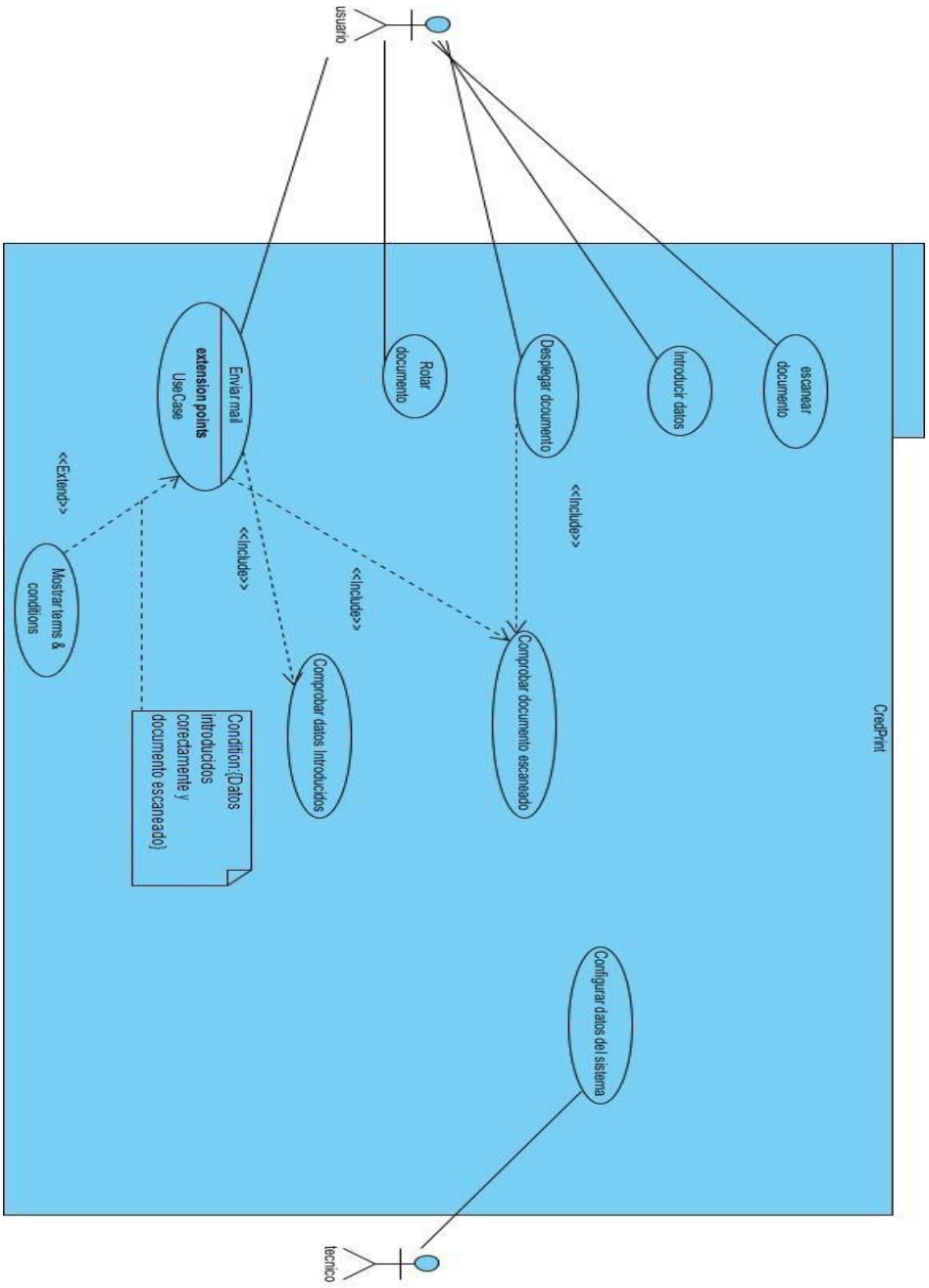

Figura 9 - Paquete casos de uso I

Para la realización de este paquete de casos de uso lo que se hizo fue identificar las acciones que podrían realizar los actores correspondientes a este.

A continuación, se describirán los casos de usos de este paquete:

Enviar email: El actor usuario ejecutará este caso de uso cuando quiera enviar un correo electrónico con su documentación y datos.

Escanear documento: Caso de uso que será ejecutado cuando el usuario quiera escanear documentación.

Desplegar documento: Caso de uso que el usuario seleccionará cuando quiera desplegar el documento escaneado.

Rotar documento: Caso de uso que el usuario seleccionará cuando quiera rotar el documento escaneado.

Introducir datos: Caso de uso que permite al usuario introducir sus datos personales.

Configurar aplicación: Caso de uso que permite al técnico configurar aspectos de la aplicación.

Una vez identificados los casos de uso se establecerá la secuencia normal de pasos y en caso de que sea necesario sus excepciones. A continuación, en la Figura 10 se ve como se explican las secuencias del caso de uso “enviar mail”, además de mostrar una breve descripción del caso de uso. Para más ejemplos, Anexo II.

UC-0005	**Enviar mail**	
Versión	1.0 (27/06/2021)	
Autores	• Alejandro González González	
Fuentes	?	
Dependencias	Ninguno	
Descripción	El sistema deberá comportarse tal como se describe en el siguiente caso de uso cuando *el usuario una vez haya introducido susa datos y escaneado su documentacion clicara en envair email y el sistema enviara el mail con la documentacion adjuntada y los datos* o durante la realización de los siguientes casos de uso: [UC-0006] Terms & conditions	
Precondición	PD	
Secuencia normal	**Paso**	**Acción**
	1	El actor Usuario (ACT-0001) *Clicara en enviar email*
	2	El sistema *comprobara que los datos introducidos son correctos*
	3	El sistema *comprobara que se ha escaneado documentacion*
	4	Se realiza el caso de uso Terms & conditions (UC-0006)
	5	El sistema *envia el email con los datos introducidos y adjuntando la documentacion*
Postcondición	PD	
Excepciones	**Paso**	**Acción**
	2	*Si Faltan datos por introducir o no estan en un formato adecuado*, el sistema *indica el fallo* , a continuación este caso de uso *queda sin efecto*
	3	*Si si no se ha escaneado documentacion*, el sistema *indica el fallo*, a continuación este caso de uso *queda sin efecto*
Rendimiento	**Paso**	**Tiempo máximo**
	-	-
Frecuencia esperada	PD	
Importancia	PD	
Urgencia	PD	
Estado	PD	
Estabilidad	PD	
Comentarios	Ninguno	

Figura 10 - Rem caso de uso enviar mail

5.3.2.2 Modelo de dominio

El modelo de dominio en la resolución de problemas e ingeniería de software, es un modelo conceptual de todos los temas relacionados con un problema específico. En él se describen las distintas entidades, sus atributos, papeles y relaciones, además de las restricciones que rigen el dominio del problema.

En general, las entidades conceptuales identificadas inicialmente en el modelo de dominio, se convertirán en entidades reales que se guardarán en el sistema. Es decir, el diagrama de clases en el modelo de dominio, evolucionará hasta un modelo de datos.

Sin embargo, en el proyecto actual el modelo de datos es muy escaso, aunque si se ha modelado un diagrama de clases. Para más información Anexo III.

Durante la realización del modelo de dominio de este sistema se identificó que no se necesitaría almacenar mucha información. Esta información contendrá los datos personales del usuario que podrán variar por región, el o los documentos escaneados y además seidentificó que se necesitaría guardar los datos referentes a una cuenta de correo electrónico y a un mensaje de correo electrónico. En la Figura 11 se puede ver el diagrama de clases generado en la especificación del análisis.

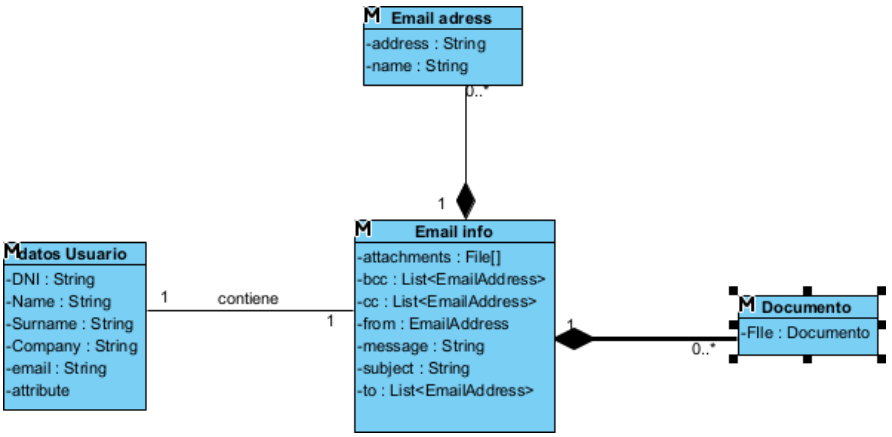

Figura 11 - Diagrama clases análisis

Las clases de este diagrama de clases son:

Datos usuarios: Clase que guardará los datos personales de un usuario.

Email address: Entidad que se usará para almacenar una dirección de email con nombre de usuario.

Email info: Entidad usada para construir el email que se enviará para generar la acreditación.

Documento: Entidad que representa a un documento escaneado.

5.3.2.3 Diagramas de interacción

Una vez realizada la estructura estática del proyecto, se diseña la estructura dinámica. En este apartado se diseña la interacción entre los distintos objetos o componentes del sistema a partir de los casos de uso ya establecidos.

Durante la realización de este proyecto se han realizado diagramas de secuencia y diagramas de actividad.

Un diagrama de secuencia del sistema muestra, para un escenario específico de un caso de uso, los eventos que generan los actores externos, el orden y los eventos entre los sistemas.

Un diagrama de actividad representa el comportamiento mediante un modelo de flujo de datos y flujo de control.

En la Figura 12 se muestran las entidades que se usarán para el modelado del diagrama de secuencia del caso de uso “enviar mail”.

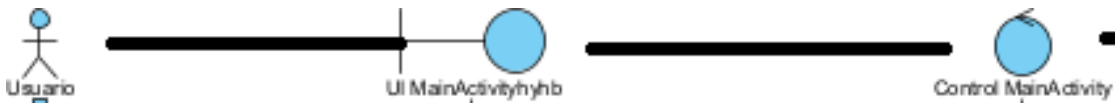

Figura 12 - Diagrama clase de análisis de enviar mail

La entidad “UI MainActivity” hace referencia a la vista de la actividad principal con la que interacciona el usuario.

“Control MainACtivity” es la entidad que se encargará de recibir las interacciones del usuario.

La Figura 13 es un diagrama de secuencia que muestra las interacciones entre el usuario y las clases, además de la interacción entre las propias clases. Para más ejemplos consultar el Anexo III.

En la Figura 13 se muestra como el usuario, al clicar el botón send mail hace que la entidad “UI MainActivity” le envíe mensajes al “Control MainActivity”, para que esta entidad instancie y ejecute la tarea encargada de enviar un mail “SendEmailTask”. Una vez hecho esto la tarea indicará a la entidad “Control MainActivity” que finalizó y este se lo indicará a “UI MainActivity” que se lo indicará al usuario. En caso de algún problema los mensajes cambiaran y se indicará el fallo.

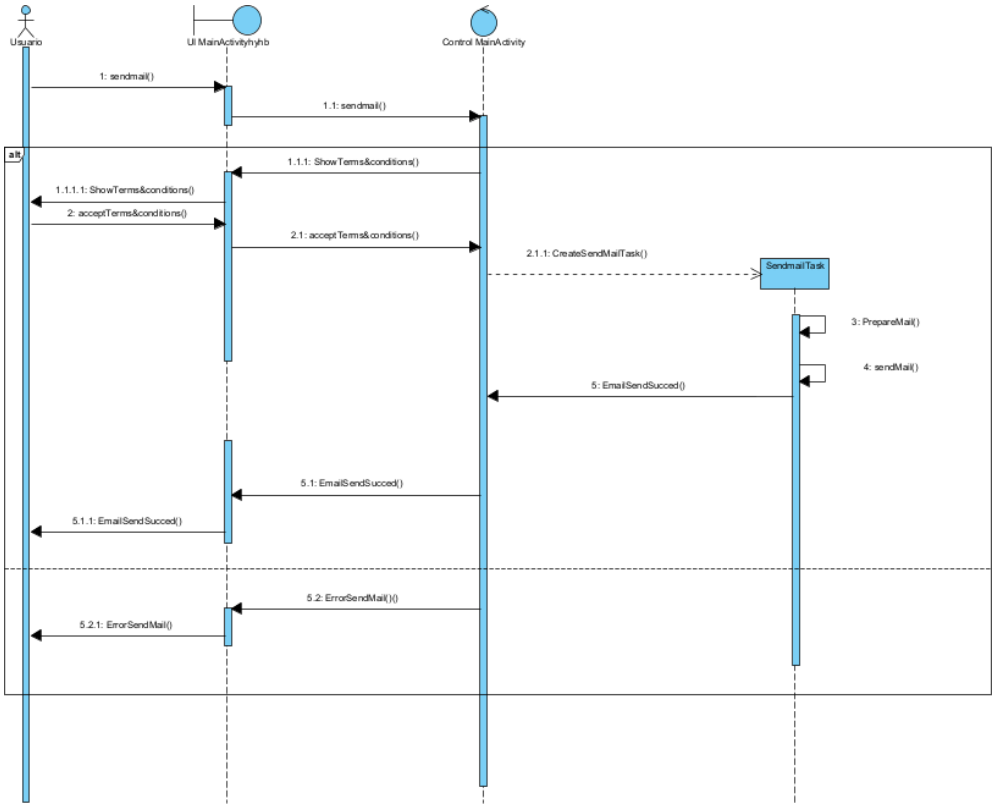

Figura 13 - Diagrama secuencia send mail

En la Figura 14 se muestra el diagrama de actividad del caso de uso “enviar mail”, este diagrama muestra las actividades por las que se puede pasar durante la realización de este caso de uso, como llegar a ella y como llegar a la finalización de este caso de uso.

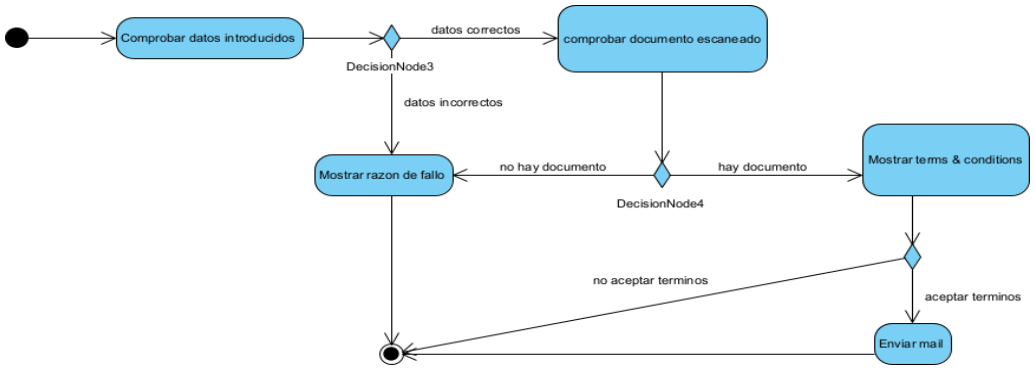

Figura 14 - Diagrama de actividad enviar mail

5.3.3 Patrones arquitectónicos

Los patrones arquitectónicos, o patrones de arquitectura, también llamados arquetipos ofrecen soluciones a problemas de arquitectura de software en ingeniería de software.

En este apartado se explicarán los patrones arquitectónicos elegidos para el proyecto. Además, se explicarán los paquetes del sistema y sus relaciones a través de un diagrama de paquetes que se mostrara a continuación (Figura 15).

5.3.3.1 MVC

Para el desarrollo del Sistema se utilizará el patrón MVC (Modelo-Vista-Controlador). Este patrón se compone de tres capas, que se encargan de las diferentes tareas:

- Modelo: Es la capa donde se trabaja con los datos, por tanto, contendrá mecanismos para acceder a la información y también para actualizar su estado.

- Vista: Contiene el código de nuestra aplicación que va a producir la visualización de las interfaces de usuario.
- Controlador: Contiene el código necesario para responder a las acciones que se solicitan en la aplicación, como por ejemplo visualizar un elemento.

Como se ve en la Figura 15, se muestra un diagrama que muestra de manera detalladalos paquetes del diagrama de clases siguiendo el patrón MVC.

Para la aplicación de MVC en este proyecto se identificaron los 3 paquetes del patrón:

Vista: A la vista se añadieron las Interfaces de usuario que se decidió que iba a tener la aplicación. Estas serán cuatro vistas, aunque realmente será una vista que desplegará otras vistas más pequeñas sobre ella, sin que esta vista padre desaparezca de la pantalla.

Controlador: Para el diseño del controlador se estableció que cada vista tendría un controlador encargado de manejar a cada una de ellas, además se añadió al controlador las distintas clases encargadas de realizar las tareas necesarias para el correcto funcionamiento de la aplicación.

Modelo: En el caso del modelo que es la parte más simple del sistema, se diseñaron las clases necesarias para almacenar la información de un email.

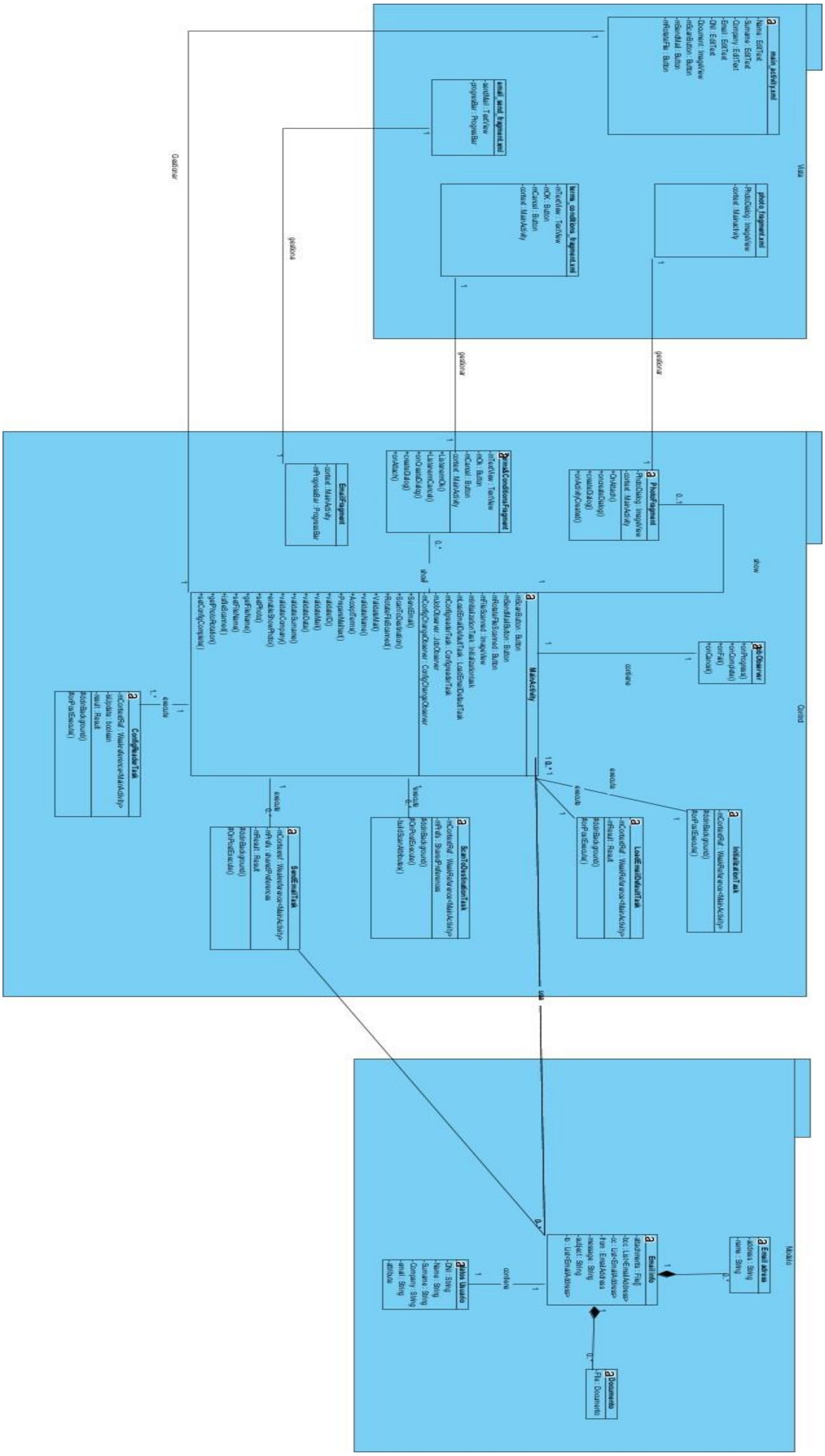

Figura 15 - MVC proyecto

A continuación, se describirá cada uno de los paquetes del patrón MVC por separado:

1. Vista

   Este paquete es el que contiene todas las vistas de la aplicación (Figura 16):

   activity_main.xml: Fichero XML en el que se encuentra la interfaz gráfica de la actividad principal.

   Photo_fragment.xml: Fichero XML en el que figura el dialogfragment que se despliega cuando quieres desplegar la foto.

   terms_conditions_fragment.xml: Fichero en el que figura el dialogfragment de los términos y condiciones.

   Email_send_fragment.xml: Fichero en el que se encuentra el dialogfragment que se mostrará mientras se esté enviando el mail.

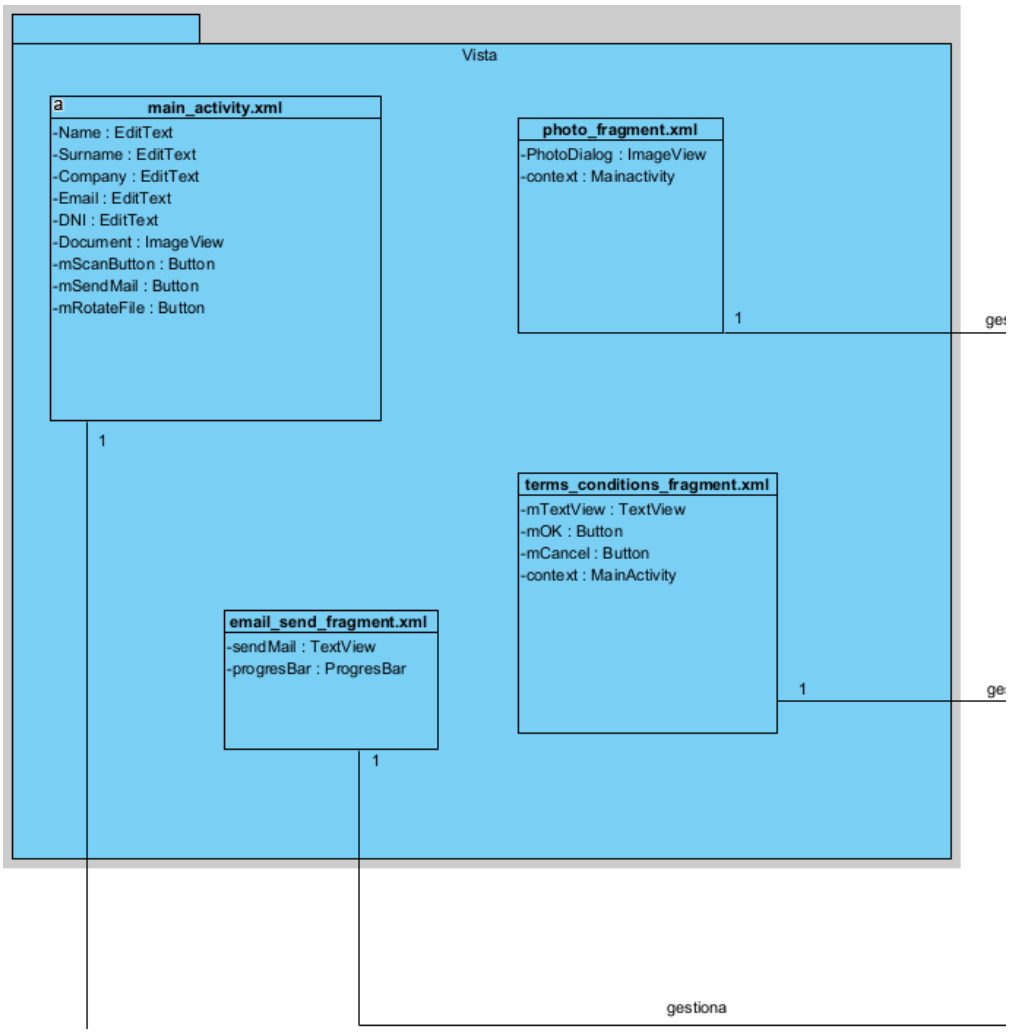

Figura 16 - MVC vista

2. Controlador

Este paquete contiene todas las clases que gestionan el funcionamiento lógico de la aplicación (Figura 17).

MainActivity.java: Controlador que se encarga de recibir y gestionar todos los eventos provenientes de la interfaz activity_main.xml, además de lanzar todas las tareas asíncronas necesarias.

PhotoFragment.java: Controlador que gestiona la vista photo_fragment.xml.

TermsConditionsFragmen.java: Controlador que gestiona la vista terms_conditions_fragment.xml.

EmailFragment.java: Controlador que gestiona la vista que se muestra mientras se envía el correo email_send_fragment.xml

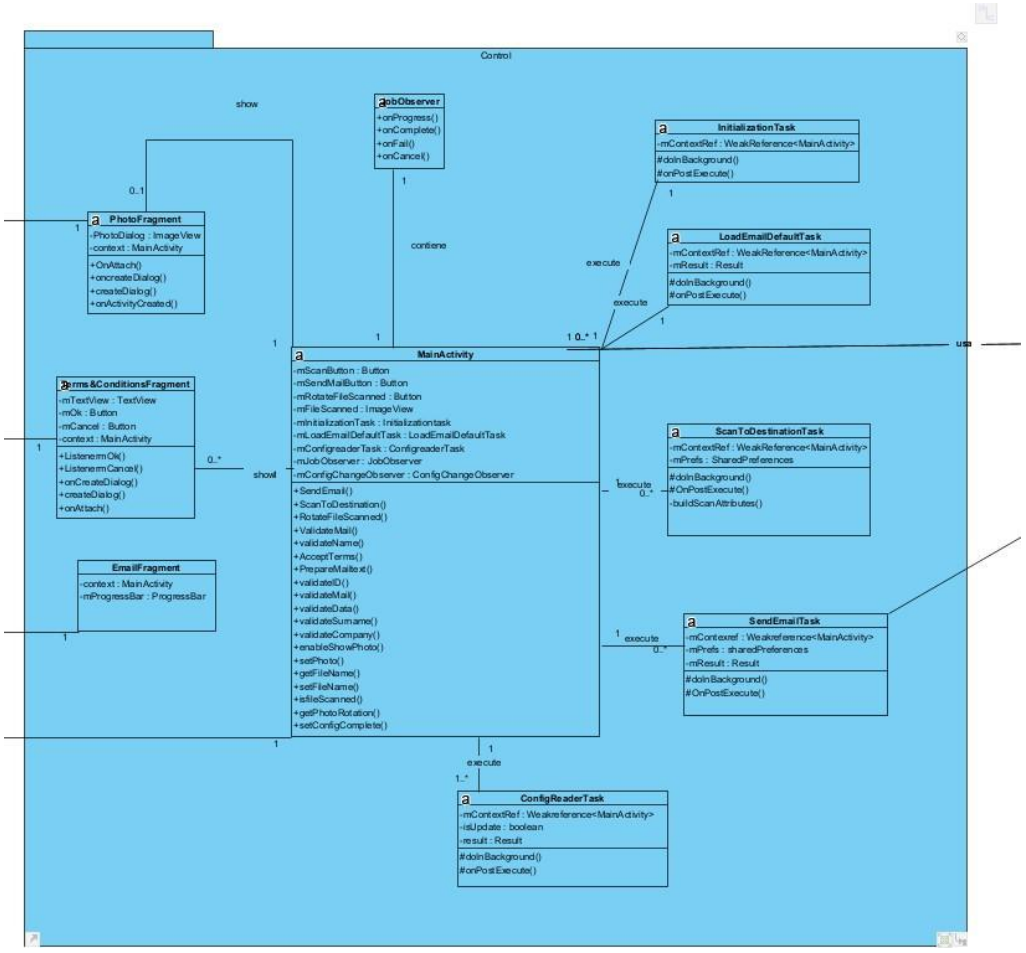

Figura 17 - MVC controlador

3. Modelo

Este paquete contiene las clases que guardan la información de la aplicación (Figura 18).

Email Address: Clase que se usará para guardar una dirección de mail y su nombre.

Email Info: Clase que se usa para guardar todos los datos necesarios para enviar el mail, además del documento escaneado que se envía.

Datos Usuario: Clase que se usa para guardar los datos personales que se creen necesarios para las personas que usan la aplicación.

Documento: Hace referencia al documento escaneado.

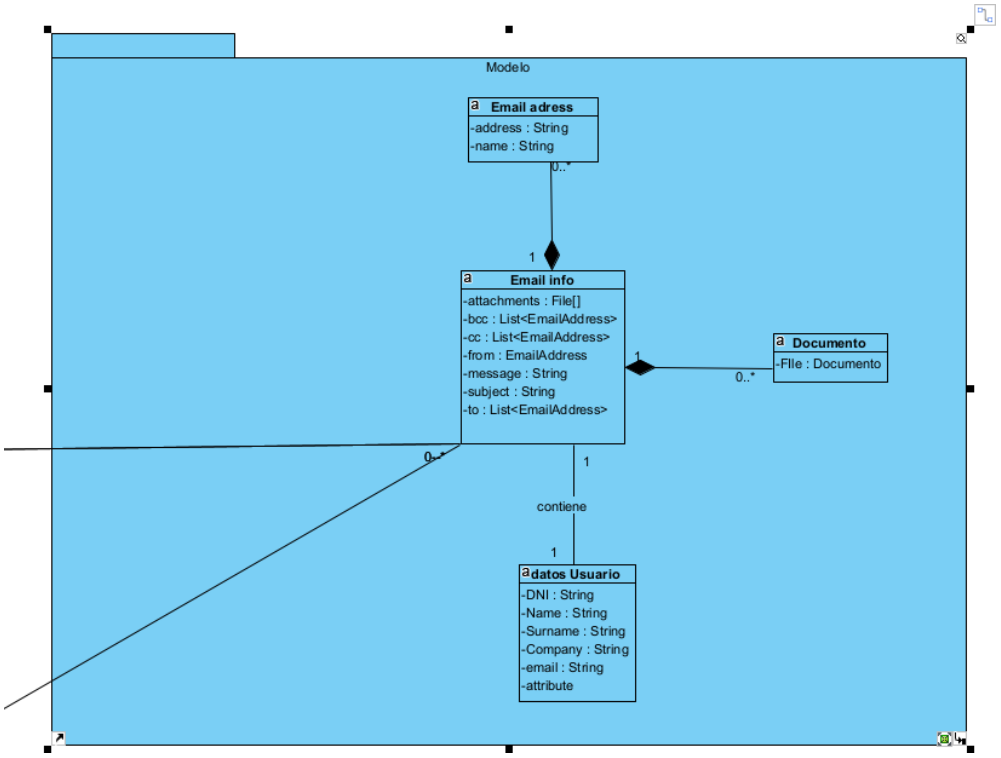

Figura 18 - MVC modelo

5.4 Scanner service

El servicio de escaneo es el primer servicio implementado en la aplicación. Para llevarla cabo se ha utilizado el Workpath SDK que nos brinda servicio de escaneo.

Este servicio nos permite enviar el archivo escaneado a un USB, enviarlo por mail, guardarlo en la memoria de la impresora, etc. En este caso se usó la opción de guardar el archivo escaneado en la memoria de la impresora porque así lo especificó el clientede la aplicación.

5.5 Email service

Para el desarrollo de esta funcionalidad se utilizó el servicio de email que nos proporciona Workpath SDK. Como vimos antes, el emulador nos ofrecía un panel de control para configurar ciertos servicios. El email es uno de esos servicios que se pueden configurar. En la Figura 19 se muestra el panel de control para configurar el email.

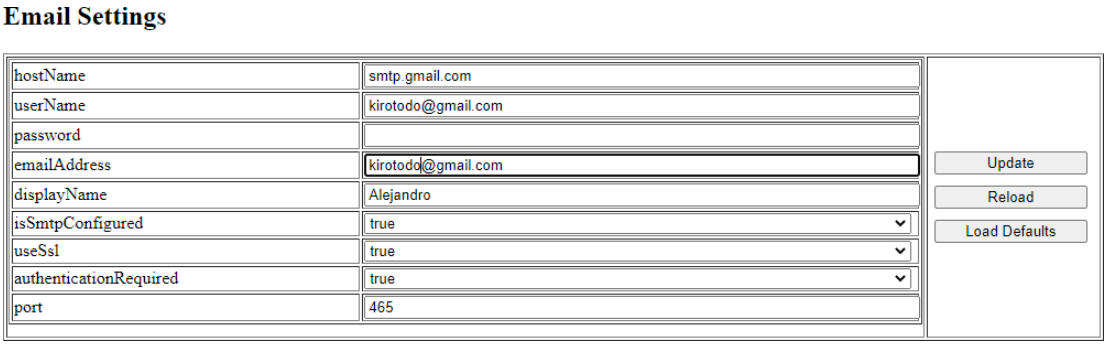

Figura 19 - Panel control emulador email

Este panel nos permitirá establecer la dirección de email a través de la cual se enviarán los correos con los datos y archivos escaneados.

El SDK provee una API para obtener el email por defecto que se ha establecido a través del panel de configuración. En la Figura 20 se ve el código para obtener el mail configurado a través del panel de control.

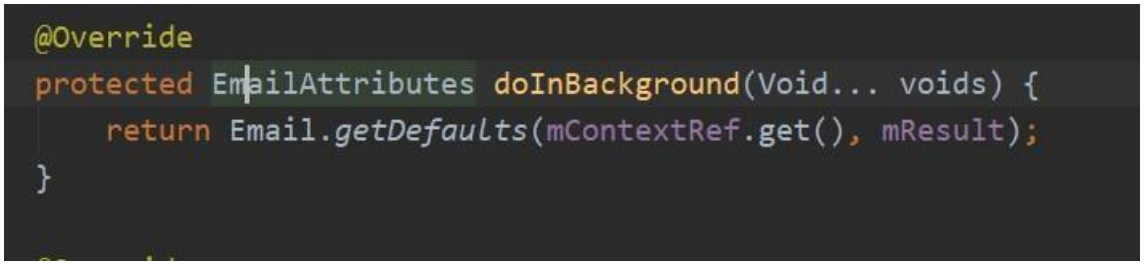

Figura 20 - Código email service

Además, el servicio de email proporciona una API por opciones para enviar correos electrónicos en Workpath que nos permite que la aplicación pueda enviar un correo a través de su propio servidor smtp.

5.6 Config service

El servicio config nos permite configurar nuestra aplicación Workpath cuando queremos que sea la propia aplicación quien gestión su configuración.

Como hemos visto antes ya se ha establecido el email a través del cual enviar los correos, pero falta configurar a que dirección enviarlos. En este caso el servicio config nos permitirá establecer el email destino de los correos.

Para poder hacer uso de este servicio lo primero sería entrar al panel de control que nos brinda el emulador.

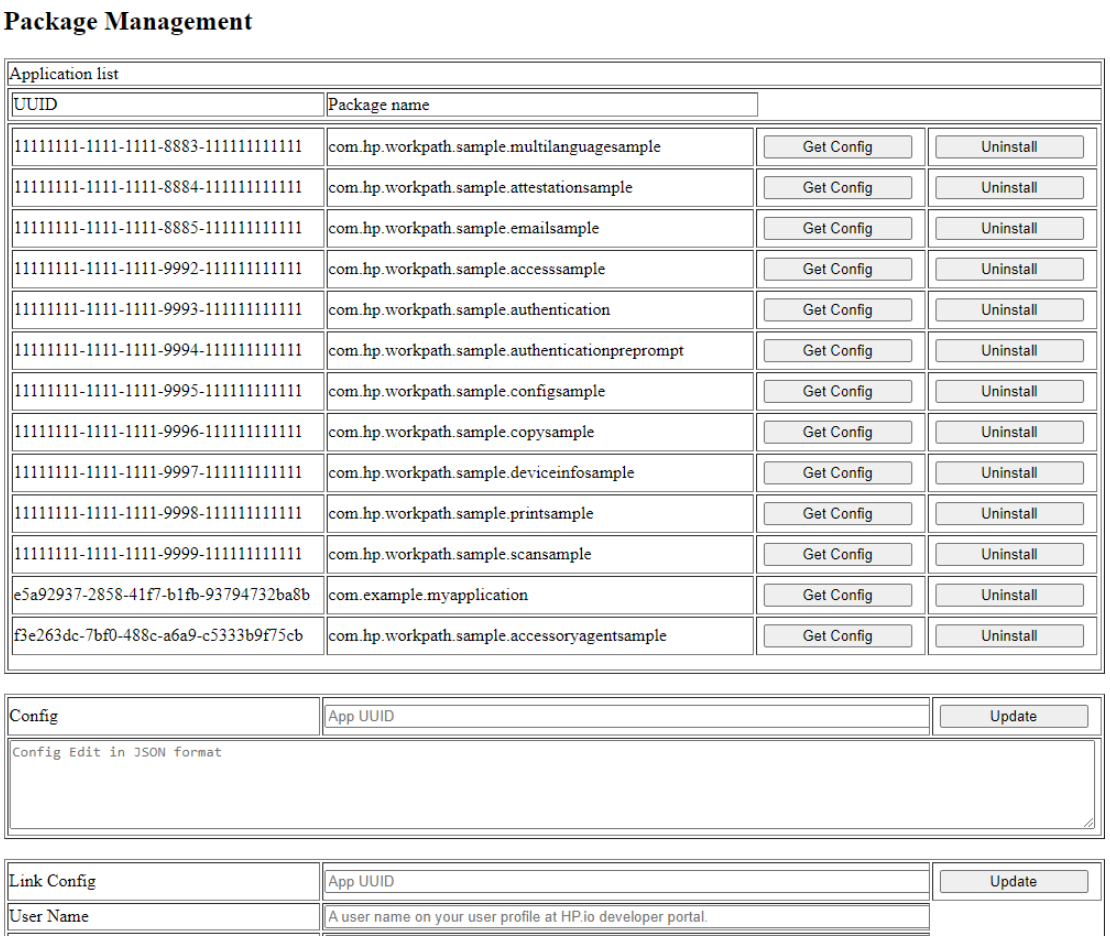

Figura 21 - Panel control emulador config

Remarcar que para que nuestra aplicación aparezca en el panel de control en la sección package management como se ve en la Figura 21, hay que instalarla en el emulador y para ello se necesitará generar un archivo hpk. El archivo hpk viene a sustituir los apk de Android para aquellas aplicaciones que usen Workpath SDK. Para generarlo se hace a través del apk de la aplicación usando el hpktool que es un programa que HP nos ofrece. La interfaz de este programa se ve en la Figura 22.

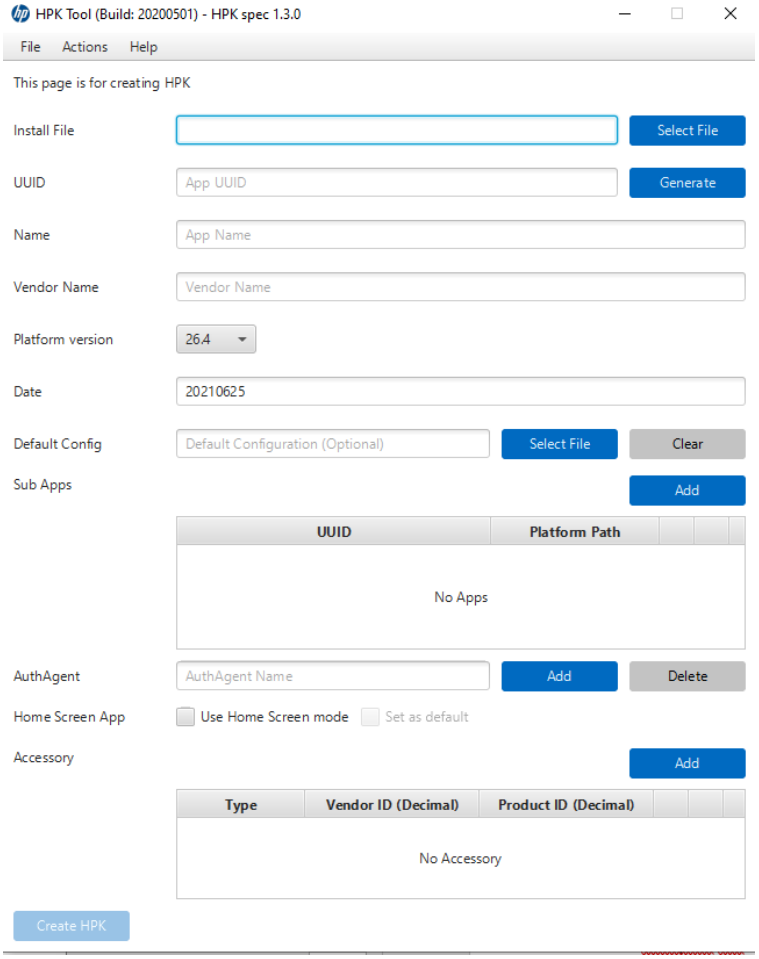

Figura 22 -HPK Tool

Una vez creado el hpk se usará el panel de control del simulador que se muestra en la Figura 23 para instalar la app usando el hpk. Una vez hecho esto ya se nos permitirá usar el servicio config en nuestra aplicación.

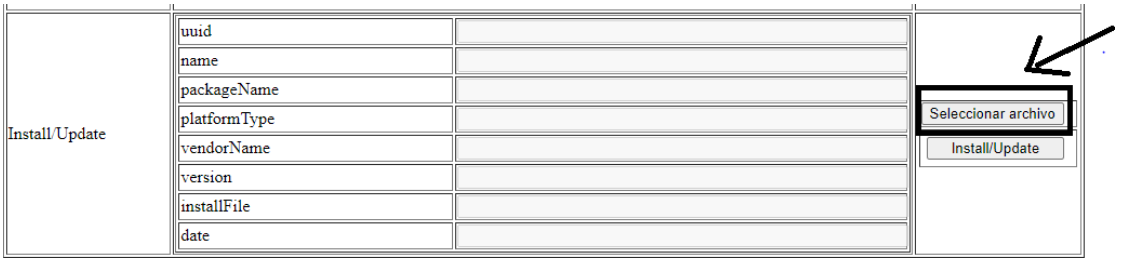

Figura 23 - Panel control config hpk

En este caso se ha usado el servicio config para enviar a través de un JSON tanto el email destino como los términos y condiciones de la aplicación.

5.7 Puntos a destacar de la aplicación

En este apartado se explicarán aspectos que se considera que han sido importantes para el desarrollo de la aplicación.

5.7.1 AsyncTask

Dado que la aplicación se desarrolla en un sistema Android hay que remarcar el uso de los AsyncTask, ya que es un concepto muy importante para la ejecución de servicios en un sistema Android. Esto se debe a que en Android una actividad gestiona también la UI que muestra, luego si esa actividad realiza alguna tarea que requiera un tiempo considerado, la aplicación se congelaría hasta que esta terminase. Sin embargo, los AsyncTask son la solución que Android brinda para evitar esto. Los AsyncTask son tareas asíncronas que se ejecutan en segundo plano sin congelar la interfaz de la actividad que los lanza, permitiéndonos usar la UI de la actividad a la vez que se realiza la tarea por detrás. Añadir que para la realización de este proyecto se han usado cinco AsyncTasks.

Para el uso de un AsyncTask lo primero es crear una clase que herede de la clase AsyncTask Figura 24.

```
public class SendEmailTask extends AsyncTask<Void, Void, Result> {

    private static final String TAG = MainActivity.TAG;

    private final WeakReference<MainActivity> mContextRef;
    private final SharedPreferences mPrefs;

    private Result mResult;
    private String mErrorMsg = null;

    public enum SendType {
        SEND
    }

    private SendType sendType;

    private EmailInfo emailInfo;
```

Figura 24 - AsyncTask

Lo siguiente seria sobrescribir el método doInBackground, en este método ira todo lo que habrá que ejecutar de forma asíncrona Figura 25.

```
@Override
protected Result doInBackground(Void... voids) {
    try {


        EmailAttributes.Builder emailAttrBuilder = new EmailAttributes.Builder()
                .setFrom(emailInfo.getFrom().getAddress(), emailInfo.getFrom().getName())
                .addAttachments(emailInfo.getAttachments()) //adjuntar el archivo scaneado hacer luego
                .setSubject(emailInfo.getSubject())
                .setMessage(emailInfo.getMessage());


        if (emailInfo.getTo() != null && emailInfo.getTo().size() > 0) {
            for (EmailAddress mail : emailInfo.getTo()) {
                if (TextUtils.isEmpty(mail.getName())) {
                    emailAttrBuilder.addToAddresses(mail.getAddress());
                } else {
                    emailAttrBuilder.addToAddress(mail.getAddress(), mail.getName());
                }
            }
        }


        EmailAttributes emailAttributes = emailAttrBuilder.build();
        Log.i(TAG, msg: "EmailAttributes=" + Logger.build(emailAttributes));

        switch (sendType) {
```

Figura 25 - DoInBackground

Añadir que también puede ser útil sobrescribir el método onPostExecute que nos servirá para hacer algo una vez la tarea haya terminado Figura 26.

```
@Override
protected void onPostExecute(final Result result) {
    super.onPostExecute(result);
    if (result != null && result.getCode() == Result.RESULT_OK) {
        mContextRef.get().showResult("Email send succeed");
    } else if (mErrorMsg != null) {
        mContextRef.get().showResult(mErrorMsg);
    } else {
        mContextRef.get().showResult( msg: "Email.send(): ", result);
    }
    mContextRef.get().showProgressBar(View.GONE);
    mContextRef.get().endmail();
    mContextRef.get().deleteData();
}
```

Figura 26 - OnPostExecute

5.7.2 BroadCastReceiver

Un Broadcast Receiver es un componente que está destinado a recibir y responder ante eventos globales generados por el sistema, como un aviso de batería baja, un SMS recibido, un SMS enviado, una llamada, un uso del escáner o cámara, etc y también eventos producidos por otras aplicaciones.

En el caso de este proyecto se ha usado para recibir eventos del sistema de escaneo, en concreto para saber cuándo ha terminado de escanear la impresora.

Para el uso de un BroadCastReceiver lo primero sería crear una clase que herede de BroadCastReceiver y sobrescribir el método onReceive e indicar en este que evento registrar (ACTION_SCAN_COMPLETED) Figura 27.

```
public final class JobCompleteReceiver extends BroadcastReceiver {
    private static final String TAG = MainActivity.TAG;

    public static final String RID_EXTRA = "rid";
    public static final String JOB_ID_EXTRA = "jobid";
    public static final String CURRENT_JOB_ID = "pref_currentJobId";
    @Override
    public void onReceive(final Context context, final Intent intent) {
        if (intent == null) {
            Log.d(TAG, msg: "Received intent is null");
            return;
        }

        final String action = intent.getAction();
        final ComponentName component = intent.getComponent();
        // Verify that received Job Id is same as expected one
        final String jobId = intent.getStringExtra(JOB_ID_EXTRA);
        final String expectedJobId = PreferenceManager.getDefaultSharedPreferences(context)
                .getString(CURRENT_JOB_ID, defValue: null);

        if (MainActivity.ACTION_SCAN_COMPLETED.equals(action) &&
                component != null && context.getPackageName().equals(component.getPackageName()) &&
                jobId.equals(expectedJobId)) {
            Log.d(TAG, "Monitoring Job: received Completed intent");
        }
    }
}
```

Figura 27 - BroadCastReceiver

5.8 Pruebas

Las pruebas se han ido realizando durante todo el proceso de desarrollo del proyecto.

5.8.1 Pruebas emulador Workpath

Dada la situación solo se han podido realizar pruebas en el emulador de Workpath que ayudaba simulando una impresora MFP de HP. La interfaz del simulador se puede ver en la Figura 28.

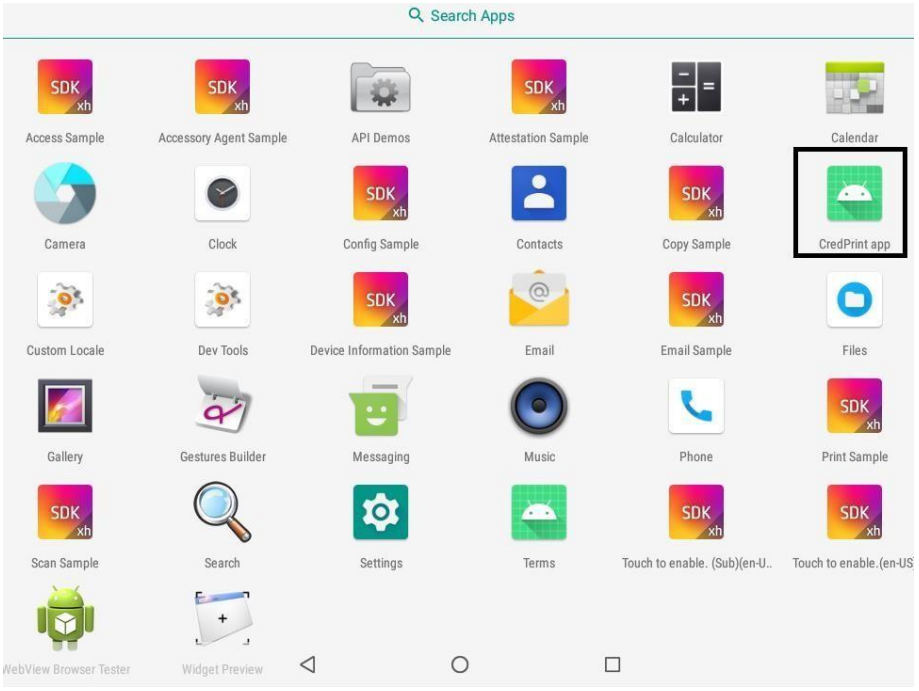

Figura 28 - Emulador Workpath

6. Conclusiones

En este apartado detallar las conclusiones a las que como autor de este TFG he llegado tras la realización de este proyecto. Estas conclusiones se basan en los resultados obtenidos, el aprendizaje, la tarea de investigación y el grado de dificultad que el proyecto me ha supuesto.

6.1 Conclusiones técnicas

Para comenzar, el proyecto contaba con cuatro objetivos que se debían cumplir satisfactoriamente para obtener la plataforma descrita. El primer objetivo era conseguir escanear la documentación de un usuario que usara la impresora.

El siguiente objetivo era la recogida de datos personales de un usuario que usara la aplicación en la impresora comprobando que fueran correctos.

Otro objetivo del sistema era adjuntar tanto los datos personales del usuario como la documentación escaneada y enviarla por email a alguien en recepción que imprima las credenciales.

Como último objetivo del sistema se estableció que algunos aspectos fueran configurables sin necesidad de tener que cambiar el código, siendo estos el mail usado para enviar correos, la dirección a la que enviar los mails o términos y condiciones de la aplicación.

Estos cuatro objetivos se consiguieron satisfactoriamente, luego una vez terminado el proyecto se puede decir que se han superado los requisitos funcionales básicos propuestos.

6.2 Conclusiones personales

La realización de este proyecto me ha servido para aprender cómo trabajan en una empresa de gran renombre como HP. Además, he aprendido a cómo usar una versión privada del Workpath SDK que ofrece HP para la realización de software específico para sus MFPs, lo cual al no ser open source hacia más complicado la búsqueda de información.

Quiero añadir que la realización de este proyecto me ha permitido ver que las impresoras tienen más funcionalidades que imprimir, ya que desconocía la existencia de las MFP y sus posibles usos en el mundo empresarial.

En lo que respecta de la complejidad a nivel técnico a la finalización de este proyecto se ve que no era un proyecto para nada complejo. Sin embargo, cuando elegí este proyecto las impresiones que tenía sobre la complejidad del proyecto eran mucho mayores.

7 Líneas de trabajo futuro

Durante la realización de este proyecto se ha desarrollado una aplicación perfectamente funcional para la agilización de recepción en eventos o instalaciones. A partir de los objetivos conseguidos en esta app se pueden mejorar algunos aspectos de la aplicación.

Aunque, el cliente solicitó los criterios de configuración establecidos en la aplicación, una de las posibles líneas de trabajo futuro podría ser añadir más capacidades de configuración a través del servicio config o scan que el Workpath nos brinda como escanear directamente a un USB.

También se podría añadir una funcionalidad que nos permita recortar o acercar el documento escaneado para una mejor visualización o añadir o quitar campos como datos de usuario.

Otra línea de trabajo futuro podría ser la opción de cambiar el idioma de la aplicación.

8 Bibliografía

https://es.wikipedia.org/wiki/Patrones_de_arquitectura

https://es.wikipedia.org/wiki/Requisito_funcional

https://es.wikipedia.org/wiki/Proceso_unificado

https://es.wikipedia.org/wiki/Modelado_del_software

Modelo de dominio - Wikipedia, la enciclopedia libre

https://developers.hp.com/Workpath-SDK/how-it-works

https://developers.hp.com/Workpath-SDK/remote-configuration

https://developers.hp.com/Workpath-SDK/compatible-devices

https://developers.hp.com/Workpath-SDK

I - El lenguaje de programacion Java (unam.mx)

https://es.slideshare.net/JessicaSanchezMarin/tcnicas-para-identificar-requisitos-funcionales-y-no-funcionales

https://elvex.ugr.es/idbis/db/docs/design/2-requirements.pdf

https://www.ctr.unican.es/asignaturas/mc_oo/doc/casos_de_uso.pd

http://www.lsi.us.es/docencia/get.php?id=5807

https://www.visual-paradigm.com/scrum/what-is-scrum-master/

https://www.visual-paradigm.com/guide/agile-software-development/what-is-user-story/

https://es.wikipedia.org/wiki/Scrum_(desarrollo_de_software)#Pila_del_producto_(o_product_backlog)

https://es.wikipedia.org/wiki/GitLab

https://developer.Android.com/studio/intro?hl=es-419