Universidad de las Ciencias Informáticas

FACULTAD 1

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informática

Título: Sistema para administración de documentos pdf en android (Sunsetdroid)

Autor: Yesenia de la Caridad Valdés Torres

Tutor: Msc. Yariel Ramos Negrín

Agradecimientos.

A mi tutor por el apoyo que me brindo durante todo este curso. Al tribunal y al oponente por sus oportunas críticas y sugerencias. A amistades como la de Rosalba, Ortelio, Alejandro, a mis compañeras de apartamento Ana Isabel, Arlene y María con las que compartí más que un apartamento. A mi hermano y mi mami, por su apoyo constante, por sus sacrificio y amor a lo largo de estos 23 años y por ser la razón de mi vida y mi felicidad. A toda mi familia en especial a los que conviven conmigo día a día, a mi abuela Isabel, mis tías la gorda, Daymi a mi tío Alfredo, a mis primos, Harold, Heidi, Yandro y la más chiquita de la casa Yandira. A todo mi grupo quienes me acompañaron durante esta instancia en la UCI, en especial a tres de ellos que llegaron a sobrepasar los límites de la amistad convirtiéndose en parte de mi familia Abel, Carlos Yordan, Mariam.

Dedicatoria.

A toda mi familia en especial a mi mami y mi hermano.

Declaración jurada de autoría

Declaro que soy el autor de la presente investigación titulada Sistema para administración de documentos pdf en android. (Sunsetdroid), para optar por el título de Ingeniero en Ciencias Informáticas.

El presente trabajo fue desarrollado en el período 2016-2017.

Finalmente declaro que todo lo anteriormente expuesto se ajusta a la verdad y asumo la responsabilidad moral y jurídica que se derive de este juramento profesional. Autorizo como único autor, a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente declaración jurada de autoría en La Habana a los _ días del mes de _ del año 201_.

Yesenia de la Caridad Valdés Torres

_____

Firma del Autor

Resumen

La gestión y acceso a la documentación por parte de una comunidad de usuarios representa una necesidad para muchas instituciones. La búsqueda exhaustiva sobre documentos constituye una tarea compleja para cualquier sistema de gestión bibliográfica que se disponga a automatizar dicho proceso. La misma resulta imprescindible cuando el volumen de información almacenada se incrementa considerablemente. En la Universidad de las Ciencias Informáticas (UCI) se desarrolló una alternativa que consiste en un entorno colaborativo de administración de archivos en formato pdf, que tiene como objetivo disminuir la redundancia y aumentar la disponibilidad de la bibliografía que poseen los usuarios de la universidad. Sunset debe permitir la búsqueda de documentos, la gestión en una biblioteca personal, la posibilidad de compartirlos con el resto de la comunidad, así como la retroalimentación por medio de comentarios. El incremento masivo del uso de las tecnologías móviles se ha hecho extensivo a los usuarios de la universidad. Por tal motivo se decidió desarrollar una aplicación android para dispositivos móviles que permita gestionar archivos en formatos pdf. La aplicación permite, descargar, eliminar y adicionar el archivo.

Palabras claves: gestión, entorno colaborativo

Índice

Índice de figuras.

Índice de Tablas.

Introducción

En la actualidad las tecnologías de la información y comunicación (TIC) e Internet están provocando el nacimiento de la nueva sociedad basada en el conocimiento. La globalización de la economía, de los mercados, de la información, configura un entorno abierto y sin fronteras. Las integraciones de las TIC en los futuros sistemas educativos posibilitan la formación a lo largo de la vida, el aprendizaje electrónico y la alfabetización en aptitudes para el acceso y uso de la información[1].

La controversia internacional en la actualidad es propiamente sobre la gestión del conocimiento, más bien debería tratarse, en aras de la exactitud, como gestión de la información organizacional interna y externa para la generación de nuevos acervos que influirán en el desarrollo de productos y servicios con alto valor agregado, porque el conocimiento, puede gestionarse solo convertido en información, reunido, procesado, organizado, almacenado y diseminado mediante base de datos, redes de información compartida, comunidades virtuales, entre otros medios de transferencia de datos e información[2].

Se es consciente de la importancia de la biblioteca universitaria en la formación académica, concebida como un espacio informal de aprendizaje donde el estudiante encuentra todas las herramientas que va a manejar a lo largo de su vida universitaria. En la actualidad, el alumno accede desde la biblioteca no sólo a la bibliografía básica recomendada por los profesores, sino también a revistas impresas y electrónicas, materiales audiovisuales y multimedia, diccionarios, enciclopedias y otras obras de consulta en distintos soportes y en línea, por tanto, es imprescindible que la biblioteca informe y forme a los alumnos de estos recursos. [3].

Diversos factores, relacionados más bien con necesidades prácticas y objetivas, como el incremento de los volúmenes de información, los límites humanos para su procesamiento, la diversidad de formatos portadores de contenidos valiosos, así como las dificultades, llevaron a que investigadores, especialistas y técnicos del ámbito bibliotecario, la computación y las telecomunicaciones, asumieran el reto de representar en el medio electrónico los sistemas desarrollados tradicionalmente por las bibliotecas con el propósito de satisfacer las necesidades de información de los usuarios[4].

En Cuba, aunque hasta hace poco tiempo solo existían acercamientos teóricos al tema y algunas prácticas aisladas, se han dado pasos para el desarrollo de las bibliotecas digitales y el acceso a la información para todos, experiencias como Ecured y Infomed así lo evidencian. La digitalización de fuentes primarias de la bibliografía nacional cubana va de la mano de la definitiva automatización de los catálogos de la biblioteca Nacional ‘’ José Martí’’ y el programa de digitalización de las colecciones históricas, aunque para su realización requieren significativos aportes financieros[5].

En la UCI existe una primera alternativa que posibilita encontrar localmente bibliografía de apoyo a la producción de software, a la investigación y la docencia. Sunshine, demostró la aprobada aceptación de estrategias de este tipo para los usuarios del campus; sin embargo, los problemas asociados a la inestabilidad de los servicios que brinda el sitio, así como la falta de opciones y usabilidad del mismo, provocaron el surgimiento de una alternativa en favor de soluciones de esta índole. Sunset, surge con la intención de brindar un servicio de calidad, este permite gestionar archivos en formato pdf, no presenta opciones de lectura, ni sincroniza datos e información con el resto de los usuarios a través de la propia plataforma. Una gran parte de la comunidad universitaria, no tiene acceso a Sunset ya que, todos no cuentan con las posibilidades de tener una PC y los que la tienen, no pueden acceder a este a toda hora ni ha todo momento. El incremento masivo del uso de las tecnologías móviles se ha hecho extensivo a los usuarios de la universidad, teniendo esta plataforma una amplia variedad de resoluciones, y siendo Sunset un sitio no adaptativo es por ello aceptable ofrecer una solución móvil para el entorno colaborativo Sunset.

De acuerdo a la problemática planteada anteriormente emerge el siguiente **problema de investigación** ¿Cómo desarrollar una aplicación para dispositivos móviles que permita gestionar archivos en formato pdf?

La presente investigación enmarca su **objeto de estudio** en la gestión de documentación en formato pdf.

Se establece como **objetivo general** desarrollar una aplicación móvil versión Android para la gestión de archivos en formato pdf.

Campo de acción: gestión de documentación en formato pdf en dispositivos móviles.

Para alcanzar este objetivo generar se definen los siguientes **objetivos específicos**:

- Analizar lo referente a aplicaciones homólogas e identificar las metodologías, herramientas y tecnologías para el desarrollo de la propuesta de solución.
- Diseñar la propuesta de solución.
- Implementar las funcionalidades de la aplicación y realizar las pruebas de software a la propuesta de solución.

Preguntas científicas.

¿Cuáles son los presupuestos científicos que sustentan el desarrollo de aplicaciones android para la gestión de archivos pdf?

¿Cuál es la situación actual del desarrollo de aplicaciones android para la gestión de archivos pdf?

¿Cómo debe estructurarse una aplicación android para lograr que la misma permita gestionar archivos en formato pdf?

¿Qué factibilidad tiene la utilización de la solución desarrollada en la universidad de las Ciencias Informáticas?

Posibles resultados: Una aplicación disponible para dispositivos Android, capaz de gestionar archivos en formato pdf.

Para lograr el cumplimiento del objetivo general planteado se utilizarán los siguientes **métodos de investigación**:

Teóricos:

Analítico-Sintético: Se utiliza para realizar un estudio de los elementos más significativos relacionados con las aplicaciones que gestionen archivos en formato pdf.

Análisis Histórico-Lógico: Se utiliza para el estudio del estado del arte de las aplicaciones móviles que permitan la gestión de archivos en formato pdf.

Modelación: Se utiliza en la realización de los diagramas correspondientes a los modelos de análisis, diseño e implementación de la aplicación capaz de gestionar archivos pdf.

Empíricos

Observación: Se utiliza para analizar el comportamiento de las aplicaciones estudiadas.

Estructura del documento:

Capítulo 1: Fundamentación Teórica. En este capítulo se describen los principales conceptos asociados al problema de investigación planteado. Se hace un estudio sobre los sistemas homólogos al propuesto. Finalmente, se seleccionan las herramientas, tecnologías y metodologías adecuadas para el desarrollo del entorno colaborativo.

Capítulo 2: Propuesta de solución. Se hace énfasis en la evolución de dicha solución a través de las fases iniciales: Inicio y elaboración, de la metodología de desarrollo AUP. Presentando los artefactos generados por la misma.

Capítulo 3: Implementación y Prueba. En este capítulo se da cumplimiento a los planes trazados mediante la implementación de los requisitos funcionales de la propuesta solución y finalmente se examina la calidad del entorno colaborativo a través de pruebas de software.

Capítulo 1: Fundamentación Teórica

En este capítulo se establecen varios conceptos que son fundamentales para la mejor compresión de la investigación. Se realiza un análisis referente a las aplicaciones móviles que presentan características afines a la gestión de archivos pdf. Finalmente se identifican las metodologías y herramientas a utilizar para el desarrollo del entorno colaborativo.

1.1Conceptos asociados al dominio del tema

Aplicación móvil: programa hecho solo para función de teléfonos móviles, Tablet o cualquier dispositivo inteligente. Pueden encontrarse en diferentes plataformas que dependerán del fabricante y desarrollador en cuestión y su distribución se encuentra en manos de tiendas virtuales, en donde pueden ser adquiridas gratuitamente o por una suma de dinero específica[6].

PDF: Documentos de formatos portátiles, son una serie de archivos que son creados y editados en un formato destinado a almacenar diferentes tipos de datos virtuales complejos (imágenes, sonidos, mapas de bit, texto y otros). Es la plataforma para procesar y almacenar información más utilizada en gran parte del mundo, por la sencillez con la que funciona y la calidad que ofrece al consumidor[7].

API: Conjunto de comandos, funciones protocolos informáticos que permiten a los desarrolladores crear programas específicos para ciertos sistemas operativos[8].

TXT: Formato abierto para contener texto plano, o texto sin formato[9].

Epub: Es un formato de documento estándar orientado a la visualización en dispositivos tipo libro electrónico[10].

Doc: Formato de texto plano utilizado principalmente, por el procesador de texto Microsoft Word, presente en el paquete Microsoft Office. Doc es una abreviatura de documento [11].

1.2 Estudio de Sistemas Homólogos

El análisis de las particularidades de los sistemas a fines al propuesto, permite conocer características, que, por su relevancia, pueden ser fundamentales para la propuesta a desarrollar, o servir como objeto de estudio para realizar una selección aceptada de estándares, herramientas y tecnologías a utilizar para su construcción.

Principales alternativas identificadas a nivel Internacional

Cool Reader: No tiene tienda incorporada, en su lugar importa los libros desde el dispositivo o directamente de cualquier tienda externa. Es de gran ayuda, la función de búsqueda simple en la tarjeta MicroSD para incorporar los libros rápidamente. Permite configurar muchos parámetros de pantalla, como el tipo de letra, el tamaño, el color, las negrillas entre otras. Dispone de un modo de lectura diurno/nocturno para que al leer por la noche con la luz apagada no deje los ojos encandilados. Entre sus tipos de archivos soporta los formatos epub, doc, txt y otras más, excepto por el pdf[12].

Aldiko: Puede venir preinstalada en la mayoría de los dispositivos. Tiene una interfaz de funcionamiento simple. También es altamente personalizable mediante tamaños de fuente, iluminación, tipos de fuentes, colores, márgenes, y disposición de las páginas. Admite los formatos, PDF, o Adobe DRM[12].

Moon + Rearder: Aplicación altamente configurable y que admite todo tipo de formatos incluyendo PDF. No tiene tienda integrada, pero si una interfaz muy agradable y pulida. Se puede elegir el tipo de fuente, el tamaño, varios temas de fondo, espaciado, autoscroll, distintos tipos de vista, y mucho más[12].

Universal Book Reader: Aplicación famosa para leer libros electrónicos en Android. Permite tanto leer cualquiera de los libros que se tenga en el teléfono o Tablet (en formato ePUB o PDF)[13].

Fabrik: Tiene soporte completo para Dropbox y Google Drive. Así que se puede guardar los libros en la nube y utilizar este lector y leerlos. Ese apoyo en la nube solo los hace digno de ser llamados uno de los mejores que hay. Tiene soporte para una variedad de formatos de libros electrónicos y algunas características ingeniosas para lectores de libros electrónicos. Es una aplicación gratuita[13].

ShuBook: Viene precargado con las fuentes OPDS donde se puede navegar a través de varios ebook gratuitos en varios idiomas (chino, japonés) no es necesario registrarse. También es compatible con archivos Epub y archivos de Microsoft Office, como .doc, .xls, .ppt y rtf. Su biblioteca se puede ordenar por título o autor, también se pueden buscar palabras en texto, cambiar el tamaño del texto y la fuente, y determinar la alineación del texto, y separaciones entre párrafos, líneas o letras[14].

Stanza: Es una aplicación versátil que cuenta con más de 4 millones de descargas de libros, ya que soporta muchos tipos de archivos incluyendo ePUB, pdf y CBR/CBZ en idiomas distintos del inglés. Se puede insertar un enlace de descarga de un formato de archivo compatible que puede ser descargado y guardado en la biblioteca personal. Con Stanza, se puede cambiar el tema o el papel del libro, utilizar el tema de noche, donde el color texto es de color blanco, mientras que el fondo es negro, o personalizar los colores de fondo y de texto a preferencia[15].

OverDrive Media Console: Aplicación que está conectada a más de 18.000 biblioteca de todo el mundo. Permite importar los formatos ePUB y pdf de los archivos de libros electrónicos. Mientras se navega por la biblioteca, se reciben sugerencias de otros libros similares[15].

24Symbols: Con esta aplicación se puede acceder a gran cantidad de libros tanto de android como de ios. Con este servicio se puede leer libros digitales de miles de género en varios idiomas. La lectura de los libros es a través de streaming, por lo que no se tendrá problemas con incompatibilidades o formatos. Desde la aplicación se pueden crear estanterías, organizar y colocar tus ebooks[15].

Ibooks: Esta aplicación que permite, descargar libros cómodamente colocados en una estantería. Permite leer muestras gratuitas antes de comprar el libro, además tiene una variedad de tipos de letras disponibles y colores de página para facilitar la lectura. Podrás resaltar frases, añadir a favoritos, compartir reflexiones en redes sociales, buscar palabras, ajustar el brillo para encontrar la iluminación perfecta en cada ubicación. Permite abrir en formatos ePub y pdf[15].

Resultados.

A partir del estudio realizado, se concluye que ninguna de las aplicaciones analizadas anteriormente cuenta con todos los requisitos funcionales, que debe cumplir la propuesta a

desarrollar. Estos sistemas son aplicaciones, que nos permiten leer libros electrónicos, otras, guardar libros en la nube, y algunas permiten conectarse con varias bibliotecas. La propuesta a desarrollar además de permitirle al usuario leer un documento antes de descargarlo, debe permitir subir, y eliminar dicho documento, que estas aplicaciones permitan que los documentos que lean o suban a la nube sean de todo tipo de formato, es otro motivo por el cual es necesario la creación de una nueva aplicación ya que la se desea ofrecer a los usuarios del campus solo permite el formato pdf. A demás la mayoría de las aplicaciones descritas anteriormente son de uso privado.

1.3 Metodología de desarrollo de software

Una metodología de desarrollo de software describe un entorno que es usado para organizar, planificar, y dirigir un proceso desarrollo de software. Existen varias metodologías de desarrollo de software, todas contienen algunas etapas básicas del ciclo de desarrollo de software como son la planificación, análisis, diseño, implementación y mantenimiento.

1.3.1 Métodos agiles

El desarrollo ágil comprende a los métodos de ingeniería de software basados en el desarrollo iterativo e incremental, donde los requisitos y soluciones evolucionan mediante la colaboración de todos los involucrados en el proyecto. El autor define a las metodologías agiles como un conjunto heterogéneo de métodos con más o menos reglas, principios, recomendaciones, buenas practicas. Cada uno de estos métodos engloba una filosofía que se puede caracterizar en estos cuatro términos[16]:

- Incremental: Las versiones de software son pequeños con ciclos de desarrollo rápido.
- Cooperación: Existe una estrecha interacción entre el cliente y el equipo desarrollador.
- Sencillo: El método en si es fácil de aprender, modificar y está suficientemente documentado.
- Adaptación: Gran capacidad para reaccionar ante de cambios a todo momento.

Es recomendable usar metodologías agiles cuando:

Cuando se necesite hacer nuevos cambios, los nuevos cambios pueden ser implementados a muy bajo costo debido a la frecuencia de nuevos incrementos que se producen [17].

Tanto los desarrolladores del sistema, así como los clientes encuentran más libertad en el tiempo y las opciones, que si el software estuviera desarrollado en una manera más rígida.

Cuando se asume las necesidades del usuario final no siempre son cambiantes. Los cambios pueden ser discutidos y las características pueden ser afectadas o eliminadas basándose en la realimentación.

Ventajas de las metodologías agiles.

- Satisfacción del cliente por entregas rápidas y continuas del producto.
- Estrecha cooperación entre las personas del negocio y los desarrolladores [18]
- Reducción de costo y tiempos [19]

1.3.1.1 Extreme programming (XP)

La programación externa o Extreme programming (XP) es una metodología de desarrollo de software formulada por Kent Beck, autor 1999 del primer libro sobre la materia Extreme programming Explained: Embarace Change. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementada y coraje para enfrentar los cambios. XP define como especialmente adecuada para proyectos con requisitos imprecisos, muy cambiantes y donde existe un alto riesgo técnico. Las historias de usuarios son la técnica utilizada en XP para especificar los requisitos [20].

Ventaja:

- Da lugar a una programación sumamente organizada.
- Ocasiona eficiencias en el proceso de planificación y prueba.
- Cuenta con una tasa de errores muy pequeña.
- Propicia la satisfacción de la programación.

Desventaja:

- Es recomendable emplearla solo en proyectos a corto plazo.

- En caso de fallar, las comisiones son muy altas.
- Requiere de un rígido ajuste a los principios de XP.
- Puede no siempre ser más fácil que el desarrollo tradicional.

1.3.1.2 SXP

SXP es una metodología compuesta por las metodologías SCRUM y XP que ofrece una estrategia tecnológica, a partir de la introducción de procedimientos agiles que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo, ayudando al líder del proyecto a tener un mejor control del mismo. SXP está especialmente indicada para proyectos pequeños equipos de trabajos, rápido cambio de requisitos o requisitos imprecisos, muy cambiantes, donde existe un alto riesgo técnico y se orienta a una estratega rápida de resultados y una alta flexibilidad[21].

Según [16]algunas de las ventajas de SXP son:

- La metodología estable, el uso de sistemas automatizados para la generalización de algunos artefactos. Y además los recomienda de manera explícita.
- La descripción de una historia de usuario suele ser más sencilla que la da un caso de uso.
- SXP ha sido excelente para equipos pequeños, siempre ha sido recomendado para menos de 20 personas.

1.3.1.3 Proceso unificado ágil (AUP)

AUP[22] fue desarrollado por Scott Ambler en septiembre del 2005. Se caracteriza por estar dirigido por casos de uso, centrado en la arquitectura y por ser iterativo e incremental. Es una forma simplificada de RUP. Describe un desarrollo simple del desarrollo de software usando técnicas y conceptos ágiles. Algunas técnicas por AUP incluyen el desarrollo orientado a pruebas, modelos y gestión de cambios ágiles y refactorización de base datos para mejorar la productividad.

Ventajas.

- El personal sabe lo que está haciendo: no obliga a conocer detalles.
- Simplicidad: apuntes concisos.
- Agilidad: procesos simplificados de RUP.
- Centrarse en actividades de alto valor: esencias para el desarrollo.
- Herramientas independientes: a disposición de usuario.
- Fácil adaptación de ese producto: de fácil acomodo (HTML).

Desventaja.

- Es un producto pesado en relación al RUP.
- Como es un proceso simplificado, muchos desarrolladores eligen trabajar con el RUP, por tener a disposición más detalles en el proceso.

1.3.1.4 Metodología de desarrollo de software AUP versión UCI

La UCI desarrolló una versión de la metodología de desarrollo de software AUP, con el fin de crear una metodología que se adapte al ciclo de vida definido por la actividad productiva de la Universidad. Esta versión decide mantener para el ciclo de vida de los proyectos la fase de inicio, pero modificando el objetivo de la misma, y se unifican las restantes fases de la metodología de desarrollo de software AUP en una sola, denominada Ejecución, y agregándose también una nueva fase denominada Cierre [22]

De acuerdo a las características de las metodologías antes planteadas se decide utilizar AUP versión UCI para guiar el desarrollo de la propuesta de solución.

Tabla 1 Fases de la variante de AUP para la UCI.

Fases AUP	Fases Variación AUP-UCI	Objetivos de las fases(variación AUP-UCI)
Inicio	Inicio	Durante el inicio del proyecto se llevan a cabo las actividades relacionadas con la planeación del proyecto. En esta fase se realiza un estudio inicial de la organización cliente que permite obtener información fundamental acerca del alcance del proyecto, realizar estimaciones de tiempo, esfuerzo y costo y decidir si este ejecuta o no el proyecto.
Elaboración	Ejecución	En estas fases se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura y el diseño, se implementa y se libera el producto.
Construcción		
Transición		
	Cierre	En esta fase se analizan tanto los resultados del proyecto como la ejecución y se realizan las actividades formales de cierre del proyecto.

1.3.2.5 Disciplinas de AUP

AUP define 7 disciplinas (4 ingenieriles y 3 de gestión de proyectos): Modelo, Implementación, Prueba, Despliegue, Gestión de configuración, Gestión de proyectos y Entorno. Para el ciclo de vida de los proyectos de la UCI se definen 7 disciplinas también, pero a un nivel más atómico. Los flujos de trabajos: Modelado de negocio, Requisitos y Análisis y diseño en AUP están unidos en la disciplina Modelo, en la variación para la UCI se consideran a cada uno de ellos disciplinas. Se mantiene la disciplina Implementación. En el caso de Prueba se desagrega en 3 disciplinas: Pruebas Internas, de Liberación y Aceptación. Las restantes 3 disciplinas de AUP asociadas a la parte de gestión para la variación UCI se cubren con las áreas de procesos que define CMMI DEV v1.3 para el nivel 2.

1.3.2.6 Escenarios para la disciplina de requisitos

Existen tres formas de encapsular los requisitos: Casos de Uso del Sistema (CUS), Historias de usuario (HU) y Descripción de requisitos por proceso (DRP), agrupados en cuatro escenarios condicionados por el Modelado de negocio. Para este escenario en específico no se modela el negocio sólo se puede modelar el sistema a partir de las HU. Es aplicado a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio muy bien definido. Además, el cliente siempre acompañará al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos. Se recomienda en proyectos no muy extensos, pues una HU no debe poseer demasiada información.

1.4 Herramientas y tecnologías

La selección acertada de herramientas y tecnologías para emplear en el proceso de desarrollo, apoyado en el dominio que se tenga de las mismas, es crucial para la obtención en el tiempo de un producto de calidad. Con esa idea presente, se hizo de los recursos a continuación:

1.4.1 Modelado

El lenguaje de modelado **UML** en su versión 5.2, permite comunicar la estructura de sistema complejo, especificar el comportamiento deseado del sistema, comprender lo mejor

que se está construyendo y a la vez descubrir oportunidades de simplificación y reutilización, Se basa en el hecho de que un modelo es la simplificación de la realidad[23]

Visual Paradimg: en su versión 8.0, se selecciona como herramientas CASE de modelado profesional, que utiliza UML para la completa representación de las etapas por las que transita un producto de software. Este permite la realización de una amplia gama de diagramas como: caso de uso, de actividades, de despliegue, entre otros, así como la generación de código fuente desde los mismos y la documentación asociada al proceso que está siendo modelado.

1.4.2 Lenguaje de programación

Java: es un lenguaje de programación poderoso y fácil de aprender. Cuenta con estructuras de datos eficientes y de alto nivel y un enfoque simple pero efectivo a la programación orientada a objeto. Su elegante sintaxis y tipiado dinámico junto con su naturaleza interpretada, hacen de este un lenguaje ideal para scripting y desarrollado rápido de aplicaciones en diversas áreas y sobre la mayoría de las plataformas.

1.4.3 Gestores de Base de Datos

Un Sistema Gestor de Bases de Datos (SGBD) es una colección de programas que permiten a los usuarios crear y mantener una base de datos. Sistema software de propósito general que facilita los procesos de definición, construcción y manipulación de la base de datos para distintas aplicaciones.

- Definición de las bases de datos: especificar tipos de datos, estructuras y restricciones.
- Construcción de las bases de datos: almacenar datos.
- Manipulación de las bases de datos: consultar, actualizar el diseño y generar informes.

Si la base de datos son los datos almacenados el Sistema Gestor de Bases de Datos es el programa o conjunto de programas que gestionan y mantienen consistentes estos datos [24].

MySQL [25]**:**

- Sistema de gestión de Base Dato Relacional.
  - La información se guarda en tablas:
  - Una tabla es una colección de datos relacionados.

- Una tabla consta de columnas (campos) y filas (registros).
- Las tablas se enlazan por relaciones entre columnas.
    - Implementa casi todo el estándar SQL (Structured Query Language).
    - Código abierto.
        - Actualmente de Oracle, que adquirió Sun, que tenía MySQL AB.
- Escalable.
    - Aplicaciones pequeñas y grandes (millones de registros).
- Transacciones, Multiusuario

Eficiente:

- Multihilo.
- Varias técnicas de hash, b-tree.
- Conexión al servidor MySQL con sockests TCP/IP.
    - Esto permite conectarla con casi cualquier plataforma.

1.4.4 Entorno de desarrollo

1.4.4.1Android

Android es un sistema operativo basado en Linux de código abierto, por lo que, cualquier persona puede acceder de forma gratuita. Fue diseñado para dispositivos táctiles como pueden ser tabletas, Smartphone, entre otros.

1.4.4.2 Android-Studio: en su versión 2.1, es un entorno de desarrollo que cuenta con sencillo asistente de configuración utilizado para desarrollar en el lenguaje de programación Java. Provee funcionalidades que permiten una experiencia única y aumenta en gran medida la productividad:

- Permite importar ejemplos de plantillas.
- Permite ver a editar y pre visualizar los diseños de android a través de múltiples tamaños de pantalla, idioma e incluso versiones de API.
- Analizar el rendimiento de las aplicaciones y muchas más herramientas que ayudaran a los desarrolladores a crear con mayor facilidad mejores aplicaciones para Android.

- Renderización en tiempo real.
- Consola de desarrollo: consejo de optimización, ayuda para la traducción, estadísticas de uso.
- Soporte para construcción basada en Gradle.
- Refactorización especifica de android y arreglos rápidos.
- Herramientas Lint para detectar problemas de rendimiento, usabilidad, compatibilidad de versiones, y otros problemas.
- Plantillas para crear diseños comunes de Android y otros componentes.

figura 1 Android Studio

1.4.4.3 Android-SDK (Android Software Development kit [24]), contiene herramientas necesarias y gratuitas para el desarrollo Android en Eclipse. Combinado con el Bundle ADT ante mencionado forman la combinación perfecta para este tipo desarrollo.

El SDK contiene emuladores y un depurador bastante potente. Además, incorpora la herramienta Manager, que permite descargar e incorporar librerías de la versión deseada al proyecto, así como todo tipo de documentación desea.

1.4.5 Tecnología para la creación de servicio

REST

La tecnología REST[26] fue creado enfocado a proporcionar un modelo arquitectónico para construir servicios web escalables. Las claves de esta tecnología ayudan a mejorar el rendimiento, escalabilidad de las interfaces, modificación de los componentes, visibilidad

de los componentes de la aplicación, portabilidad de los componentes, moviendo el código por los datos, y por último la seguridad.

La tecnología REST se basa en las siguientes claves:

- **Protocolo cliente-servidor sin estado**: cada mensaje HTTP contiene la información necesaria para comprender la petición. Como resultado, ni el cliente ni el servidor necesitan recordar ningún estado de las comunicaciones entre mensajes.
- **Conjunto de operaciones que se aplican a todos los recursos de información**: HTTP en si define un conjunto pequeño de operaciones, las más importantes son POST, GET, PUT y DELETE.
- **Sintaxis universal para identificar recursos:** En un sistema REST, cada recurso se puede direccionar únicamente por su URL.
- **Hirpermedios:** La representación de este estado es un sistema REST son típicamente HTML o XML.
- **Sin estado:** cuando se realiza la comunicación entre cliente servidor no se almacena ninguna información entre ellos. Cada petición de cliente contiene toda la información necesaria para realizar la información necesaria para realizar la petición, siendo el cliente el que guarda los datos de la sesión.
- **Cachable:** los clientes pueden cachear las respuestas.
- **En capas:** Un cliente sabe si está conectado directamente al servidor final, o a un intermediario. Los servidores intermediarios pueden mejorar la escalabilidad del sistema, permitiendo el equilibro de carga, proporcionando caches compartidas e implementando la seguridad.

1.5 Conclusiones Parciales

Como parte de desarrollo del presente capítulo se determinaron las siguientes conclusiones parciales

- El estudio de las principales alternativas a la solución propuesta, evidenció que estas aplicaciones no pueden ser utilizadas por los usuarios de la universidad ya que la mayoría son privativas o hay que tener internet a tiempo completo para poder utilizarlas, o por la simple razón que estas no cumplen con todos los requisitos de la solución propuesta.

- El estudio sobre las metodologías, herramientas y lenguajes permitió definir los componentes bases para el desarrollo de la solución, donde se define AUP como metodología de desarrollo y como herramienta CASE Visual Paradigm 8.0 para el modelado de los artefactos del análisis y diseños de la solución.

Capítulo 2

2.1 Introducción

En el presente capítulo se caracteriza la propuesta de solución a través de la definición de los requisitos funcionales, las tareas a realizar durante la implementación y la arquitectura y patrones utilizados en el desarrollo de la aplicación móvil. Además, se presentan los artefactos generados en las primeras fases de desarrollo.

2.2 Propuesta de solución

El presente trabajo propone como solución una aplicación cliente para dispositivos móviles con sistema operativo Android. Dicha aplicación consumirá desde un servidor web, el cual proveerá los servicios necesarios que permita al usuario listar la información de los libros almacenadas en esta. De igual manera, se incluye la funcionalidad de permitir al usuario subir o descargar libros del servidor web.

2.3 Características del sistema

La aplicación móvil debe presentar las siguientes funcionalidades.

- Permitir al usuario subir un documento.
- Permitir al usuario descargar un documento.
- Permitir al usuario listar los documentos.
- Permitir al usuario eliminar documento.

2.4 Modelo Conceptual

Los modelos conceptuales, se entienden en ingeniería de software como un tipo de modelo relativamente poco sofisticado y, por tanto, más simple de comprender. Para lograr favorecer la compresión de las necesidades del usuario y los requisitos del software de la propuesta de solución se construye el siguiente modelo conceptual.

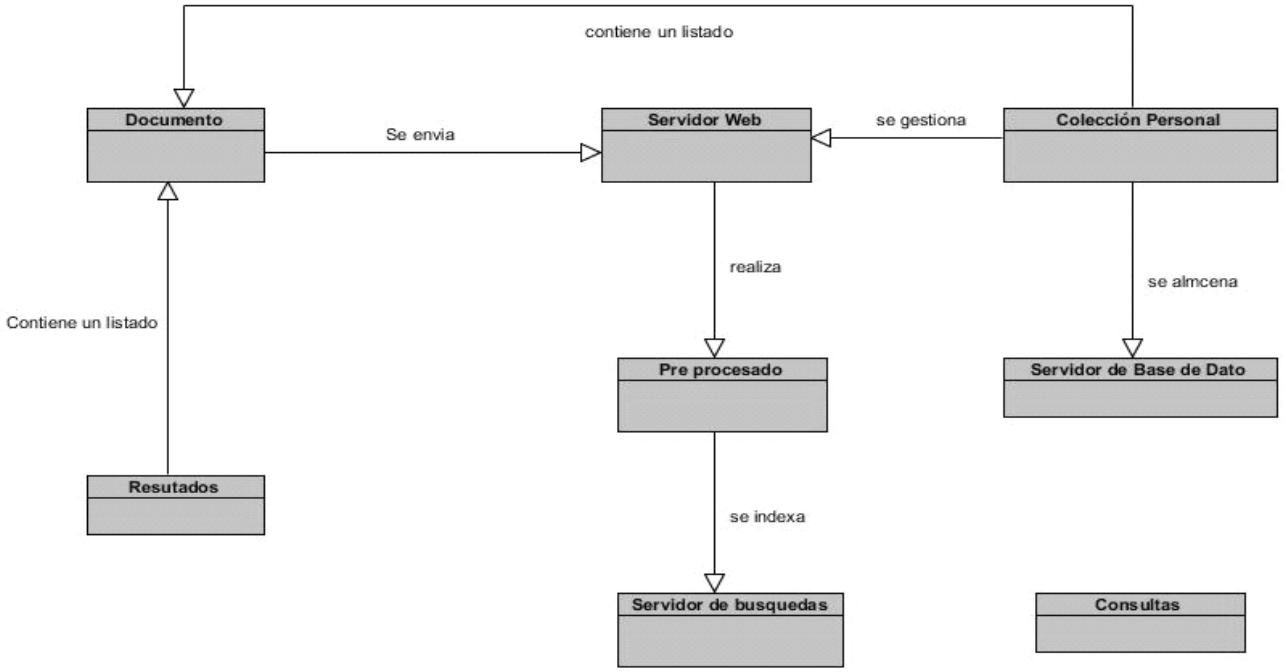

figura 2 Modelo de dominio.

2.5 Requisitos de la propuesta de solución

Como paso fundamental en el proceso de desarrollo de la propuesta de solución se realizará el levantamiento de requisito. Los requisitos definen que es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen. Los requisitos funcionales (RF) son las condiciones que el sistema debe cumplir, mientras que los requisitos no funcionales (RNF) son cualidades que el producto debe tener. A continuación, se reflejan estas necesidades funcionales.

2.5.1 Requisitos funcionales

- Autenticar usuario.
- Subir documento.
- Descargar un documento.
- Listar los documentos.
- Eliminar documento.

2.5.2 Requisitos no funcionales

Interfaz.

- El entorno colaborativo presentara una interfaz agradable y fácil de utilizar.
- El entorno colaborativo tendrá un diseño adaptable para su correcta visualización en dispositivos móviles.

Rendimiento.

- El entorno colaborativo debe ser escalable, permitiendo incorporarles nuevas funcionalidades sin afectar las existentes.

Seguridad.

- Los Errores mostrarán información que no comprometan la seguridad e integralidad del entorno colaborativo.

Software.

- El cliente necesitará un dispositivo con sistema operativo android versión 2.4 o superior.

Hardware.

- El dispositivo debe tener android instalado.
- El dispositivo tiene que tener 512 de RAM o superior.

Operacionales.

- Los documentos subidos a la plataforma estarán en formato PDF.

2.6 Historias de usuario

La metodología AUP utiliza para describir los requisitos de software las historias de usuario, estas utilizan las funcionalidades del sistema como entrada principal para su desarrollo. Contienen la información suficiente para que los desarrolladores puedan producir una

estimación razonable para desarrollar una funcionalidad. A continuación, se describen las historias de usuario de mayor prioridad, las demás se encuentran descritas en el Anexo A.

2.6.1 Historias de usuario

Tabla 2 HU1 Autentificar Usuario

Historia de usuarios	
Numero: HU1	Usuario: Todos
Nombre de Historia: Autentificar usuario	
Prioridad en el negocio: Alta	Riesgo en desarrollo: medio
Puntos estimados:1.5	Iteración asignada
Programador Responsable: Yesenia de la Caridad Valdés Torres	
Descripción: El usuario inserta su identificador (Ejemplo: yctorres) y contraseña en el formulario de autenticación para acceder a las funcionalidades que este autorizo.	
Observación: si el sistema no identifica al usuario le mostrara una notificación sobre el problema.	

Tabla 3 HU2 Subir documento

Historia de usuarios	
Numero: HU2	Usuario: Todos
Nombre de Historia: Subir documento	
Prioridad en el negocio: Alta	Riesgo en desarrollo: Alto
Puntos estimados:1.5	Iteración asignada:1
Programador Responsable: Yesenia de la Caridad Valdés Torres	
Descripción: El usuario activa el cuadro de dialogo para subir el documento, proporciona los datos correspondientes y selecciona el archivo en formato PDF.	
Observación: El sistema debe notificar al usuario cuando la operación termine.	

Tabla 4: HU3 Eliminar documento

Historia de usuarios	
Numero: HU5	Usuario: Todos
Nombre de Historia: Eliminar documento.	
Prioridad en el negocio: Alta	Riesgo en desarrollo: medio

Puntos estimados:1.5	Iteración asignada
Programador Responsable: Yesenia de la Caridad Valdés Torres	
Descripción: El usuario elige la opción eliminar, y se le muestra un mensaje.	
Observación: Si la plataforma se encuentra vacía, se muestra un mensaje (No hay libro que mostrar).	

2.7 Diseño

El papel del diseño en el ciclo de vida un software es facilitar la comprensión de su funcionamiento y proveer una representación o modelo del mismo con el propósito de definirlos con los suficientes detalles como para permitir su realización física. El modelo de diseño provee una representación arquitectónica del software que sirva de punto de partida para las tareas de implementación, dando al traste con los requisitos del sistema.

2.7.1 Arquitectura

El estándar IEEE define la arquitectura de software como la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos el ambiente y los principios que orientan su diseño y evolución [27]Según Roger Pressman: “En su forma más simple, la arquitectura del software es la estructura u organización de los componentes del programa, la manera en que estos interactúan y las estructuras de datos que utilizan” [28]

O sea, la arquitectura de software es una forma de representar sistemas mediante el uso de la abstracción, de forma que aporte el más alto nivel de compresión de los mismos. Esta representación incluye los componentes fundamentales del software, su comportamiento y formas de interacción para satisfacer los requisitos del sistema.

Para el presente trabajo de diploma se propone una arquitectura Modelo-Vista-Contralor (MVC), ya que la aplicación estará basada en la interacción de un cliente con servidor a través de la web y esta arquitectura es ampliamente usada para este propósito, además que es empleada cada vez más en la implementación de aplicaciones Android.

Esta arquitectura separa presentación e iteración de los datos del sistema. El sistema se estructura en tres componentes lógicos que interactúan ente sí. El componente Vista define y gestiona como se presentan esos datos al usuario. El componente Controlador dirige la iteración del usuario y pasa estas interacciones a Vista y Modelo [29]

Permite que los datos cambien de manera independiente de su representación y viceversa. Soporta en diferentes formas la representación de los mismos datos, y los cambios en una representación se muestran en todos ellos.

La solución propuesta constara de tres componentes fundamentales: Modelo, Vista y Controlador. A continuación, se muestra una vista que lo ilustra.

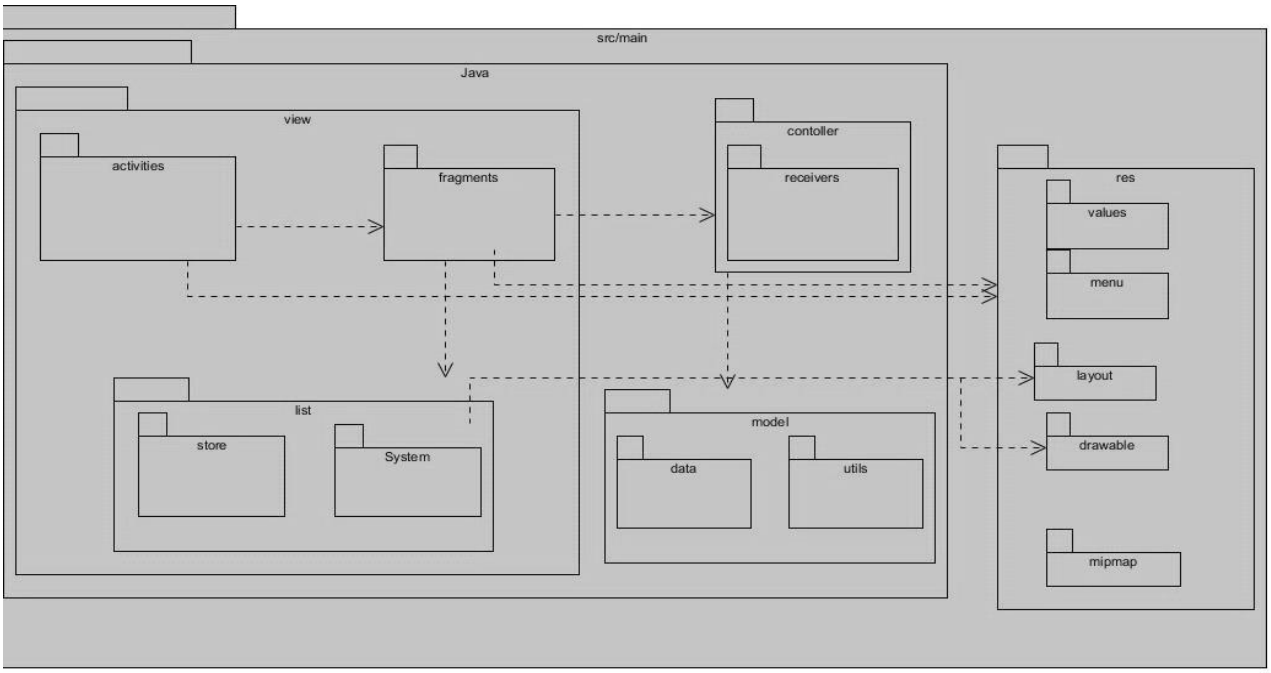

figura 3 Diagrama de paquetes de la propuesta de solución

Modelo: Maneja todo lo referente a la persistencia de datos de la aplicación, las clases entidades, el acceso a la red y los elementos necesarios para manejar dichos elementos (paquete model).

Vista: Representa la interfaz gráfica para la interacción con el usuario. Dentro se ubican todos los componentes que intervienen en la visualización del resultado de la comunicación con el Controlador (paquete view).

Controlador: Aquí se tendrán las clases que interactúan con el componente Vista recibiendo las solicitudes de eventos de los usuarios y con el Modelo registrando los cambios realizados por el mismo (paquete controller).

2.7.2 Patrones de diseño

Un patrón de diseño es una descripción de la comunicación entre objetos y clases, personalizada para resolver un problema de diseño general en un contexto particular. Identifica clases, instancias, roles, colaboraciones y la distribución de responsabilidades

[30].Se presenta como pares de problema-solución con nombre, sugiriendo aspectos relacionados con la asignación de responsabilidades.

Los patrones de diseños se caracterizan por:

- Representar soluciones técnicas a problemas concretos.
- Propiciar la reutilización.
- Representar problemas frecuentes.

2.7.2.1 Patrones GRASP

Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos[31]. El nombre se eligió para indicar la importancia de captar estos principios, si quiere diseñar un software de manera eficaz.

- **Experto:** Se usa más que cualquier otro al asignar responsabilidades, es un principio básico que suele utilizarse en el diseño orientado a objeto[31]. Consiste en la asignación de una responsabilidad a la clase que cuenta con la información necesaria para llevarla a cabo. El uso de este patrón da pie a un bajo acoplamiento y una alta cohesión que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento. El cumplimiento de una responsabilidad requiere a menudo información distribuida en varias clases de objetos.
- **Bajo Acoplamiento:** El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras [31].El bajo acoplamiento soporta el diseño de clases más independientes y reutilizables, lo cual reduce el impacto de los cambios y acrecienta la oportunidad de una mayor productividad. Ejemplo del uso de este patrón en la solución propuesta se muestra en las clases Store, Útil e installedApplication, donde se minimizan las relaciones de estas con el resto de las clases.
- **Alta Cohesión:** En la perspectiva del diseño orientado a objetos, la cohesión (o más exactamente, la cohesión funcional) es una media de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades modernas en un área funcional que colaboran con las otras para llevar a cabo las tareas [31].En el diseño de la solución se

propone, se evidencia este patrón en la clase System, cuya funcionalidad getinsDate de la clase InstalledApplication.

- **Controlador:** Un controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un ventó del sistema[31]. En esta solución propuesta se evidencia en las clases Store y System, las cuales se encargan de gestionar los procesos involucrados en la aplicación separándolos en los que tienen que ver con los libros almacenados en el API.

2.7.2.2 Patrones GOF

Los patrones GOF son alternativas de solución a problemas conocidos, pero son mucho más específicas las soluciones donde se aplican. Se clasifican en creacionales, estructurales y de comportamiento[30]. A continuación, se enuncian brevemente los utilizados en el sistema.

- **Singleton:** Está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intensión consiste en garantizar que una clase solo tenga la instancia y proporcionar un punto de acceso global a ella[30]. Se empleó en la creación de las clases Util, BackgroundTask y DBManeger para garantizar para evitar que no exista duplicación de los objetos.

2.7.3 Diagrama de Clases de la propuesta de solución

Un diagrama de clases de diseño (DCD) representa las especificaciones de las clases e interfaces software en una aplicación. Entre la información general encontramos:

- Clases, asociaciones y atributos.
- Interfaces, con sus operaciones y contantes.
- Métodos.
- Información acerca del tipo de los atributos.
- Navegabilidad.
- Dependencias.

A diferencia de las clases conceptuales del Modelo del dominio, las clases de diseño de los DCD muestran las definiciones de las clases de software en lugar de los conceptos del

mundo real[31]. A continuación, se muestra el Diagrama de Clases de Diseño de la propuesta de solución.

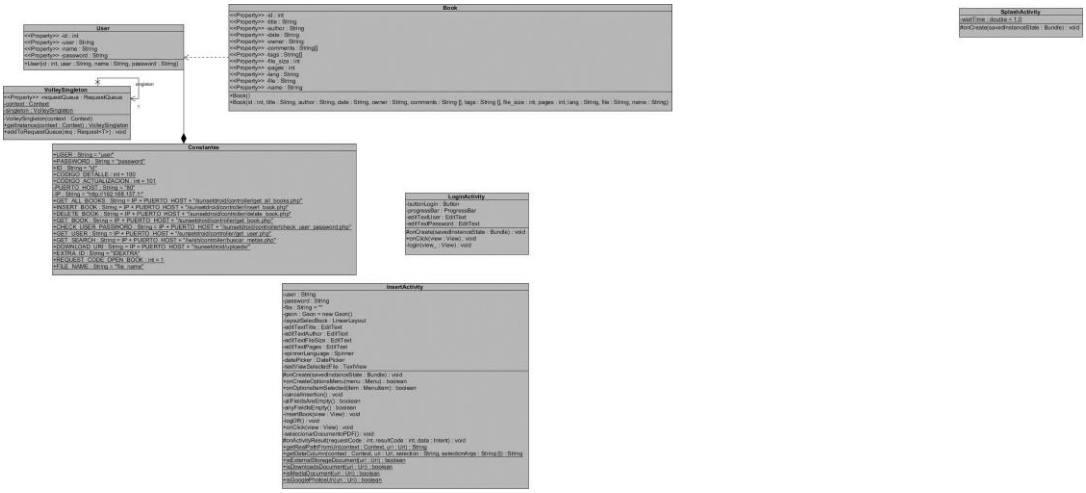

figura 4 Diagrama de clases.

2.7.4 Modelo de datos entidad-relación

El modelo de datos Entidad-Relación (E-R) está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades y de relaciones entre objetos. Una entidad es una cosa <<cosa>> u <<objeto>> en el mundo real que se distingue de otros objetos. Las entidades se describen en una base de datos mediante un conjunto de atributos[32]. A continuación, se muestra el Modelo de datos de la propuesta de solución.

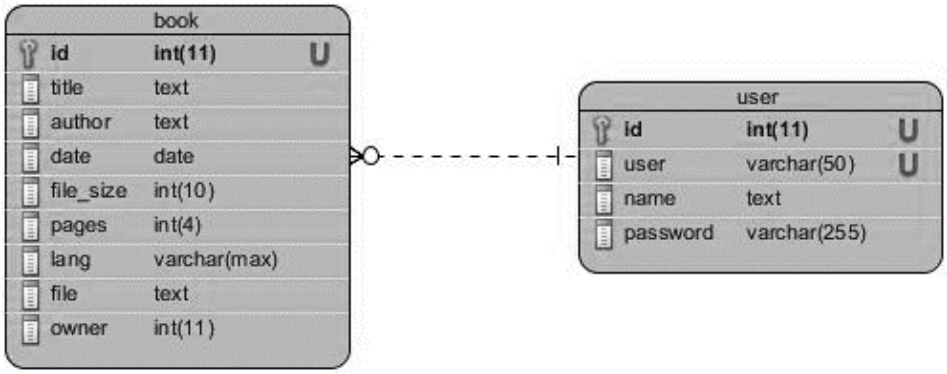

figura 5 Modelo de la Base de Dato de la propuesta de solución.

Capítulo 3. Implementación y prueba

3.1 Introducción

Con el objetivo de materializar la solución propuesta y cumplir con los requisitos obtenidos al inicio de la investigación, se lleva a cabo la fase de implementación y prueba. En ella quedan recogidas las especificaciones asociadas a la implementación de la aplicación. Se describen las pautas de codificación utilizadas y el diagrama de despliegue como resultado de la implementación. Al término de esta, la aplicación resultante será sometida a un proceso de pruebas con el objetivo de validar el cumplimiento de los requerimientos especificados anteriormente.

3.2 Tareas de ingeniería

Iteración 1	
Historia de Usuario	Tareas
Autenticar Usuario	• Definir interfaz de autentificación. • Validar los datos introducidos por los usuarios (nombre de Usuario y contraseña). • Notificar al usuario si los datos son incorrectos.
Subir documento	• Definir interfaz para el formulario de subida de documento.

	• Validar los datos introducidos por el usuario. • Notificar al usuario al finalizar el proceso.

Las tareas de la ingeniería representan el trabajo realizado durante una iteración expresada en tareas de la programación. Debido a que las historias de usuarios no brindan el suficiente nivel de detalle para llevar a cabo la implementación, las tareas de ingeniería juegan un papel fundamental al indicar a los programadores las acciones a realizar por cada una de ellas, donde cada tarea es asignada a un programador directamente. A continuación, se muestran las tareas de ingeniería implicadas en la iteración 1.

3.3 Implementación

La codificación de la solución propuesta tiene lugar una vez que se ha definido las historias de usuarios y se ha concluido el diseño de la aplicación. Está encaminada a desarrollar de forma iterativa e incremental un producto completo listo para el despliegue, obteniendo versiones útiles de forma rápida, las que paulatinamente completan el desarrollo de la aplicación.

3.3.1 Estándares de codificación

Los estándares de codificación permiten un mejor entendimiento del código por parte de todos los miembros del equipo de desarrollo y en consecuencia hacen que el código sea fácil de mantener. El uso de estándares de codificación trae consigo los siguientes beneficios:

- Facilita el mantenimiento de una aplicación.
- Permiten que cualquier programador entienda y pueda mantener la aplicación.
- Mejoran la legibilidad del código, al mismo tiempo que permiten su rápida compresión [33]

3.4 Diagrama de componente

Los diagramas de componentes se utilizan con el propósito de modelar la vista estática de un software, o sea como un sistema se encuentra dividido y qué relaciones de dependencias existen entre dichas partes que lo conforman. A continuación, se expone el diagrama definido para la aplicación Sunsetdroid.

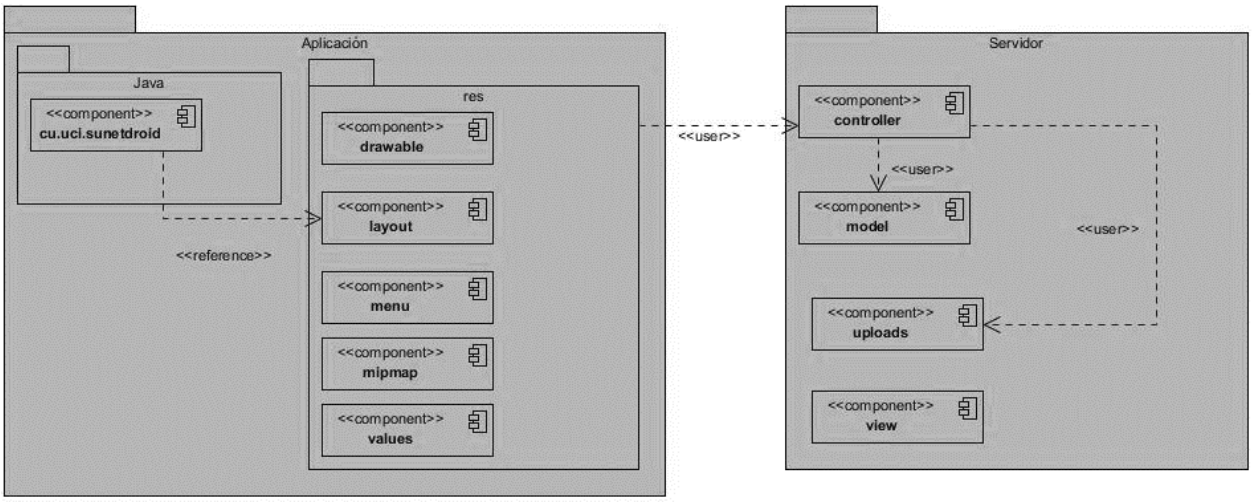

figura 6 Diagrama de componentes

Los componentes que intervienen en la aplicación quedan recogidos en dos grupos: Aplicación y Servidor. El primero agrupa todas las funcionalidades que se ejecutan del lado del cliente y la segunda del lado del servidor.

3.4.1 Aplicación

- En la carpeta cu.uci. sunsetdroid se encuentran todas las clases implementadas en la aplicación. En estas clases se implementan todos los métodos necesarios para implementar la aplicación.
- En la carpeta drawable se encuentran todos los archivos xml que contienen las configuraciones de las imágenes empleadas en la aplicación.
- En la carpeta layout se encuentran todas las activity de la aplicación, o sea las interfaces de usuario.
- En la carpeta menú se encuentran los archivos xml de los activity que contienen menú.
- En la carpeta mipmap se encuentran las imágenes utilizadas en la aplicación.

3.4.2 Servidor

- En la carpeta controller se encuentran todos los archivos php donde se implementan todos los métodos necesarios para crear el servicio.
- En la carpeta model se encuentran los archivos php que trabajan con la base de dato.
- En la carpeta ulploads se encuentran todos los archivos en formato pdf subidos por el usuario.

3.5 Diagrama de despliegue:

Un diagrama de despliegue muestra las relaciones físicas que se establecen entre componentes de software y hardware dentro de un sistema de cómputo determinado. Se modela a través de un conjunto de nodos las relaciones existentes entre ellos. Cada nodo en este tipo de diagrama representa un tipo de unidad computacional, en la mayoría de los casos de tipo hardware[34].

A continuación, se presenta el diagrama de despliegue de la propuesta de solución:

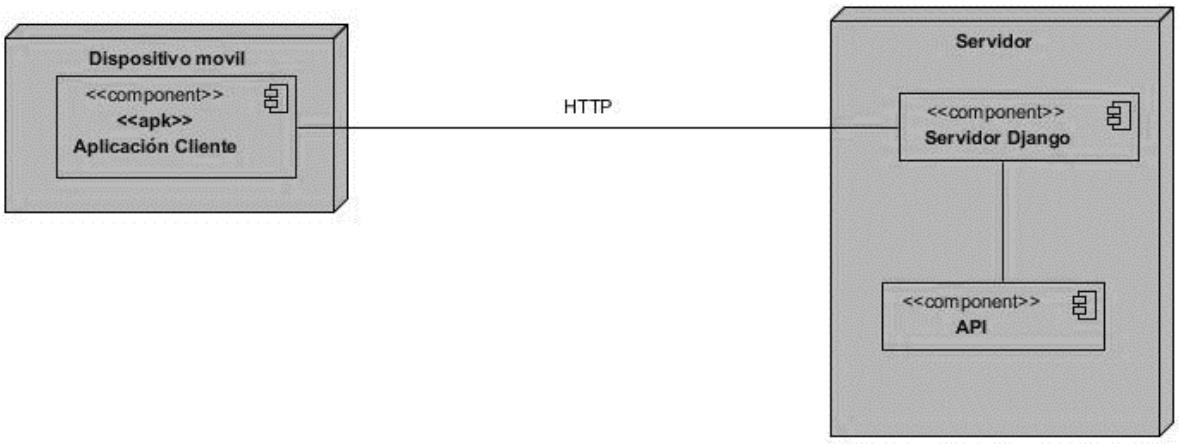

figura 7 Diagrama de despliegue.

3.6 Interfaz Gráfica

La interfaz de usuario es el medio con el que el mismo puede comunicarse con el móvil y comprende todos los puntos de contacto entre la persona y el equipo. Normalmente suelen

ser fáciles de entender y fáciles de accionar. A continuación, se muestran las principales interfaces de la aplicación de administración de documentos digitales en formatos pdf.

3.6.1 Splash

Cuando el usuario ejecute la aplicación por unos instantes se le mostrará una pantalla con el logo de la aplicación.

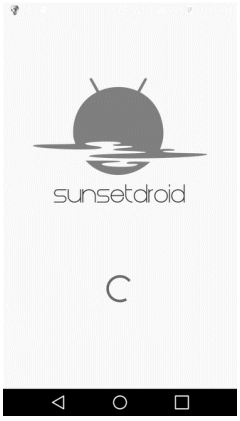

figura 8 Splash de la aplicación.

3.6.1 Autenticar usuario

Cuando un usuario desea entrar a la aplicación se encuentra con la interfaz de inicio donde se encuentra con un formulario de dos campos, donde le pide a este usuario y contraseña.

figura 9. Interfaz de autenticar usuario

Si el usuario intenta continuar sin rellenar los campos, se les mostrará un mensaje avisando que se debe rellenar los campos antes de continuar.

figura 10 Interfaz rellene los campos para continuar

Si el usuario introduce un usuario o una contraseña incorrecto, se mostrará un mensaje notificándolo.

figura 11 Interfaz de usuario o contraseña incorrectos

Si cuando el usuario intente autenticarse no hay conexión, se mostrará un mensaje notificándolo.

figura 12 Interfaz Conexión no disponible.

3.6.2 Subir Documento

Cuando un usuario desea añadir un libro a la plataforma, activa el formulario donde quedan recogidas todos los datos del mismo: formato en PDF, título, autores, etiquetas e idiomas. Dichos archivos pueden proceder de disímiles fuentes y en ocasiones el nombre del mismo no coincide con el título del libro. Por esta razón, el usuario debe proveer al menos el título y el idioma en que fue escrito.

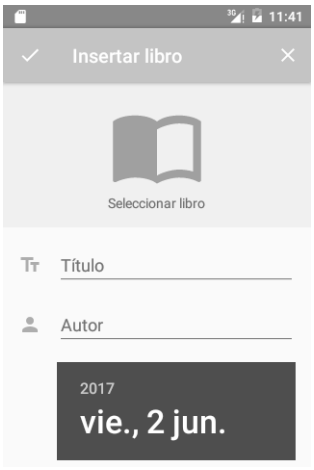

Figura 13Interzas de usuario subir documento.

3.6.3 Listar libros

Cuando el usuario pulse el botón de listar libro se le mostrará una lista de los libros existentes en Sunset, de estos se muestra el título, el autor, el tamaño y la fecha. Además, se mostrará un botón eliminar por cada elemento existente en la lista y un botón adicionar, por si se desea subir algún libro a la colección.

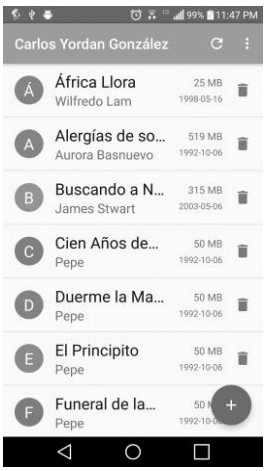

figura 14 Listado de libros

Si la lista se encuentra vacía se mostrará un mensaje notificándolo y además se contará con un botón para adicionar algún libro.

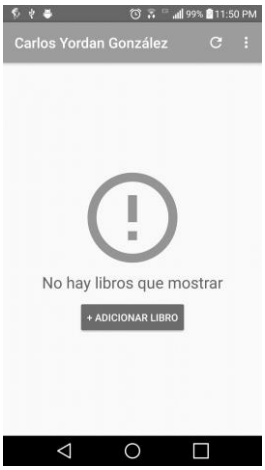

figura 15 No hay libros que mostrar

3.7 Pruebas de software

La prueba de software es un elemento crítico para la garantía de la calidad de software y representa una revisión de las especificaciones, del diseño y la codificación. Una vez generado el código fuente es necesario probar el software para descubrir y corregir la mayor cantidad de errores posibles antes de ser entregados. Su objetivo es diseñar una serie de casos de prueba que tengan una alta probabilidad de encontrar errores [28].

3.7.1 Estrategia de prueba

Una estrategia de prueba software proporciona una guía que describe los pasos que deben realizarse como parte de la prueba, cuando se plantean y se lleva a cabo dichos pasos, y en cuanto esfuerzo, tiempo y recursos requerían. Por tanto, cualquier estrategia de prueba debe incorporar la planificación de la prueba, el diseño de casos de pruebas, la ejecución de la prueba y la recolección y evaluación de los resultados [28].

Las pruebas se dividen en los siguientes niveles principales: pruebas de unidad, de integración, de validación, de sistema y de aceptación [28]. Para la solución propuesta se utilizó la estrategia de pruebas basadas en la ejecución de las mismas tomando como guía dos de estos niveles: unidad, validación.

3.7.1.1 Prueba de Unidad

Las pruebas de unidad y unitarias son el proceso de probar componentes del programa tales como métodos o clases de objetos. Las funciones o los métodos individuales son el tipo más simple de componente. Las pruebas deben llamarse para dichas rutinas con diferentes parámetros de entrada [29].

El método usado para esta prueba fue el de caja blanca, donde las pruebas se enfocan en la estructura de control del programa. Los casos de prueba se derivan para asegurar que todos los enunciados en el programa se ejecutaron al menos una vez durante las pruebas y que todas las condiciones lógicas se revisaron [28].

3.7.1.2 Pruebas de Validación

La prueba de validación proporciona un aseguramiento fino de que el software con todos los requisitos funcionales, de comportamiento y desempeño. La prueba se concentra en las acciones visibles para el usuario y en la salida que puede reconocer. La validación se alcanza cuando el software funciona de tal manera que satisface las expectativas razonables (especificación de requisitos de software) del cliente. Se logra mediante una serie de pruebas que demuestran que se cumplen con los requisitos [28].

Para realizar estas pruebas se hace necesario emplear pruebas funcionales, ya que se aseguran el apropiado trabajo de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. Las metas de estas pruebas son verificar la apropiada aceptación de datos y verificar el procesamiento, recuperación e implementación adecuada de las reglas de negocio [28].

Para llevarlas a cabo, el método a emplear fue el de caja negra, el cual se centra en los requisitos funcionales del software. Es decir, la prueba de la caja negra permite al ingeniero de software obtener conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa [28].

Como técnica se utilizó la de partición de equivalencia, o sea, dividir el dominio de entrada de un programa en clases de datos a partir de las cuales puedan derivarse casos de prueba. El diseño de casos de prueba para participación equivalente se basa en una evaluación de las clases de equivalencia para una condición de [28].

Los casos de prueba se diseñaron según las funcionalidades descritas en las historias de usuario. La intención que se persigue con los artefactos es lograr una compresión específica de las condiciones que la solución que debe cumplir. Cada planilla de casos de pruebas recoge la especificación de una historia de usuario, dividida en secciones y escenarios, detallando las funcionalidades descritas en ella y describiendo cada variable.

A continuación, se muestran los casos de pruebas correspondientes a las historias de usuarios “Subir documento” y Listar documento” ya que representan el proceso fundamental del negocio. Los casos de prueba asociados al resto de las historias de usuarios son definidos en el Expediente de Proyecto.

Abreviaturas utilizadas:

CP: Caso de Prueba.

HU: Historia de Usuario.

EC: Escenario.

Tabla 5 Caso de prueba de Validación para la Historia de Usuario Autenticar usuario.

		Caso de Prueba de Validación.	
CP_1 HU1		HU1:Autenticar usuario.	
Responsables: Yesenia de la Caridad Valdés Torres			
Descripción: El caso de prueba se inicia al ejecutarse la aplicación. Se presenta al usuario una interfaz con dos campos, el primero para el usuario y el segundo para la contraseña. El caso de prueba termina cuando el usuario termina la ejecución de la aplicación.			
Condiciones de ejecución: Debe existir al menos un usuario registrado en la aplicación.			
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**

EC1.1 Autenticar Usuario	El usuario ejecuta la aplicación y esta automáticamente mostrándole al usuario la interfaz de autenticación. Este escribe usuario y contraseña, el sistema verifica si estos son correctos.	Se muestra un listado de todos los libros existentes en la plataforma Sunset. De cada libro se muestra el autor, el tamaño del libro, título, fecha, idioma.	Se inicia la aplicación.
EC1.2 Error de conexión con la plataforma Sunset.	El usuario ejecuta la aplicación y esta no logra conectare a la plataforma de Sunset.	La aplicación muestras al usuario un mensaje de error indicando que no ha podido conectarse y automáticamente muestra el listado de los libros del usuario en cuestión.	Se inicia la aplicación.

Tabla 6 Caso de prueba de Validación para la Historia de Usuario Descargar Libro

CP_2 HU3	HU4:Autenticar usuario.
Responsables: Yesenia de la Caridad Valdés Torres.	
Descripción: El caso de prueba se inicia luego de ejecutarse la aplicación. Para cada libro presente en el listado de libros de la plataforma Sunset se muestran las siguientes acciones disponibles: Descargar, adicionar. El usuario escoge la acción descargar de uno en específico. El caso de prueba termina cuando la descarga ha finalizado.	

Condiciones de ejecución: Debe existir al menos un libro almacenado en la plataforma Sunset.			
Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC3.1 Descargar un libro de la lista de libros de la plataforma Sunset.	El usuario selecciona la acción Descargar de un libro de la lista de libro de plataforma Sunset y descarga dicho libro al dispositivo móvil.	La aplicación muestra al usuario un mensaje para indicar el inicio de la descarga de la plataforma Sunset. Luego informará acerca del proceso de la descarga. Finalmente mostrará un mensaje indicando	El usuario selecciona la acción descargar libro.
EC3.2 Error en la descarga de un libro de la plataforma Sunset.	El usuario selecciona la acción descargar de un libro de la plataforma Sunset y no se puede completar la descarga desde la plataforma Sunset.	La aplicación muestras al usuario un mensaje de error indicando que ha fallado la descarga y la causa por la cual no se ha podido completar la acción.	El usuario selecciona la acción descargar libro.

3.7.2 Ejecución y resultados de las pruebas de software

3.7.2.1 Validación

Para la ejecución de la prueba de validación de tipo alfa se utilizaron la siguiente terminal:

Emulador de android:

- Versión de SO Android 6.0 (Marshmallow).
- SDK 23
- CPU ARMv7 Processor rev 4 (v7l)
- RAM 2.0 GB
- Almacenamiento interno 196,3 Mb
- Almacenamiento externo 500 Mb

A partir de la interpretación de dichos resultados se puede afirmar que los mismos garantizan el correcto funcionamiento de los métodos de las clases DetailActivity y InsertActivity, los cuales constituyen las unidades comprobables más pequeñas con que consta la solución implementada.

Los problemas detectados en el período de pruebas de validación se clasificaron en: No Conformidades Significativas (NCS) y en No Conformidades no Significativas (NCNS). A continuación, se describen los aspectos que se tuvieron en cuenta en cada clasificación.

NCS: son las no conformidades referentes a las funcionalidades de la aplicación: validaciones incorrectas o respuestas diferentes a lo descrito previamente en las historias de usuarios.

NCNS: Son las no conformidades en cuanto al diseño de la propuesta de solución y errores ortográficos.

Fueron realizadas 3 iteraciones de pruebas, ejecutándose al término de cada una de ellas pruebas de regresión, con el objetivo de asegurar que al resolverse las no conformidades detectadas estas no introdujeran nuevos errores en la solución.

En la primera iteración se detectaron **6NCS** y **3NCNS**. Las mismas fueron resueltas satisfactoriamente en la misma iteración.

No conformidades Significativas:

1. La aplicación se detuvo al tratar de acceder al servidor sin tener conexión.
2. La aplicación mostró la información de un libro almacenado en la plataforma Sunset distinta a la selecciona en el listado correspondiente de la interfaz principal.
3. La aplicación no mostró una actividad con información acerca de los libros al dar clic sobre el libro.
4. La aplicación no mostró ningún resultado al ejecutar una búsqueda en el listado de libros almacenados en la plataforma Sunset, conociéndose que existen libros que coinciden con el criterio de búsqueda introducido.
5. La aplicación no actualizó el listado de libros existentes en la plataforma Sunset al eliminarse un libro.

No Conformidades No Significativas:

1. La aplicación tiene falta de ortografía en los nombres de algunas de las categorías presentes en el menú de despliegue.
2. La aplicación no muestra completamente el texto de las acciones a realizar que presenta cada elemento de la lista de libros de la plataforma Sunset.
3. La aplicación muestra el nombre y el título de un libro del listado de libros de la plataforma Sunset sin un espacio entre ambos textos.

En la segunda iteración se detectaron 1 NCS y 1 NCNS, siendo solucionadas de la misma forma de las anteriores.

No Conformidades Significativas:

1. La aplicación no mostró un mensaje de error cuando no pudo conectarse al servidor.

No Conformidades No Significativas:

1. La aplicación no muestra el nombre del usuario que se autentica.

En la tercera iteración no se detectaron NCS ni NCNS, por lo que se demostró que la aplicación cumple con los requisitos funcionales establecidos y fue considerada concluida. A continuación, se muestra un gráfico con resumen de los resultados obtenidos tras la realización de las pruebas:

figura 16 Gráfica correspondiente a las no conformidades.

3.7.3.2 Unitaria

La prueba unitaria se le realizó al método seleccionarDocumentoPDF () de la clase InsertActivity.

Los problemas detectados en los períodos de pruebas de la prueba unitaria fueron:

En una primera Iteración: cuando el usuario selecciona un archivo pdf este le dice que este formato no es aceptado, por lo que el método no hace bien la comprobación del tipo de formato. Este error fue corregido en la propia iteración, por lo que en una segunda iteración no se encuentran errores.

3.8 Conclusiones parciales

A partir del desarrollo del presente capítulo se arribaron a las siguientes conclusiones: La descripción del proceso de implementación de la aplicación, a través de la definición de las convenciones utilizadas para la codificación, posibilitó una mejor legibilidad del código, haciéndolo más comprensible y estandarizado.

A través del diagrama de despliegue se obtuvo la especificación de las características que debe cumplir la aplicación correcto funcionamiento, así como la distribución final que tendrá.

Las pruebas realizadas, permitieron comprobar el correcto funcionamiento del código de la aplicación, validar la completitud de los requisitos.

Conclusiones

Con este proyecto se ha logrado profundizar en la programación con Android enfrentándonos con todo el proceso de realización de una aplicación para móvil. Se ha intentado implementar las últimas funcionalidades y tecnología en el ámbito de Android usando la versión de la API más moderna.

En este desarrollo se ha conseguido un diseño robusto y sencillo de usar, con las opciones de usabilidad necesarias para que el sistema sea lo más sencillo posible, pero con la potencia suficiente para realizar todas las operaciones necesarias.

Por tanto, se puede decir que el desarrollo cumple todas las expectativas y resuelve los problemas que se planteaban a la hora de realizar el proyecto.

Recomendaciones

- Añadir a la aplicación la funcionalidad, crear colección, así el usuario solo tendrá permiso para eliminar los documentos subidos por el mismo.
- Anadir a la aplicación la funcionalidad búsqueda.

Bibliografía:

1. DOMÍNGUEZ AROCA, Isabel. La biblioteca universitaria ante el nuevo modelo de aprendizaje: docentes y bibliotecarios, aprendamos juntos porque trabajamos juntos. S.l.: s.n.

2. ZAMORA FONSECA, Raquel. La Biblioteca Virtual: Reflexiones y consideraciones teóricas. S.l.: s.n.

3. ALFARO TORRES, Paloma. La importancia de la biblioteca universitaria en la formación académica. S.l.: s.n.

4. CABRERA FACUNDO, Ana Margarita. Las bibliotecas digitales. Parte I. Consideraciones teóricas. S.l.: s.n.

5. CABRERA FAGUNDO, Ana Margarita. Revista Cubana de Información en Ciencias de la Salud. En: .

6. La aplicación. En: [en línea]. Disponible desde: http://appdesignbook.com/es.

7. Concepto de pdf. En: [en línea]. Disponible desde: http://conceptodefinicion.de/pdf/.

8. La tecnología. En: [en línea]. Disponible desde: http://www.abc.es/tecnologia/.

9. Apuntes y monografía. En: http://www.taringa.net/.

10. Emprendelo. En: [en línea]. Disponible desde: http://www.emprendelo.es.

11. alegsa. En: [en línea]. Disponible desde: http://www.alegsa.com.ar/Dic/doc.php.

12. JULIAN, marquina. 16-aplicaciones-para-leer-libros-en-tus-dispositivos-moviles. S.l.: s.n.

13. 3-aplicaciones-leer-libros-electronicos-android. [en línea]. S.l.: s.n. Disponible desde: 17. https://blogs.harvard.edu/allgadgetreviews/3-aplicaciones-leer-libros-electronicos-android/.

14. aplicaciones-moviles-para-leer-libros [en línea]. S.l.: s.n. Disponible desde: 21. http://dexpierta.com/aplicaciones-moviles-para-leer-libros/.

15. top-10-gratis-ibooks-alternativas-lectura-apps-para-sus-dispositivos-ios. [en línea]. S.l.: s.n. Disponible desde: 22. http://descargarsnapchatapp.com/top-10-gratis-ibooks-alternativas-lectura-apps-para-sus-dispositivos-ios/.

16. PEÑALVER y GARCÍA. Ingienería de software. S.l.: s.n., 2010.

17. CHANCHÍ GOLONDRINO, Gabriel Elías. Construcción y evaluación de servicios interactivos en entornos de TVDi. 31 diciembre 2015. S.l.: s.n.

18. MONIRUZZAMAN y HOSSAIN. Ingienería de software. S.l.: s.n., [sin fecha].

19. NAVARRO y FERNÁNDEZ. 2013. S.l.: s.n.

20. García. Ingienería de software. S.l.: s.n., 2015.

21. PEÑALVER. Ingienería de software. S.l.: s.n., 2010.

22. SÁNCHEZ. Metodología de desarrollo para la Actividad productiva de la UCI. 2015. S.l.: s.n.

23. G. BOOCH. El lenguaje unificado de modelado . Addison-Wesley. S.l.: s.n., 1999.

24. GARZÓN. 2010. S.l.: s.n.

25. PAVÓN MESTRAS, Juan. Aplicaciones Web/Sistemas Web. S.l.: s.n.

26. R. NAVARRO. REST vs Web Services [en línea]. julio 2006. S.l.: s.n. Disponible desde: http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf.

27. GARLAN. Stfware Architecture. S.l.: s.n., 1993.

28. PRESMMAN. Ingienería de software un efoque práctico. S.l.: s.n., 2010.

29. SOMMERVILLE. Ingieneria de sotfware. S.l.: s.n., 2011.

30. GAMMA ET AL. Ingeniería Aplicada a Cooperación al Desarrollo. 1994. S.l.: s.n.

31. LARMAN. Agile and iterative development. 2004. S.l.: s.n.

32. ABRAHAM, Silberschatz. Fundamentos de base de datos. S.l.: s.n.

33. MICROSOFT. Microsoft-Digital-Image-Starter-Edition. 2006. S.l.: s.n.

34. COTT M. DAVIS. La Marca: máximo valor de su empresa. 2002. S.l.: s.n.

Anexos:

Anexo A:

Tabla 7. HU4: Listar documento

Historia de usuarios	
Numero: HU4	Usuario: Todos
Nombre de Historia: Listar documento.	
Prioridad en el negocio: Alta	Riesgo en desarrollo: medio
Puntos estimados:1.5	Iteración asignada
Programador Responsable: Yesenia de la Caridad Valdés Torres	
Descripción: Cunado el usuario se autentica se le muestra una lista de los libros que se encuentran en la plataforma Sunset.	
Observación: Si la plataforma se encuentra vacía, se muestra un mensaje (No hay libro que mostrar).	

Tabla 8 HU5 Descargar documento

Historia de usuarios	
Numero: HU5	Usuario: Todos
Nombre de Historia: Descargar documento.	
Prioridad en el negocio: Alta	Riesgo en desarrollo: medio
Puntos estimados:1.5	Iteración asignada
Programador Responsable: Yesenia de la Caridad Valdés Torres	
Descripción: El usuario luego de ver los detalles de un documento, tiene la opción de descargar el mismo, si el usuario elige esta opción se le muestra una barra indicando el proceso de la descarga.	
Observación: Si la plataforma se encuentra vacía, se muestra un mensaje (No hay libro que descargar).	