Universidad de las Ciencias Informáticas

Facultad 1

Título: Sistema de gestión de archivos para la Organización Nacional de Bufetes Colectivos

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Autor: Emilio Antonio Castro Quiñones

Tutores: MSc. Leiny Pons Flores

MSc. Aneyty Martín García

Ciudad de La Habana, 2021

FRASE

“Ser bueno es fácil, lo difícil es ser justo”

Victor Hugo

AGRADECIMIENTOS

le agradezco a mis padres a mi novia, a mis tutores y a mis amigos

DEDICATORIA

DELARACIÓN DE AUTORÍA

Declaro por este medio que yo **Emilio Antonio Castro Quiñones**, con carné de identidad **96092208268** soy el autor principal del trabajo titulado “Sistema de gestión de archivos para la Organización Nacional de Bufetes Colectivos” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los _____ días del mes de _____ de _____

_____
Emilio Antonio Castro
Quiñones
Autor

_____
Msc. Leiny Pons Flores
Tutor

_____
MSc. Aneyty Martín García
Tutor

RESUMEN

Los avances de las Tecnologías de la Información y la Comunicación, han alcanzado todos los sectores de la vida y cada vez son más importante en el sector empresarial concretamente con la gestión de documental. En la organización nacional de bufetes de abogados a pesar de existir un sistema de gestión documental, este presenta varias dificultades de estabilidad y rapidez en cuanto a funcionalidad. La presente investigación propone con el objetivo de mejorar los de gestión documental de los abogados afiliados a la organización nacional de abogados, mediante las tecnologías web, desarrollar un sistema para gestionar la información, la propuesta está concebida como una opción para sustituir el sistema que lleva esta gestión. Para su desarrollo se concibe siguiendo la metodología AUP-UCI, *Visual Paradigm* para UML 8.0 como herramienta de modelado, como lenguaje de programación *Javascript* con *Node.js*, *Visual Studio Code* como IDE de desarrollo. Se espera obtener como resultado un sistema con una arquitectura ágil diseñada sobre la base de los patrones GRASP y GOF. Además, se le aplicaran pruebas de tipo caja negra y caja blanca, más las pruebas de seguridad y rendimiento.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Las Tecnologías de la Información y la Comunicación (TIC) o en ingles ICT (*Information and Communications Technology*), hace referencia a las teorías, las herramientas y técnicas utilizadas para el procesamiento y difusión de la información. Hay otros modos de referirse al concepto y que son igualmente aceptados como NTICS, que agrega la sigla N haciendo alusión a nuevas o TI que simplemente es tecnología de la información (Hernández, 2017). Actualmente las NTICs se encuentran cada vez más presentes en nuestra vida diaria y en las diferentes formas de trabajo ya sea en empresas, instituciones o hasta en los más pequeños sectores de la sociedad haciendo que crezca la cantidad de información y con esto la manera más óptima de gestionarla.

La información como concepto ha sido debate desde la edad media donde se empezó a conceptualizar y ha ido evolucionando las formas de almacenarla y procesarla. En la actualidad las tecnologías nos han brindado diferentes formas de hacerlo de manera eficiente y visto la creciente necesidad de almacenar grandes cantidades de información se han creado herramientas para solucionar estas problemáticas.

Para manejar grandes cantidades de información por diferentes usuarios es muy común usar las *cloud* dada la facilidad que muestran a la hora de implementarlas. El concepto de *cloud* según (Antonopoulos y Gillam, 2017) se define como un modelo de prestación de servicios y acceso en el que se proporcionan recursos virtualizados y escalables dinámicamente como un servicio a través de internet. Hay diferentes formatos para los cuales se puede guardar la información en estas *cloud,* entre los más utilizados se encuentran audios, videos, archivos de texto, etc.

Estos archivos en las empresas tienen un alto valor como información, en la actualidad su mayoría se encuentra de manera digital con algunas de las extensiones antes mencionadas. En los últimos años se evidencia que son muchos los archivos que nacen ya en formato electrónico y que conviven con los archivos en formato físico. Su gestión es primordial en cualquier organización para no sumirse en un caos generado por una falta de control y normalización. Aún, en los tiempos que corren, hay organizaciones que todavía no tienen un sistema de gestión documental y aunque seguiremos unos años con una gestión mixta de la documentación, papel y electrónico, es necesario tener una buena gestión en papel para que se herede este modelo en la gestión electrónica (Comunidad Baratz, 2015).

En Cuba, el marco regulatorio para la Gestión Documental marca las pautas en el tratamiento de los documentos de archivo en las diversas instituciones. Dentro de este marco destaca la Ley 265 del 2009 (Consejo de Estado, 2009). Esta establece las normas para la actividad archivística en el territorio

nacional como parte de la administración pública, la gestión de sus documentos y la administración de sus archivos. Con las disposiciones, esta ley facilita los procesos de Control Interno y Gestión de Riesgos. De igual forma, su implementación contribuye a la posterior Gobernanza de la Información como vía o recurso que demanda la iniciativa de gobierno electrónico en Cuba. El Ministerio de Ciencia, Tecnología y Medio Ambiente (CITMA) en conjunto con la Contraloría General de la República son las entidades revisoras y contraloras del estado de la gestión documental en las instituciones cubanas por la relación establecida entre gestión documental, Control Interno y Gestión de Riesgos (Jiménez y Pérez, 2020).

La Organización Nacional de Bufetes Colectivos (ONBC), es una de las instituciones que forma parte del sector jurídico del país y que debido a la naturaleza de los trámites legales produce mucha información documental que es gestionada por normas y procedimientos. Su objetivo fundamental es prestar servicios con calidad a personas naturales y jurídicas, nacionales y extranjeras, contando con profesionales calificados en todas las ramas del Derecho, poseedores de una elevada cultura intelectual y ética profesional, y eficacia en su gestión, reconocida por los clientes y la sociedad.

La Organización Nacional de Bufetes Colectivos agrupa más de dos mil abogados miembros, los cuales sustentan el servicio que se brinda al pueblo. Como parte de sus funciones de trabajo les corresponde mantener actualizada la documentación que se genera en cada proceso legal que desarrollan. Debido a los avances de las tecnologías de la información y las comunicaciones muchos de los documentos que utilizan en sus procesos se encuentran en formato digital. También, utilizando la vía digital, se procede a la comunicación con el resto de las instituciones del sector jurídico.

Este avance tecnológico ha producido el surgimiento de la necesidad de gestionar toda su información digital para organizar cada documento por procesos y materias en carpetas digitales. En la actualidad, los documentos se almacenan en las computadoras de cada bufete y estas no pertenecen a un usuario individual, provocando en ocasiones la pérdida sin recuperación de la información.

Se utilizan también para almacenar la documentación memorias USB y discos duros externos, que al ser dispositivos pequeños se encuentran sujetos a la posibilidad de pérdidas, roturas y otras cuestiones que impiden su utilización de forma ininterrumpida. En materia legal, la pérdida de información es un grave problema que afecta los momentos procesales imprescindibles provocando el vencimiento de términos legales, la pérdida de litigios y las quejas de la población sobre el servicio. Después de analizar el resultado obtenido a partir de las encuestas realizadas entre algunos bufetes capitalinos se llegó a la conclusión de que el 90% de los trabajadores de las instituciones usaban dispositivos portátiles para

almacenar contenido de trabajo de ese 90% el 70% en algún momento han presentado roturas, perdida u otras razones que han acabado en la perdida de la información y de ese 70% el 40% han visto afectado su desempeño laboral y según los resultados de la última pregunta de la encuesta se puede deducir datos más reales donde por estos inconvenientes se dedujo que las afectaciones en días variaban desde una semana hasta meses en dependencia de la información que se perdía dado que al perder estos datos se tenía que volver a desarrollar todo el proceso para recopilar todas estas informaciones y en algunos casos retrasaban muchos procesos burocráticos que se encontraban en marcha.

La Dirección de Informática de esta institución ha realizado un estudio de sistemas que puedan utilizarse para gestionar estos archivos, concluyendo que demandan la utilización de grandes volúmenes de memoria RAM en los servidores, así como de CPU para su funcionamiento. Estos recursos contratados por la compañía ETECSA son caros para mantener en un período de tiempo indefinido. Mensualmente el servicio tiene un valor que supera los 111 mil pesos, afectando también la economía de la Organización.

Teniendo en cuenta lo antes planteado se identifica el siguiente **problema de la investigación**: ¿Cómo contribuir a la gestión de la documentación digital utilizada por los abogados en la Organización Nacional de Bufetes Colectivos?

En correspondencia con el problema señalado se define como **objeto de estudio**: los procesos de gestión de la documentación digital, enmarcándose en el **campo de acción**: proceso de gestión de la documentación digital en la Organización Nacional de Bufetes Colectivos.

Se define como **objetivo general** de la investigación: Desarrollar un sistema basado en tecnologías web para contribuir a la gestión de la documentación digital utilizada por los abogados en la Organización Nacional de Bufetes Colectivos.

Para darle cumplimiento al objetivo general, se definen los siguientes **objetivos específicos:**

1. Caracterizar los fundamentos teóricos relacionados con las herramientas y sistemas de gestión documental.

2. Definir las tecnologías, las herramientas y la metodología para la implementación del sistema web que permita contribuir a la gestión de la documentación digital utilizada por los abogados en la Organización Nacional de Bufetes Colectivos.

3. Implementar el sistema web que permita contribuir a la gestión de la documentación digital utilizada por los abogados en la Organización Nacional de Bufetes Colectivos.

4. Validar mediante las pruebas de rendimiento caja negra y caja blanca, así como la técnica de IADOV el correcto funcionamiento del sistema.

Se define como **hipótesis**: Con el desarrollo de un sistema basado en tecnologías web que permita la carga, descarga y divulgación de archivos, se contribuirá a la gestión de la documentación digital utilizada por los abogados en la Organización Nacional de Bufetes Colectivos.

Métodos Científicos:

Teóricos:

- ✓ **Analítico-Sintético:** permitió realizar el estudio teórico de la investigación facilitando el análisis de documentos y la extracción de los elementos más importantes relacionados con el proceso de desarrollo de sistemas web para contribuir a la gestión documental en el ámbito empresarial.

- ✓ **Hipotético-Deductivo:** Este método permitió obtener la hipótesis planteada y a partir de ella derivar conclusiones en el transcurso de la investigación.

- ✓ **Análisis-Documental:** Mediante este método se hizo un estudio de documentación referente al uso de las TIC en la gestión documental.

Empíricos:

- ✓ **Entrevista:** Se empleó en encuentros con el cliente para conocer la necesidad del desarrollo de la propuesta de solución, definir sus funcionalidades e identificar a la vez particularidades y las restricciones que se imponen.

- ✓ **Observación:** Posibilitó la obtención de conocimiento acerca del funcionamiento de los sistemas web de gestión documental existentes en la actualidad.

CAPÍTULO 1: FUNDAMENTOS DE LA INVESTIGACIÓN, METODOLOGÍA Y TECNOLOGÍAS EMPLEADAS

En el presente capítulo se definen conceptos fundamentales para el entendimiento de la investigación, así como también se analizarán distintos sistemas homólogos para establecer una comparación y se identificará la metodología y las herramientas a utilizar para la solución del problema.

1.1 Marco conceptual

En las instituciones se abarca mucho el manejo de la documentación, la forma de manejar la información con sus diferentes etapas y como una institución es cualquier tipo de organización humana**,** que implica relaciones estables y estructuradas entre las personas, que se mantienen en el tiempo, con el fin de cumplir una serie de objetivos explícitos o implícitos (María Estela Raffino, 2020) y un documento según (Porto y Merino, 2009) es una carta, diploma o escrito que ilustra acerca de un hecho, situación o circunstancia. También se trata del escrito que presenta datos susceptibles de ser utilizados para comprobar algo, se puede diferir teniendo claro estos dos conceptos que documentación institucional no es más que la documentación ligada a una institución. Para hacer un buen uso de la documentación se hace uso de la gestión documental que según (Alonso, 2018) es el conjunto de normas, técnicas y prácticas usadas para administrar el flujo de documentos de todo tipo en una organización, permitir la recuperación de información desde ellos, determinar el tiempo que los documentos deben guardarse, eliminar los que ya no sirven y asegurar la conservación indefinida de los documentos más valiosos, aplicando principios de racionalización y economía.

Dejando plasmadas las ideas anteriores pasaremos a hablar de la información como proceso de gestión que según (Martínez González, Alfaro Rivera y Ramírez Montoya 2012) se aborda a la luz de los conceptos y modelos de la administración del conocimiento, que se define como "la coordinación deliberada y sistemática de la gente, de la tecnología, los procesos y estructura de una organización con el objetivo de agregar valor por medio de la replicación y la innovación". Si bien esta disciplina surge y se aplica en empresas y organizaciones no necesariamente dedicadas a la educación, al considerarla, nos brinda un referente para identificar y analizar los procesos de formación de investigadores, en especial en la forma en que gestionan la información y transforman el conocimiento de explícito a tácito y luego a explícito de nuevo desde el nivel de conocimiento en el que se encuentran y cómo buscan conseguir mayores niveles, con base en qué modelos se gestiona y construye su propio aprendizaje. En particular, modelos como el de socialización–externalización–combinación–internalización (SECI), que

representan una analogía del proceso de investigación: partir del conocimiento que se tiene o se intuye a nivel individual para articularlo, sistematizarlo y compartirlo a fin de que sea aplicado o asimilado, y quede documentado en una tesis que representa el producto final, tangible, de ese proceso de administración del conocimiento. Este proceso es muy complejo y se puede desglosar en tareas haciendo más fácil su entendimiento y ejecución las cuales basados en la opinión de (Infotecarios, 2021) están descritas de las siguiente manera:

- Implantar sistemas para conservar, organizar y recuperar cualquier tipo de información interna, de carácter técnico, informes de inteligencia competitiva o cualquier otro tipo de información para lo cual utiliza el formato y los niveles adecuados de acceso según el usuario.
- Garantizar el acceso a la información externa bien en formato electrónico o no, incluyendo el acceso a la Web, o en cualquier otro soporte.
- Mantener un sistema de especialistas sobre información actualizada en cuanto a las limitaciones, legislación y condiciones del uso y explotación de la información por lo que se refiere a propiedad intelectual y legislación sobre la protección de datos.
- Desarrollar sistemas modernos y flexibles de diseminación selectiva de la información.
- Crear y mantener sistemas de comunicación para que la información fluya con rapidez y eficacia entre los miembros de la organización, por ejemplo, mediante la creación de una Intranet.

La información como proceso está caracterizada por tres estados información de reposo, información de tránsito o movimiento e información de uso que están descritos de la siguiente manera según (Sealpath 2020):

- Información en Reposo: Con este término nos referimos a la información que no está siendo accedida, usada, ni procesada y que se encuentra almacenada en un medio físico o lógico. Ejemplos pueden ser ficheros almacenados en servidores de ficheros, registros en bases de datos, documentos en unidades flash, discos duros etc.
- Información en Tránsito o Movimiento: Información que viaja a través de un email, web, aplicaciones de trabajo colaborativo tipo *Slack* o *Microsoft Teams*, mensajería instantánea, o cualquier tipo de canal privado o público de comunicación. Es información que se encuentra viajando de un punto a otro.
- Información en Uso: Hablamos de información en uso cuando esta es accedida por una o varias aplicaciones para su tratamiento. Normalmente detrás de la aplicación está un usuario que quiere acceder a los datos para visualizarlos, modificarlos, etc.

1.2 Estudio de sistemas homólogos

En la actualidad existen muchos sistemas que permiten el servicio de alojamiento de archivos y la gestión de la información; proporcionándoles a los usuarios crear, eliminar, modificar, leer y escribir archivos, además de organizarlos de acuerdo a su propio entendimiento. A continuación, se explicarán algunos de estos sistemas que nos proporcionan el almacenamiento de archivos.

- ✓ ***NextCloud***

Nextcloud es una serie de programas cliente-servidor con el objetivo de crear servicio de alojamiento de archivos. Su funcionalidad es similar al software *Dropbox*, aunque *Nextcloud* es de tipo código abierto, permitiendo a quien lo desee instalarlo en un servidor privado. Su arquitectura abierta permite añadir funcionalidad al servidor en forma de aplicaciones. *Nextcloud* es una bifurcación de *ownCloud* que también es un software de servicio de alojamiento en la nube (Derrik Diener 2017), (*Nextcloud* GmbH 2017).

- ✓ ***Seafile***

Seafile es un software de alojamiento de archivos de código abierto, y multiplataforma, por ser compatible con ***GNU/Linux***, *Microsoft Windows* y *Mac* OSX. Los archivos se almacenan en un servidor central y se pueden sincronizar con computadoras, u otros dispositivos móviles, a través de aplicaciones de terceros. También se puede acceder a los archivos, directamente, en el servidor de Seafile a través de una interfaz web (david ochobits 2019).

- ✓ ***HiDrive***

Strato HiDrive es un disco duro *on line* pensado para uso tanto personal como profesional en el que el usuario almacena toda clase de contenidos como documentos de trabajo, PDFs, de forma centralizada y sólo se requiere de un dispositivo con conexión a fotografías o música, accediendo y compartiendo todo este material en cualquier instante. En este sentido, los archivos se almacenan internet para su consulta (Francisco Javier Palazón 2017).

- ✓ ***MEGA***

MEGA es un servicio de almacenamiento en la nube que se utiliza para guardar, compartir o intercambiar archivos de forma libre. Se puede acceder a ellos desde *Windows*, *Mac* OS X, *Linux* y desde cualquier dispositivo móvil con internet. *MEGA* es la versión mejorada de *MegaUpload*, servicio de alojamiento de archivos creado por el informático Kim Dotcom en el año 2005. El principal avance de *MEGA* consiste

en que, al subir el archivo o carpeta, la información queda totalmente encriptada mediante el sistema cifrado RSA 2048 bits, que sólo controla el usuario a su entera discreción (*MEGA* 2021).

- ✓ ***Dropbox***

Dropbox es un servicio de alojamiento de archivos multiplataforma en la nube, operado por la compañía estadounidense *Dropbox* (*CrunchBase* 2008). El servicio permite a los usuarios almacenar y sincronizar archivos en línea y entre ordenadores y compartir archivos y carpetas con otros usuarios y con tabletas y móviles (Elibeth Eduardo García 2016).

- ✓ **GDA Xabal excriba 3.1**

GDA Xabal excriba es un software para la gestión documental, diseñado para tramitar los documentos administrativos que se generan o reciben dentro de las organizaciones a partir de sus funciones, por lo tanto, involucra todas las áreas de una organización, permitiéndoles gestionar de forma correcta la documentación como prueba, testimonio y evidencia de las actividades organizacionales.

La solución del software agiliza el trámite de los documentos, permitiendo controlar el estado de los mismos, evitando la pérdida de información y tributando como herramienta de apoyo para la toma de decisiones. GDA Xabal excriba, permite trabajar en un entorno de colaboración entre los usuarios de la organización, compartiendo sus documentos con otros usuarios y estableciendo niveles de acceso y permiso. Además, optimiza la organización de los documentos, posibilita la búsqueda de información y gestiona los documentos durante su ciclo de vida. Las interfaces de usuario de esta aplicación están orientadas a personas que no tienen que poseer un alto grado de conocimiento de informática para trabajar con el sistema.

Entre las funcionalidades que brinda el software para la gestión de los documentos se encuentran las siguientes: Automatización de los flujos documentales (aprobación o rechazo, revisión). Gestión de documentos (crear, actualizar el contenido del documento, cortar, copiar, eliminar, crear accesos directos, mostrar notas adjuntas al documento). Gestión de carpetas (crear, eliminar, cortar, pegar, crear accesos directos). Control de versiones (crear una copia de trabajo al documento, subir los cambios realizados de un documento, cancelar los cambios realizados de un documento, describir los nuevos cambios realizados). Control de acceso y permisos (Compartir documentos y carpetas, asignar permisos, eliminar permisos, editar permisos). Notificaciones (Notificar a usuarios y grupos de usuarios cuando se comparte un documento, cuando forma parte del flujo documental, tanto en la revisión como la aprobación). Búsquedas (Realizar búsquedas a texto completo, o sea, buscar por palabras que estén dentro del contenido, por metadatos recogidos en los documentos y carpetas). Seguimiento de la

auditoría sobre los documentos electrónicos (posibilitando conocer que personas dentro de la institución han accedido al documento, que cambios han realizado, en qué fecha se modificó) (Alexander Nápoles Terry 2016).

1.2.1 Resultados del estudio de los sistemas homólogos

Después del estudio realizado a los diferentes sistemas homólogos presenciados en el capítulo anterior se confeccionó una tabla (ver tabla 1) para llevar a cabo una comparación donde se tienen en cuenta diferentes parámetros como:

- Idiomas soportados por el sistema (ISS)
- Tamaño máximo de almacenamiento (TMA)
- Posibilidad de alojamiento en un servidor propio (PASP)
- Sistemas operativos que soportan (SOS)
- Estándar de seguridad (PDS)

Tabla 1: *Sistemas de servicios de alojamiento de archivos. Fuente de elaboración propia.*

Parámetros	***NextCloud***	***Seafile***	***HiDrive***	***Mega***	***Dropbox***
ISS	Multilenguaje	Multilenguaje	Español, inglés, alemán y portugués	Multilenguaje	Multilenguaje
TMA	Programable	Programable	5gb(gratis) Hasta 2tb (*premium*)	50 GB (gratis) hasta 8 TB (*premium*)	2 GB (gratis) 2TB (pro)
PDS	Configurable	Configurable	ISO 27001	Carece	ISO 27001
PASP	Si	Si	No	No	No
SOS	*Windows, OS X, Linux, FreeBSD, Android, iOS*	*Linux, OS X, Windows, Android, iOS.*	*IOS, Android, Windows Phone, windows y OS X*	*Windows, OS X, Linux, Android, iPhone, BlackBerry*	*Windows*, OS X, Linux, *Android*, *iPhone*, *BlackBerry*

Si tenemos en consideración la información brindada por la tabla (ver tabla 1) queda evidenciado que *DropBox, Mega* y *Hidrive* no son soluciones factibles dado que no tiene la posibilidad de alojar en un servidor propio, por lo que no dan la posibilidad de hacer uso de ellos en los servidores nacionales. *NextCloud* contiene muchas carencias en el procesamiento de los datos y no se recomienda su uso para una gran cantidad de usuarios simultáneamente, dando como consecuencia que presente problemas de lentitud, estos aspectos fueron evidenciados en la práctica porque es el sistema que se utiliza actualmente. Seafile aun cuando es de código abierto y deja alojar en un servidor propio, la versión completa con todas las funcionalidades es privada. Con lo antes planteado queda evidenciado que no hubo sistemas capaces de dar solución al problema de la investigación y lo cual hizo necesario desarrollar el sistema web propuesto.

Apoyándonos en el estudio realizado anteriormente y tomando algunas de las ventajas de los sistemas mencionados se concluye que la propuesta de solución debe tener una alta seguridad que para lograr se va a ser uso como guía el estándar de seguridad ISO 27001, debe soportar varias conexiones simultaneas sin presentar problemas de lentitud y una buena gestión de archivos. Debe ser un sistema con una capacidad configurable en dependencia de las necesidades de los usuarios que hagan uso de este. Teniendo en cuenta el poco conocimiento de los usuarios con las tecnologías y el proceso de capacitación que se le dio a los usuarios para poder manejar el sistema se recomienda que tenga una interfaz visual y funcional lo más parecida al sistema actualmente en uso.

1.3 Selección de metodología, herramientas, lenguaje y tecnología

Para el desarrollo de la aplicación se estudiarán metodologías, lenguajes de programación y herramientas específicas. Para luego escoger las más eficientes y óptimas, en aras de implementar un producto con calidad.

Framework

- ✓ **Angular**

AngularJS es un *framework* para desarrollo web construido por Google e inicialmente liberado en 2010. Angular (a secas) también es un *framework* para desarrollo web, y también construido por Google (y de hecho por varios de las mismas personas que hicieron AngularJS). Lo que hoy se conoce como “Angular” fue inicialmente llamado “Angular 2” ya que era visto como la siguiente versión de Angular 1.x (lo que hoy es llamado AngularJS). Sin embargo, dado que no son retro compatibles, y que Angular 2 tiene un

alcance más amplio, se decidió mantener la rama 1.x de Angular bajo el nombre AngularJS mientras que la rama nueva se quedó simplemente como Angular (Jorge Cano 2020).

- ✓ **Symfony**

Symfony es un entorno de trabajo estandarizado (*framework* PHP) que se utiliza para el desarrollo de aplicaciones web y es de los más utilizados en el entorno de desarrolladores de apps. En otras palabras, es una herramienta para desarrolladores para crear aplicaciones en PHP. El *framework* es escalable y usa modelo vista controlador, contiene una gran cantidad de plantillas para desarrollar webs y es muy utilizado en la creación de APIs (Quality Devs 2019).

- ✓ **Laravel**

Laravel es un *framework* web con una sintaxis elegante y expresiva. Un marco web proporciona una estructura y un punto de partida para crear su aplicación, lo que le permite concentrarse en crear algo sorprendente mientras nos preocupamos por los detalles. Laravel se esfuerza por proporcionar una experiencia de desarrollador increíble al tiempo que proporciona funciones poderosas como la inyección de dependencias exhaustiva, una capa de abstracción de base de datos expresiva, colas y trabajos programados, pruebas de integración y unidad, y más (Laravel 2021).

- ✓ **CakePHP**

CakePHP es un entorno de desarrollo basado en la arquitectura MVC (Modelo Vista Controlador) que permite la creación de páginas web de forma sencilla a través de una interfaz amigable. Está basado en el *framework* Ruby *on Rails*. La herramienta nos provee de una serie de herramientas para trabajar cómodamente con bases de datos relacionales para almacenar y mostrar la información de nuestro proyecto web, siendo extremadamente fácil crear un blog o una tienda virtual en pocos pasos gracias a sus módulos de ayuda. CakePHP integra CRUD, acrónimo de las cuatro funciones básicas de acceso a una base de datos (Crear, Obtener, Actualizar y Borrar). Es compatible con las versiones de PHP 4 y 5. Soporta *scaffolding*, permite el uso de funciones AJAX y la utilización de plantillas. La herramienta es gratuita y de código abierto, y es capaz de funcionar sobre la mayoría de servidores y hospedajes web comerciales. Su principal baza es que es un sistema muy escalable donde podemos reutilizar código de un proyecto a otro. Además, existe una gran comunidad de usuarios, tutoriales y ejemplos por la red (Álvaro Toledo 2018).

1.3.1 Lenguajes de programación

Un lenguaje de programación es una interfaz de usuario con la cual los seres humanos pueden elaborar procesos que serán ejecutados por una máquina de cómputo. Los lenguajes de programación tienen reglas y parámetros establecidos para poder realizar diferentes acciones, existen diferentes lenguajes de programación y dependen del tipo de programación (Alvaro Humberto Cisneros 2017).

A continuación, se explicará brevemente algunos de los lenguajes de programación existentes para desarrollar aplicaciones web.

- ✓ **Python**

Python es un lenguaje de programación interpretado, multiparadigma y multiplataforma usado, principalmente, en *Big Data*, AI (Inteligencia Artificial), *Data Science, frameworks* de pruebas y desarrollo web. Esto lo convierte en un lenguaje de propósito general de gran nivel debido a su extensa biblioteca, cuya colección ofrece una amplia gama de instalaciones (Fernando Machuca 2021).

- ✓ **PHP**

PHP (*Hypertext Preprocessor*) es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo Web de contenido dinámico. Es un lenguaje adecuado para el desarrollo de aplicaciones Web de manera dinámica. Es un lenguaje incrustado en HTML lo que significa que se combinan código PHP y HTML en un mismo archivo en un determinado servidor. Una de las principales razones de la popularidad de PHP como lenguaje de creación de scripts para Web es su amplio soporte a diferentes bases de datos, facilitando que los desarrolladores creen sitios sustentados en bases de datos, y que se hagan nuevos prototipos de aplicaciones web de manera rápida y eficiente sin demasiada complejidad (Milton Rafael Valarezo Pardo y otros 2018).

- ✓ **JavaScript**

JavaScript (JS) es un lenguaje de programación ligero, interpretado, o compilado justo-a-tiempo (*just-in-time*) con funciones de primera clase. Si bien es más conocido como un lenguaje de *scripting* (secuencias de comandos) para páginas web, y es usado en muchos entornos fuera del navegador, tal como Node.js, Apache CouchDB y Adobe Acrobat. JavaScript es un lenguaje de programación basada en prototipos, multiparadigma, de un solo hilo, dinámico, con soporte para programación orientada a objetos, imperativa y declarativa (por ejemplo, programación funcional) (MDN *contributors* 2021).

- ✓ **TypeScrypt**

TypeScript es una extensión de JavaScript destinada a facilitar el desarrollo de aplicaciones JavaScript a gran escala. Si bien cada programa JavaScript es un programa TypeScript, TypeScript ofrece un sistema de módulos, clases, interfaces y un rico sistema de tipos graduales. La intención es que TypeScript proporcione una transición fluida para los programadores de JavaScript; los modismos de programación JavaScript bien establecidos son compatibles sin ninguna reescritura o anotaciones importantes. Una consecuencia interesante es que el sistema de tipos TypeScript no es estáticamente sólido por diseño. El objetivo de este artículo es capturar la esencia de TypeScript dando una definición precisa de este sistema de tipos en un conjunto básico de construcciones del lenguaje. Nuestra principal contribución, más allá de las ventajas familiares de una formalización matemática sólida, es una refactorización en un fragmento interno seguro y una capa adicional de reglas inseguras (Kristin Fjola Tomasdottir y otros 2018).

1.3.2 Tecnologías y herramientas

A continuación, se presentan varias herramientas y tecnologías utilizadas para el desarrollo de sistemas web, de las cuales se seleccionarán las más adecuadas, de acuerdo a las características del proyecto y en aras de implementar un producto con calidad.

- ✓ **Visual Paradigm 8.0**

Se realizó un estudio de las herramientas CASE más utilizadas en el ciclo de vida de desarrollo del software, donde se destacan: Oracle Designer, Visual Paradigm y Power Designer. La herramienta escogida para el modelado de la aplicación es Visual Paradigm. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos.

Visual Paradigm es una herramienta CASE profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue (Visual Paradigm 2021).

Esta herramienta permite aumentar la calidad del software, a través de la mejora en el desarrollo y mantenimiento del mismo, de igual forma potencia la reutilización del software y estandarización de la documentación, además del uso de las distintas metodologías propias de la Ingeniería del Software (PRESSMAN, Roger S, 2010).

- ✓ **Lucidchart**

Lucidchart es una herramienta de diagramación basada en la web, que permite a los usuarios colaborar y trabajar juntos en tiempo real, creando diagramas de flujo, organigramas, diagramas de procesos de redes TI, esquemas de sitios web, diseños UML (Lenguaje Unificado de Modelado), mapas mentales, prototipos de software y muchos otros tipos de diagramas. La plataforma basada en la nube y la interfaz intuitiva facilitan la creación de diagramas, sin importar el dispositivo, el navegador o el sistema operativo. Lucidchart mejora la forma en que los equipos trabajan juntos con co-autoría en tiempo real, chat en el editor, comentarios específicos de forma y cursores colaborativos, ya sea que estén en el mismo país como Colombia o en diversos lugares del mundo.

Esta aplicación de diagramación en línea facilita la creación y el intercambio de diagramas profesionales entre todos los miembros del equipo de trabajo, además es muy fácil instalarlo en Colombia y en español. Desde el diseño de un sistema hasta la posibilidad de hacer lluvias de ideas de forma remota. Además, la gestión de proyectos cubre todas las necesidades de comunicación y colaboración de una empresa (SAS 2021).

- ✓ **Miro**

Miro es una plataforma que te permite crear un espacio de trabajo o pizarra colaborativa para dibujar esquemas gráficos y diagramas en tiempo real con otras personas o alumnos. Su principal ventaja es que ofrece en un mismo espacio virtual todas las funciones necesarias para trabajar en equipo, tablero digital, posibilidad de videoconferencia, elementos gráficos para dibujar y escribir en el tablero y un montón de plantillas para no empezar desde cero. Miro es una plataforma que favorece la creatividad y la innovación de las personas facilitando un espacio de trabajo libre, visual y colaborativo. Miro te ofrece múltiples plantillas, con diagramas, esquemas, mapas mentales y flujos de trabajo diseñados para el desarrollo de todo tipo de proyectos.

Con Miro puedes dar forma a las principales metodologías y técnicas de creatividad, innovación, desarrollo de proyectos, design thinking, etc de forma rápida y sencilla. Todo lo que pongas en la pizarra o espacio de trabajo será editable por todos los miembros del equipo y los cambios que cada persona realiza se ven en tiempo real en el mismo espacio de trabajo. Una de las ventajas de Miro es que te permite crear un entorno digital de trabajo para colaborar con tu equipo, tus colegas o tus compañeros en en tiempo real. Podrás ver en el mismo tablero dónde se encuentra cada miembro del equipo. Todos

los participantes podrán modificar el contenido en tiempo real. Miro tambien permite realizar videollamadas desde la misma plataforma y compartir pantalla con los miembros del equipo. Miro se integra perfectamente con las plataformas de trabajo o nubes colaborativas existentes como Dropbox, Google Suite, Slack, Microsoft Teams, Sketch, Asana, Notion, Trello, Excel, Evernote, etc (Allende 2021).

1.3.3 Metodología de desarrollo de software

Una metodología es el conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal (ASALE y RAE 2021). El uso de una metodología permite el dominio del proceso software. Se puede decir que una metodología es un enfoque, una manera de interpretar la realidad o la disciplina en cuestión, que en este caso particular correspondería a la Ingeniería del Software. A su vez, un método, es un procedimiento que se sigue en las ciencias para hallar la verdad y enseñarla. Es un conjunto de técnicas, herramientas y tareas que, de acuerdo a un enfoque metodológico, se aplican para la resolución de un problema. Ingeniería del Software Desde el punto de vista específico de la Ingeniería del Software, la metodología describe cómo se organiza un proyecto, el orden en el que la mayoría de las actividades tienen que realizarse y los enlaces entre ellas, indicando asimismo cómo tienen que realizarse algunas tareas proporcionando las herramientas concretas e intelectuales (Dr. Francisco José García Peñalvo 2018). Se puede concluir que las metodologías son un conjunto de pasos secuenciales que conllevan a la construcción de un software, las cuales se pueden adaptar dependiendo de las condiciones de trabajo, los requisitos del software y los diferentes aspectos que rodean al desarrollo del software, teniendo en cuenta esto se debe plasmar que las metodologías se dividen en dos métodos ágil y tradicional.

Por definición, las metodologías ágiles son aquellas que permiten adaptar la forma de trabajo a las condiciones del proyecto, consiguiendo flexibilidad e inmediatez en la respuesta para amoldar el proyecto y su desarrollo a las circunstancias específicas del entorno. En esencia, las empresas que apuestan por esta metodología consiguen gestionar sus proyectos de forma flexible, autónoma y eficaz reduciendo los costes e incrementando su productividad (Vanessa Rosselló Villán 2019).

Las metodologías tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo del software, para ello, se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto software. Se centran especialmente en el control del proceso, mediante una rigurosa definición de roles, actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada. Además, las metodologías tradicionales no

se adaptan adecuadamente a los cambios, por lo que no son métodos adecuados cuando se trabaja en un entorno, donde los requisitos no pueden predecirse o bien pueden variar (PMT 2019).

1.4 Selección del entorno de desarrollo

A través del estudio de las diferentes tecnologías presentadas anteriormente y otras que se mencionarán a continuación, pasaremos a escoger las más eficientes para el desarrollo del sistema. Del lado del servidor se ha decidido utilizar Node.js dada su facilidad para manejar mucha cantidad de usuarios de manera asincrónica. Cada conexión dispara la ejecución de un evento dentro del proceso del motor de Node.js. De este modo, Node.js permite que un solo servidor que lo ejecute pueda soportar decenas de miles de conexiones, en conclusión Node.js es perfecto para un sitio que maneje muchas conexiones y dada la cantidad de usuarios dentro de la Organización de Bufetes Colectivos Node.js cumple de manera satisfactoria con este parámetro. Para hacer uso de este entorno se va a ejecutar en el lenguaje JavaScript dado que es el lenguaje en el cual se ejecuta dicha tecnología. Del lado del cliente se toma la decisión de usar angular como *framework* dado que permite crear aplicaciones web robustas y la posibilidad de generar proyectos desde el CLI con Node.js. Para ambas tecnologías se usará Visual Studio Code como IDE de desarrollo, por las facilidades que le brinda al desarrollador y porque como ide contiene *plugins* y herramientas para albergar todos Node.js y Angular. Como metodología a utilizar se ha escogido AUP-UCI por sus características como metodología ágil. Como herramienta de modelado UML se escoge Visual Paradigm ya que a través del curso escolar del desarrollador fue la herramienta más usada y presenta una buena habilidad en dicha herramienta y teniendo en cuenta que las otras herramientas como Miro que es de pago, lo que sería una dificultad para utilizarla. Lucidchart que para su uso hay que estar conectado a internet de forma permanente lo cual sería otra dificultad para desarrollar dado las dificultades con la internet.

Conclusiones parciales

En el presente capítulo se ha realizado un estudio de algunos elementos teóricos que sustentan la investigación, luego del cual se obtienen las siguientes consideraciones finales:

- ✓ El estudio del proceso de la gestión documental, con el apoyo de las tecnologías web, permitió comprender los principales conceptos asociados al desarrollo de las mismas para el almacenamiento y gestión de información.
- ✓ Los sistemas existentes estudiados, para el servicio de alojamiento de archivos, no aportan una solución al problema, pero sirvieron de base para desarrollar una idea general de cómo se desea representar la solución final.

- ✓ El marco de trabajo para el desarrollo de la aplicación web para la gestión documental se definió mediante el uso de herramientas libres y bajo el desarrollo de la metodología AUP-UCI como guía del proceso de desarrollo.

CAPÍTULO 2: DESCRIPCIÓN DE LA APLICACIÓN WEB PARA GESTIONAR INFORMACIÓN UTILIZADA POR LOS ABOGADOS DE LA ONBC

En el presente capítulo se describen las características de la aplicación web a desarrollar, realizando un análisis de la propuesta de solución. Con la construcción del modelo de dominio se obtendrá un mayor acercamiento del problema a resolver y con la identificación de los requisitos funcionales y no funcionales serán mostradas las cualidades que deberá de cumplir la aplicación web para contribuir a la gestión de la documentación digital utilizada por los abogados en la Organización Nacional de Bufetes Colectivos. Además, se muestran un conjunto de diagramas desarrollados durante el análisis y diseño de la aplicación según las características propuestas por la metodología seleccionada AUP-UCI.

2.1 Descripción de la propuesta de solución

El Sistema de Gestión Documental Jurídico va a permitir que los usuarios que estén afiliados a la abogacía y a la ONBC puedan guardar su información de manera eficiente, subir archivos hacia su almacén de datos en la red, descargarlos para su uso posterior o crearlos. Además, podrán acceder a través de un portal web desde sus aparatos electrónicos que tengan la capacidad para ejecutar un navegador. También tendrán la opción de compartir la información entre ellos y así de esta manera potenciar la comunicación y trabajo en equipo entre los usuarios, de visualizar las operaciones que realicen como compartir o los que han abierto recientemente y de esta manera tener una organización en las acciones que cometan en el sistema. En conclusión, el sistema proveerá una manera digital de tener almacenada toda la información pertinente por parte de los abogados generalmente afiliada a su labor.

A continuación, se presenta un prototipo de cómo se vería la interfaz de la aplicación una vez terminada con los diferentes menuces desplegados:

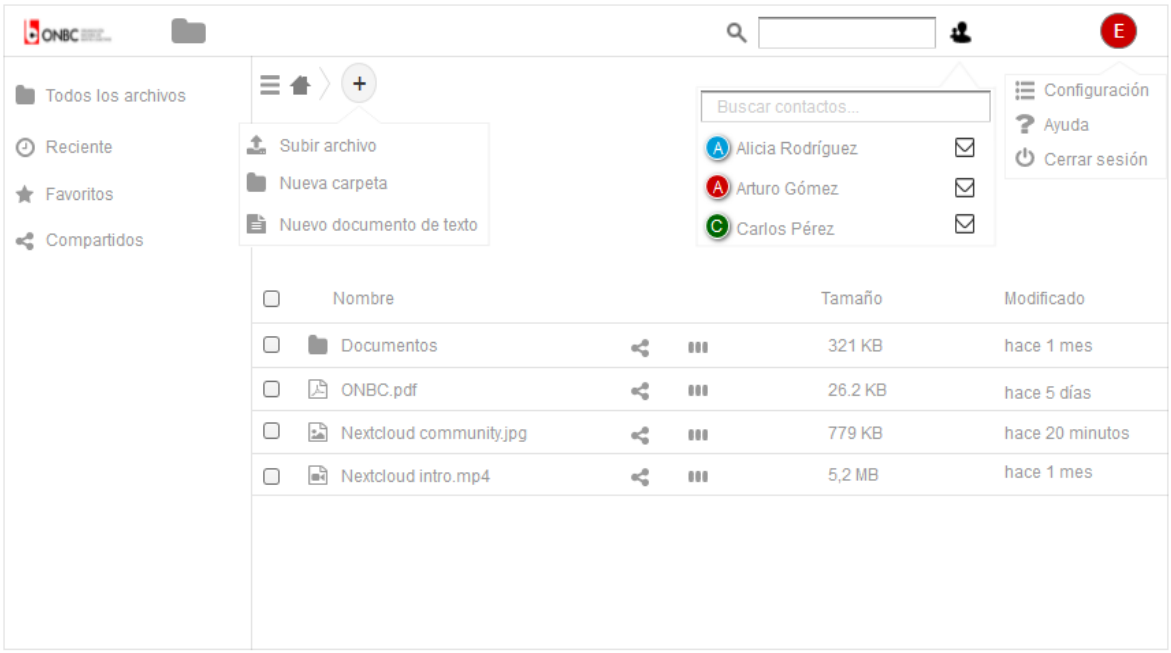


Figura 1: Prototipo de la propuesta de solución del sistema.

2.2 Modelo del dominio

El proceso de análisis o investigación orientada a objeto es un paso a realizar a la hora de separar el problema en conceptos o en objetos individuales. El modelo conceptual proporciona la ventaja de subrayar una concentración en los conceptos del dominio y no en las entidades del software (CRAIG, Larman 2003). Un modelo de dominio muestra las clases conceptuales significativas en un dominio del problema, su utilidad radica en ser una forma de “inspiración” para el diseño de los objetos software. Este se centra en las abstracciones relevantes, vocabulario del dominio e información del dominio, además es el artefacto clave del análisis orientado a objetos (García Peñalvo, Francisco José y Vázquez Ingelmo, Andrea, 2019).

Este modelo de dominio constituye el punto de partida para lograr realizar un diseño acertado del producto a desarrollar y fundamente los conceptos significativos para un mejor dominio del problema y las relaciones entre ellos.

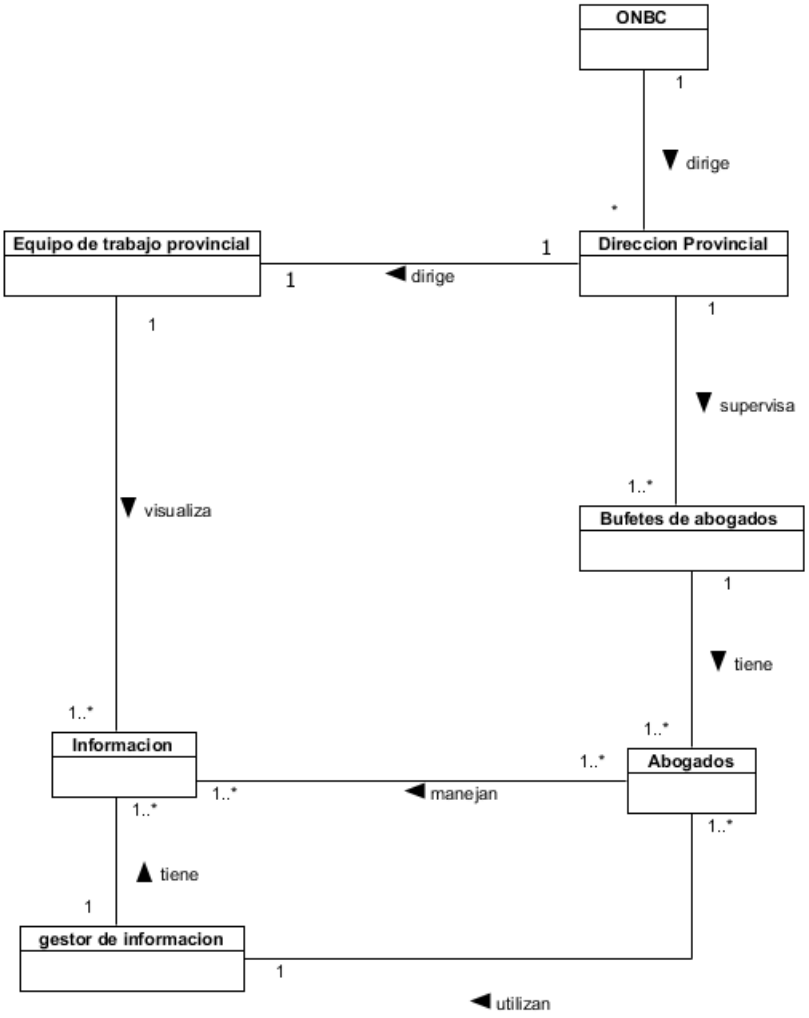


Figura 2: Modelo del dominio

En este modelo se evidencia, cómo La Organización Nacional de Bufetes Colectivos (ONBC) se encarga de dirigir la dirección de provincial la cual dirige el equipo de trabajo de la provincia y supervisa los bufetes de abogados los cuales contienen abogados que manejan información la cual debe ser visualizada por el equipo de trabajo de la provincia.

2.2.1 Descripción de conceptos

ONBC (Organización nacional de bufetes de colectivo): Es la organización responsable de dirigir y representar los bufetes a nivel nacional en Cuba.

Dirección provincial de bufetes: encargadas de dirigir los bufetes.

Bufetes de abogados: Empresas que atienden procesos judiciales y de leyes dando su servicio a la población u organización que así lo requieran.

Abogados: Profesionales en el ámbito de las leyes.

Información: Es un conjunto organizado de datos procesados que constituyen un mensaje que cambia el estado de conocimiento de la persona o sistema que recibe dicho mensaje.

Equipo de trabajo provincial: Un conjunto de personas encargado de supervisar la información que se almacena a la plataforma por parte de los abogados de los bufetes.

Gestor de información: Son las herramientas que permiten la gestión de la información.

2.3 Requisitos del sistema

Los requisitos del sistema establecen con detalle las funciones, servicios y restricciones operativas del sistema. El documento de requisitos del sistema (algunas veces denominado especificación funcional) debe ser preciso. Debe definir exactamente qué es lo que se va a implementar. Puede ser parte del contrato entre el comprador del sistema y los desarrolladores del software (Ian Somerville 2005).

Para lograr el desarrollo de un sistema correcto es necesario establecer de una forma clara los requisitos, quienes reflejan las necesidades de un cliente o de un usuario. Los requisitos pueden tener dos clasificaciones: requisitos funcionales y requisitos no funcionales.

2.3.1 Requisitos funcionales

Los requisitos funcionales de un sistema describen lo que el sistema debe hacer. Estos, dependen del tipo de software que se desarrolle, de sus posibles usuarios y del enfoque general tomado por la organización al redactar requisitos (Ian Somerville 2005).

Tabla 2: *Requisitos funcionales.*

No.	Nombre	Descripción	Prioridad	Complejidad
RF1	Autenticar usuario	Permite autenticar al usuario	Alta	Baja
RF2	Eliminar usuario	Permite eliminar un usuario	Baja	Media

RF3	Crear usuario	Permite crear un usuario	Alta	Baja
RF4	Compartir archivos	Permite compartir archivos	Baja	Media
RF5	Eliminar archivos	Permite eliminar archivos	Alta	Baja
RF6	Mover archivos	Permite mover archivos	Media	Baja
RF7	Visualizar archivos	Permite visualizar los archivos	Media	Baja
RF8	Buscar archivos	Permite buscar archivos	Baja	Baja
RF9	Descargar archivos	Permite descargar archivos	Alta	Baja
RF10	Crear archivos	Permite crear archivos	Baja	Baja
RF11	Modificar archivo	Permite modificar archivos	Media	Baja
RF12	Modificar nombre	Permite modificar nombre	Media	Baja
RF13	Modificar correo	Permite modificar correo	Media	Media
RF14	Visualizar usuario	Permite visualizar usuario	Media	Baja

RF15	Visualizar archivos recientes	Permite archivos recientes	Baja	Baja
RF16	Visualizar archivos compartidos	Permite visualizar los archivos compartidos	Baja	Baja
RF17	Visualizar archivos favoritos	Permite visualizar archivos favoritos	Baja	Baja
RF18	Visualizar actividades	Permite visualizar las actividades	Baja	Baja
RF19	Visualizar notificaciones	Permite las notificaciones	Baja	Baja
RF20	Buscar contactos	Permite buscar contactos	Media	Media
RF21	Imprimir documento	Permite imprimir los documentos	Media	Media
RF22	Insertar tarea	Permite insertar tareas	Media	Baja
RF23	Subir archivo	Permite subir archivos	Alta	Media

2.3.2 Requisitos no funcionales

Los requisitos no funcionales del software son difíciles de verificar y validar. Por su naturaleza, existen limitaciones técnicas que reducen alcance de la cobertura de pruebas a ser utiliza- da para su verificación y validación (numerosos escenarios diferentes a ser probados, multitud de combinaciones posibles y diferenciadas tanto de las entradas a las funciones como de las características que deben

verificarse, simulaciones muy costosas de entornos anómalos, etc.) Todo ello obliga al desarrollador a priorizar el conjunto limitado de pruebas a realizar considerando las restricciones en tiempo y presupuesto para cada proyecto (José Carlos Sánchez Domínguez y Patricia Rodríguez Dapena 2003).

Los requerimientos de calidad, que representan restricciones o las cualidades que el sistema debe tener tales como: precisión, usabilidad, seguridad, rendimiento, confiabilidad, performance entre otras. Estos requisitos poseen una naturaleza abstracta e intangible en comparación con los RF y esto hace que sean más difíciles de especificar o documentar formalmente (Molina Hernández, Granda Dihigo y Velázquez Cintra 2019).

Confiabilidad

RNF1- La aplicación debe ser capaz de manejar los errores y recuperarse.

Usabilidad

RNF2- Las estaciones de trabajo de los usuarios deben contar con un navegador web que soporte HTML5, CSS3 y JavaScript.

RNF3- Memoria RAM del dispositivo con capacidad mínima de 256 Mb.

RNF4- Almacenamiento de dispositivo con capacidad mínima de 100Mb.

RNF5- Soporte para acceso a internet.

Eficiencia

RNF7- La aplicación debe soportar una cantidad muy alta de usuarios conectados simultáneamente.

Seguridad

RNF10- Debe tener una alta seguridad que para lograr se va a ser uso como guía el estándar de seguridad ISO 27001.

2.3.3 Historias de usuario

Las historias de usuario son una herramienta que agiliza la administración de requisitos, reduciendo la cantidad de documentos formales y tiempo necesarios. Forman parte de la fórmula de captura de funcionalidades definidas por (Alexander Menzinsky y otros 2018) de las tres Cs:

Card: cada historia de usuario se reduce hasta hacerla fácil de memorizar y de sintetizar en una tarjeta o post-it. La tarjeta sirve como recordatorio y promesa de una conversación posterior.

***Conversation*:** el equipo de desarrollo y el propietario del producto añaden criterios de aceptación a cada historia poco antes de su implementación. Los cambios son bienvenidos en agilidad, por lo que no tiene sentido profundizar en estos detalles antes. La situación puede variar mucho desde el momento en el que se sintetiza la funcionalidad en la tarjeta hasta que se implementa.

***Confirmation*:** el propietario del producto o usuario de negocio confirma que el equipo de desarrollo ha entendido y recogido correctamente sus requisitos revisando los criterios de aceptación. A veces se pueden presentar transformados en escenarios de pruebas.

Para la descripción de los requisitos funcionales de la propuesta de solución se define agrupar los requisitos en historia de usuario. A continuación, se muestran las historias de usuarios de “Crear archivo”, “visualizar archivos compartidos” y “buscar contactos”:

Tabla 3: *Historia de Usuario Crear archivo.*

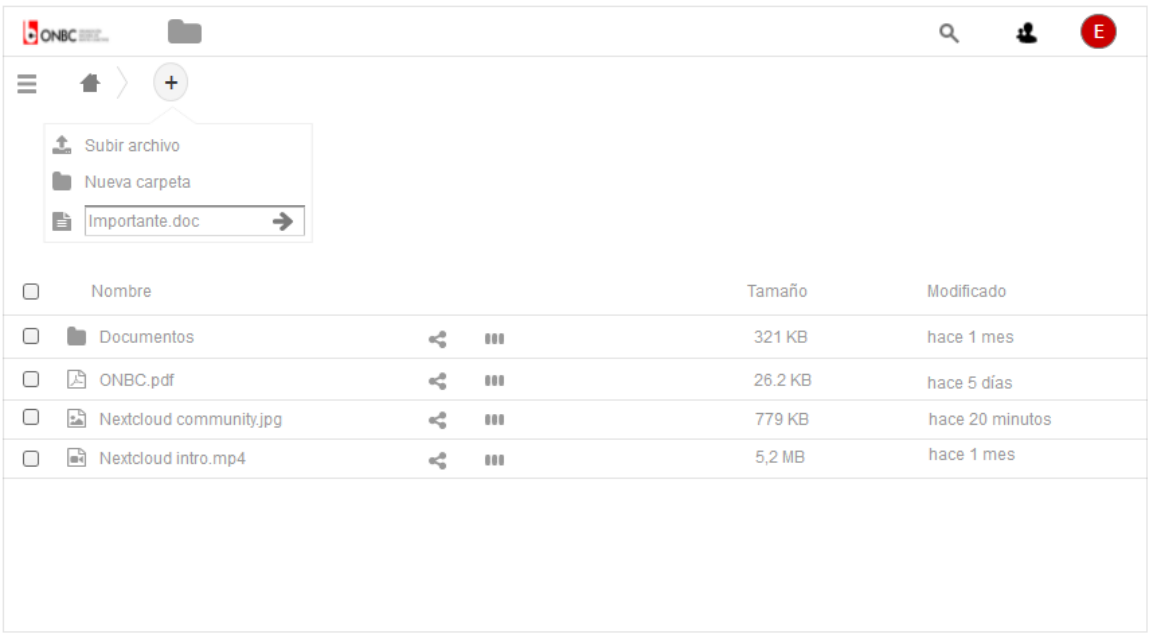
Historia de usuario	
Número: HU_1	**Nombre:** Crear archivo
Prioridad en negocio: Alta	
Descripción: Permite al usuario crear archivos para luego almacenarlos dentro del gestor de archivos.	
Prototipo: 	

Tabla 4: *Historia de Usuario Visualizar archivos compartidos.*

Historia de usuario	
Número: HU_2	**Nombre:** Visualizar archivos compartidos
Prioridad en negocio: Media	
Descripción: Permite mostrar los archivos que el usuario a compartido, empieza en el momento en que el usuario hace click en el botón de compartir.	
Prototipo	

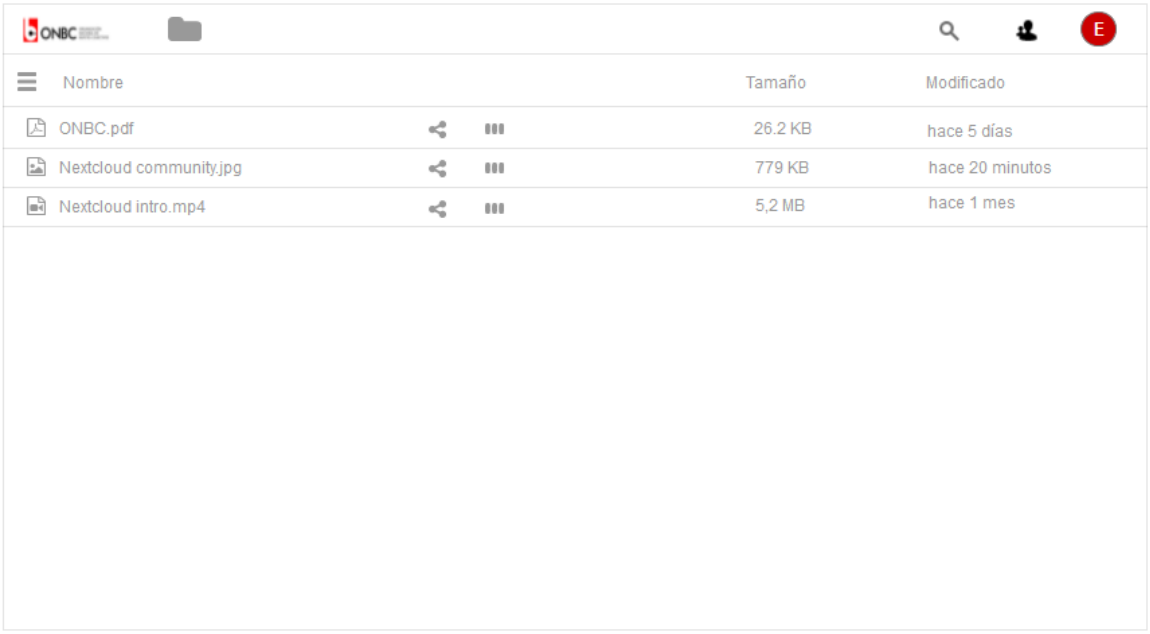
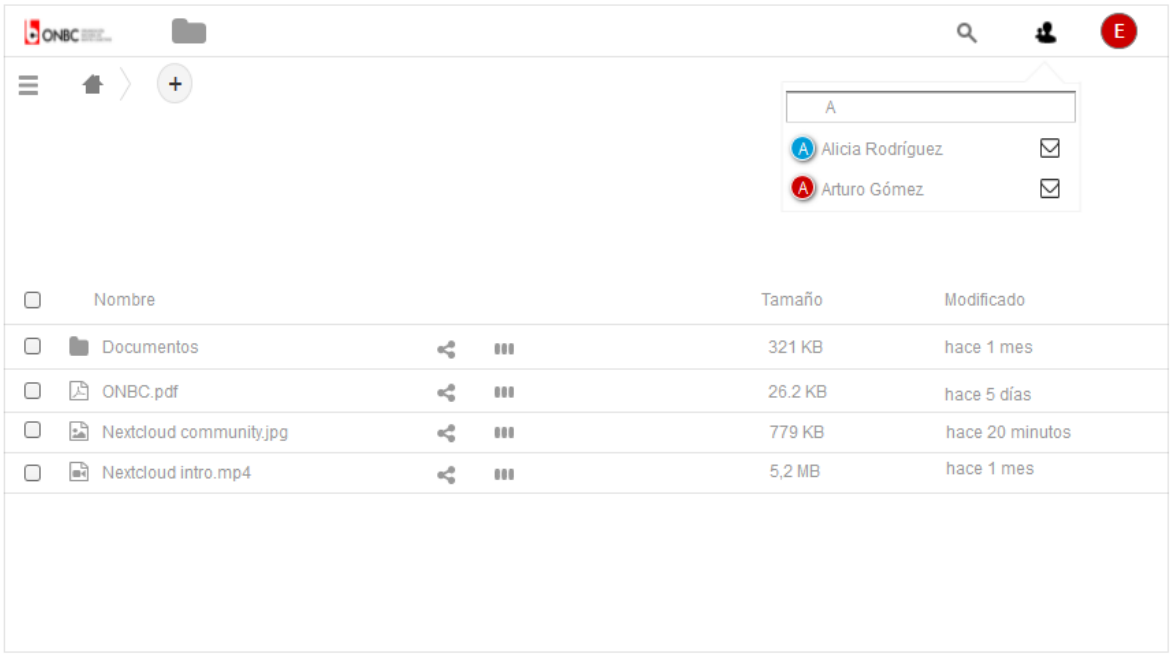


Tabla 5: *Historia de Usuario Buscar usuario.*

Historia de usuario	
Número: HU_3	**Nombre:** Buscar usuarios
Prioridad en negocio: Media	
Descripción: Permite al usuario buscar los contactos que se encuentran en el sistema para luego tener la posibilidad de mandarles un correo por otra plataforma.	
Prototipo	



Historia de usuario	
Número: HU_4	**Nombre:** Crear usuario usuarios
Prioridad en negocio: Alta	
Descripción: Permite al usuario con rol de administrador crear usuarios para el uso posterior.	
Prototipo	

2.4 Diseño del sistema web

El diseño permite modelar el sistema o producto que se va a construir. Este modelo se evalúa respecto de la calidad y su mejora antes de generar código; después, se efectúan pruebas y se involucra a

muchos usuarios finales. El diseño es el lugar en el que se establece la calidad del software (PRESSMAN, Roger S 2010). Esta actividad del ciclo de vida de ingeniería de software debe describir la arquitectura de software, donde se deben describir los componentes a un nivel de detalles que permitan su construcción.

2.4.1 Diagramas de clases del diseño

La estructura general de los diagramas de clases del diseño, de las hojas de consulta propuestas, están compuestos por páginas clientes que son construidas por páginas servidoras y que a su vez contienen formularios que muestran y capturan toda la información. Las páginas servidoras invocan métodos o responsabilidades en la clase controladora que según la acción solicitada pueden modificar las entidades. A continuación, se presentan los diagramas de clases de diseño, los cuales constituyen la base para su futura implementación, con el objetivo de lograr una comprensión más amplia de las hojas de consulta en cuestión.

Diagrama de diseño (Buscar usuario):

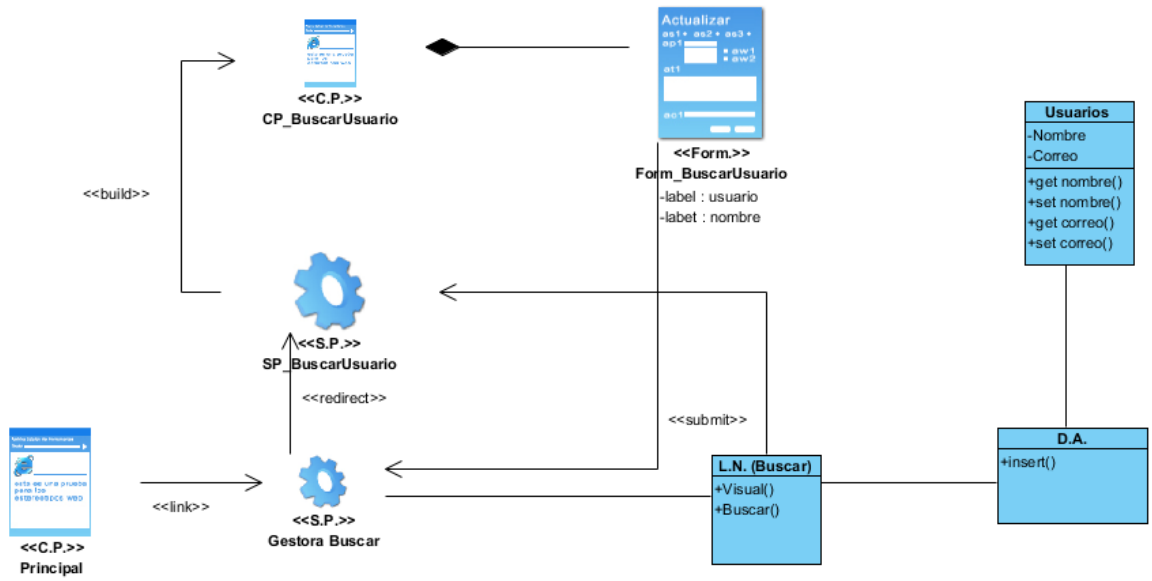


Figura 3:Diagrama de clase de diseño Buscar usuario.

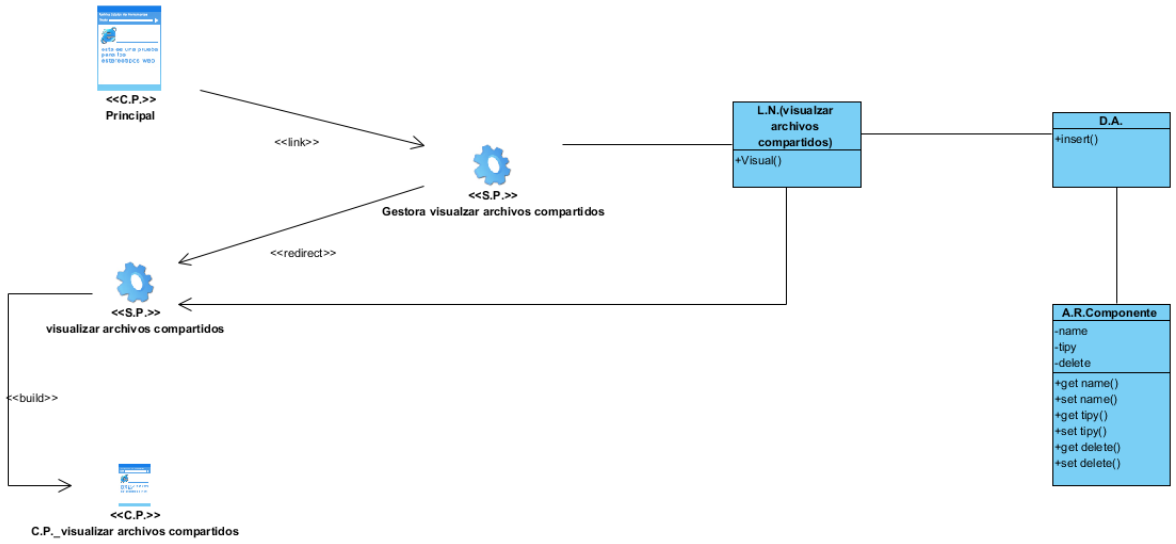


Figura 4: Diagrama de clase de diseño Visualizar archivos compartidos.

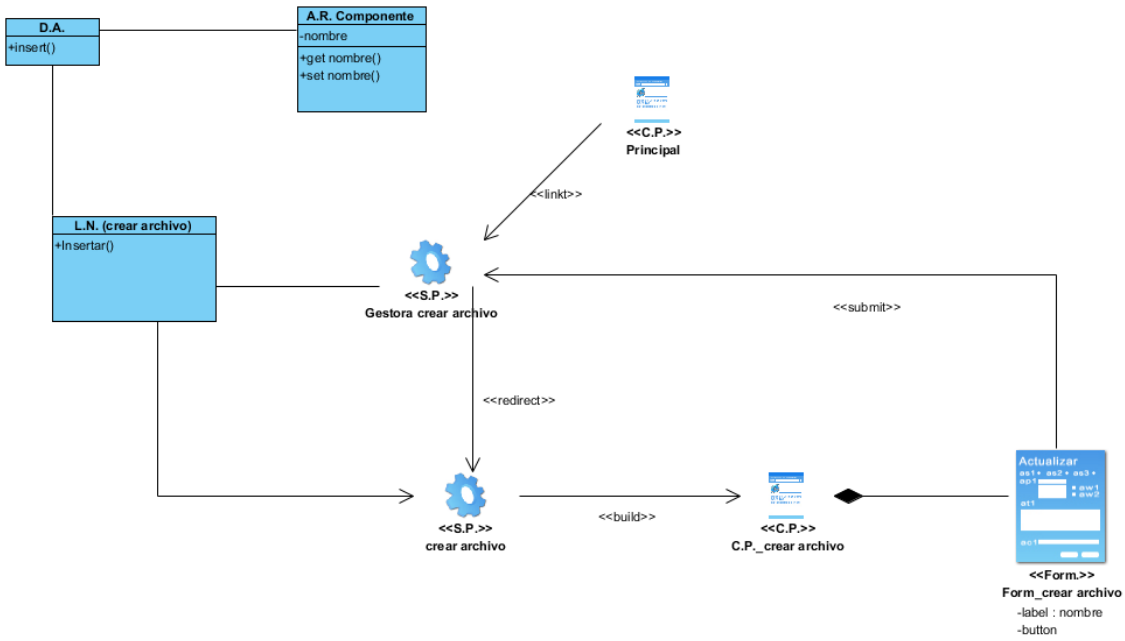


Figura 5: Diagrama de clase de diseño Crear archivo.

2.4.1 Diseño arquitectónico

El diseño arquitectónico es un proceso creativo en el que se intenta establecer una organización del sistema que satisfaga los requisitos funcionales y no funcionales del propio sistema. Debido a que es

un proceso creativo, las actividades dentro del proceso difieren radicalmente dependiendo del tipo de sistema a desarrollar, el conocimiento y la experiencia del arquitecto del sistema, y los requisitos específicos del mismo (Ian Somerville 2005).

El Modelo Vista Controlador (MVC) es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos:

Modelo: representación específica del dominio de la información sobre la cual funciona una aplicación.

Vista: presenta un formato adecuado para interactuar, usualmente un elemento de interfaz usuario.

Controlador: interacciona con el usuario y con la aplicación, ya que interpreta la información que el usuario provee (usualmente acciones) y provoca cambios en el modelo y probablemente en la vista (D r. Héctor Rafael Orozco Aguirre 2019).

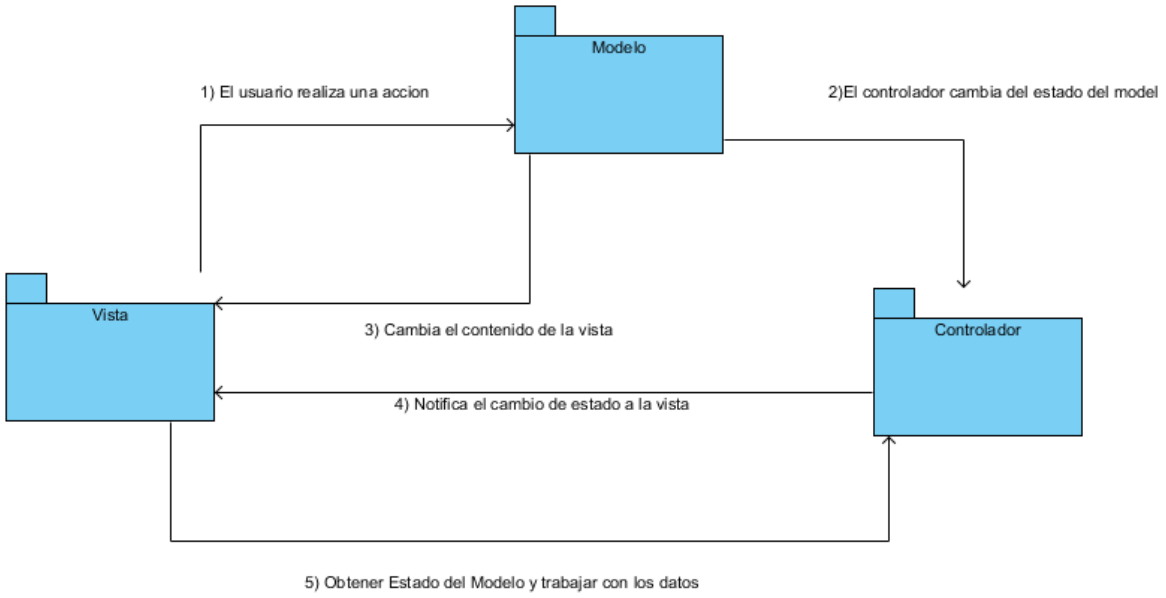


Figura 6: Diagrama de paquete de la arquitectura Modelo Vista Controlador.

2.4.2 Patrones de diseño

Los patrones de diseño están relacionados con el diseño de los objetos y frameworks de pequeña y mediana escala. Aplicables al diseño de una solución para conectar los elementos de gran escala que se definen mediante los patrones de arquitectura, y durante el trabajo de diseño detallado para cualquier aspecto del diseño local (CRAIG, Larman 2003).

Los patrones de diseño refinan los componentes basados en la experiencia obtenida. Son una solución estándar para un problema común de programación, ya que ayudan a mantener un código reutilizable y tener mayor control sobre los problemas recurrentes, por eso, son muy utilizados en el desarrollo de en múltiples aplicaciones.

Estos patrones se dividen en dos grupos:

Patrones GRASP

Los patrones GRASP constituyen un apoyo para la enseñanza que ayuda a uno a entender el diseño de objetos esencial, y aplica el razonamiento para el diseño de una forma sistemática, racional y explicable. Este enfoque para la comprensión y utilización de los principios de diseño se basa en los patrones de asignación de responsabilidades (CRAIG, Larman 2003).

Experto: este patrón es el encargado de asignar una responsabilidad al experto en información, es decir, la clase que posee la información necesaria y suficiente para ejecutar la responsabilidad asignada.

Bajo acoplamiento: este patrón consiste en mantener las clases lo menos relacionadas posible, de forma tal que, al producirse un cambio en alguna clase, se tenga el mínimo de repercusión en las otras.

Alta cohesión: cada elemento del diseño debe realizar una labor única dentro del sistema, lo cual expresa que la información que almacena una clase, debe ser coherente y estar lo más relacionada con ella posible. Este patrón es el encargado de asignar responsabilidades, de manera que la información que se almacena en una clase, sea la necesaria y esté bien delimitada.

Controlador: el patrón controlador actúa como intermediario entre una interfaz y el algoritmo que la implementa, es el encargado de controlar el flujo de datos de las funcionalidades que se le fueron asignadas y los eventos asociados a estas. Se ve evidenciado en la clase Gestora_crear_archivos.

Creador: este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. El mismo, es el encargado de asignar a las clases la responsabilidad de instanciar otra clase. En diagrama de clase crear archivos se ve evidenciado este patrón en SP crear_archivos.

Patrones GOF

Los patrones GOF los cuales se clasifican en tres grupos fundamentales: los patrones Creacionales que es donde se inicializan y configuran los objetos, los patrones Estructurales donde se separa la interfaz de la implementación y los patrones de Comportamiento los cuales describen el comportamiento entre las clases y los objetos.

***Adapter* (Estructural):** Adapta una interfaz, facilitando la implementación para que pueda ser utilizada por una clase, que, de otro modo, no podría utilizarla.

Prototype: Permite crear nuevos objetos por duplicación de objetos existentes llamados prototipos que disponen de la capacidad de clonación.

2.5 Modelo de datos

El modelo de datos se realiza si los requerimientos de software requieren la necesidad de crear, ampliar o hacer una interfaz con una base de datos, o si deben construirse y manipularse estructuras de datos complejas. Para ello se definen todos los objetos de datos que se procesan dentro del sistema, así como la relación entre ellos, mediante el diagrama entidad - relación (DER) (PRESSMAN, Roger S 2010).

A continuación, se encuentra representado el modelo entidad relación en la Figura 7 haciendo referencias a las bases de datos utilizadas en el sistema.

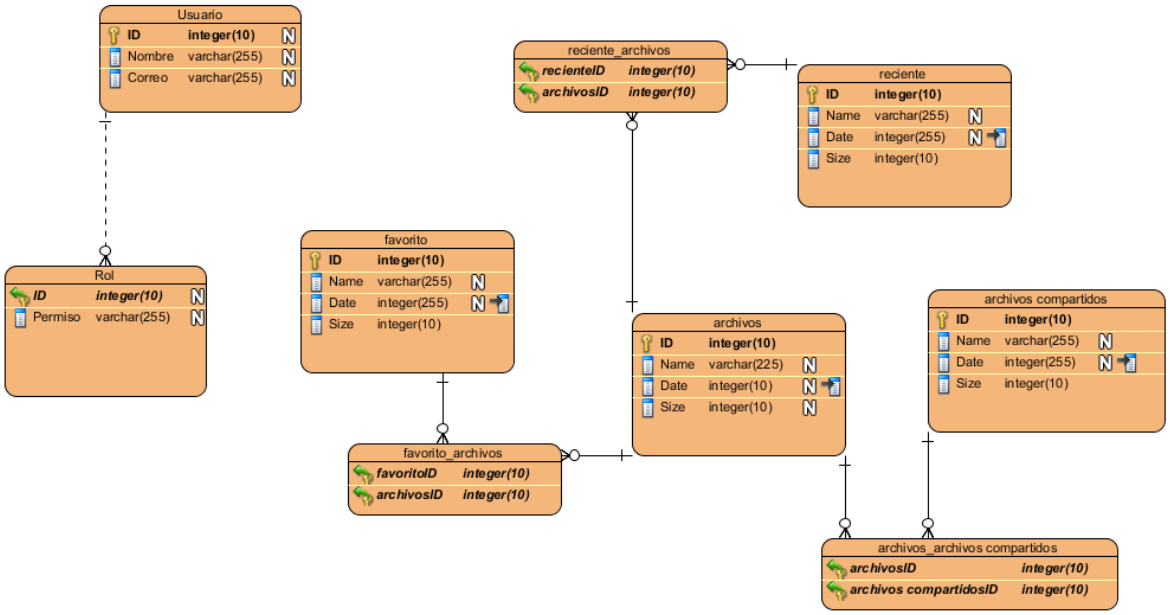


Figura 7: Diagrama Entidad-Relación.

Tabla 6: *Descripción de los atributos del Diagrama Entidad-Relación*

Atributo	**Tipo**	**Descripción**
id	Integer	Identificador único necesario en cada entidad para las referencias en las relaciones entre tablas.
permiso	Varchar	Indica el permiso que tiene el usuario creado.
size	Integer	Permite identificar el tamaño del archivo.
name	Varchar	Indica el nombre del archivo.
nombre	Varchar	Indica el nombre del usuario.

Conclusiones del capítulo

Luego de definir las características del sistema se puedo llegar a las siguientes conclusiones:

- ✓ El modelo del dominio describe cómo funciona el uso del sistema por parte de los abogados, aportando claridad sobre el contexto donde se usará la aplicación.
- ✓ Las entrevistas realizadas con el cliente permitieron la captura de 20 requisitos funcionales y 10 no funcionales, que fueron agrupados en 3 historias de usuario generando una visión para la creación del sistema.
- ✓ Rigiéndose por la metodología AUP vUCI en su escenario 4 y gracias al levantamiento de requisitos realizado se hace uso de las historias de usuario.
- ✓ La descripción de requisitos por procesos brindó la posibilidad de detallar toda la información que luego sirvió como punto de partida para el modelado de los diagramas pertenecientes a la etapa de diseño del proceso de desarrollo de software. Basado en el patrón arquitectónico MVC que posee el *framework* se modelan las clases del diseño y demás artefactos ingenieriles.

CAPÍTULO 3: VALIDACIÓN DE LA PROPUESTA

3.1 Estándares de codificación

El objetivo de los estándares de codificación de software es inculcar prácticas de programación probadas que conduzcan a un código seguro, confiable, comprobable y mantenible. Por lo general, esto significa evitar las prácticas de codificación inseguras conocidas o el código que puede causar un comportamiento impredecible (Arthur Hicken 2020).

Los estándares y estilos de codificación permiten que el código fuente de la solución que se esté en desarrollo posea mayor calidad, y que realizar el mantenimiento de esta se torne menos complejo.

Se tiene como idioma a utilizar el español, además que se debe tener en cuenta que las palabras no se acentuarán; ni se utilizará la letra “ñ” para evitar interpretaciones erróneas por parte de las PC, pues se encuentra en dependencia de las configuraciones que posea la misma.

Tabla 7: *Estándares y Codificación.*

Variables y constantes	
Aspectos generales	El nombre de la variable, debe permitir que con sólo leerlo se conozca el propósito de la misma.
Identación	
Inicio y fin de bloque	Se recomienda dejar dos espacios en blanco desde la instrucción anterior para el inicio y fin de bloque {}. Lo mismo sucede para el caso de las instrucciones *if, else, for, while, foreach*.
Aspectos generales	El identado debe ser de dos espacios por bloque de código. No se debe usar el tabulador; porque este puede variar según la PC o la configuración de dicha tecla. Los inicios ({) y cierre (}) de ámbito deber estar alineados debajo de la declaración a la que pertenecen y deben evitarse si hay sólo una instrucción. Nunca colocar ({) en la línea de un código cualquiera, esto requiere una línea propia.
Comentarios, separadores, líneas, espacios en blanco y márgenes	

Ubicación de comentarios	Al inicio de cada clase o función y al final de cada bloque de código. Se recomienda comentar al inicio de la clase o función especificando el objetivo de la misma.
Líneas en blanco	Se emplean antes y después de métodos, clases y estructuras. Se recomienda dejar una línea en blanco antes y después de la declaración de una clase o de una estructura y de la implementación de una función.
Espacios en blanco	Entre operadores lógicos y aritméticos. Se recomienda usar espacios en blanco entre estos operadores para una mayor legibilidad en el código.
Aspectos generales	Sobre el comentario. Se debe evitar comentar cada línea de código. Cuando el comentario se aplica a un grupo de instrucciones debe estar seguido de una línea en blanco. En caso de que se necesite comentar una sola instrucción se suprime la línea en blanco o se escribe a continuación de la instrucción. Sobre los espacios en blanco. No se debe usar espacio en blanco: Después del corchete abierto y antes del cerrado de un arreglo. Después del paréntesis abierto y antes del cerrado. Antes de un punto y coma.
Clases y Objeto	
Apariencia de clases y objetos	Primera letra en mayúscula. Los nombres de las clases deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing. Ejemplo: MiClase().
Apariencia de atributos	Primera letra en minúscula. El nombre que se le da a los atributos de las clases debe comenzar con la primera letra en minúscula, en caso de que sea un nombre compuesto se

	empleará notación CamellCasing. Ejemplo: especialistasAprobados.
Apariencia de las funciones	Para nombrar las funciones se debe tratar de utilizar verbos que denoten la acción que hace la función. Se empleará notación *PascalCasing*. Si son funciones que obtienen un dato se emplea el prefijo *get* y si fijan algún valor se emplea el prefijo set.
Aspectos generales	El nombre para las clases, objetos, atributos y funciones debe permitir que con solo leerlo se conozca el propósito de los mismos.

3.2 Modelo de despliegue

El modelo de despliegue muestra las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación. Los nodos representan objetos físicos existentes en tiempo de ejecución, sirven para modelar recursos que tiene memoria y capacidad de proceso y puede ser tanto ordenadores como dispositivos, memoria o personas (MELCHOR 2019).

En la figura que se muestra a continuación, se encuentran los componentes de hardware y sus relaciones para realizar el despliegue del sistema. Este modelo de despliegue está compuesto por tres nodos que representan un cómputo del usuario la aplicación web y los servidores de base de datos, la conexiones se establecen a través del cómputo hacia la aplicación web por el protocolo https y de la aplicación al servidor de base de datos por TCP:

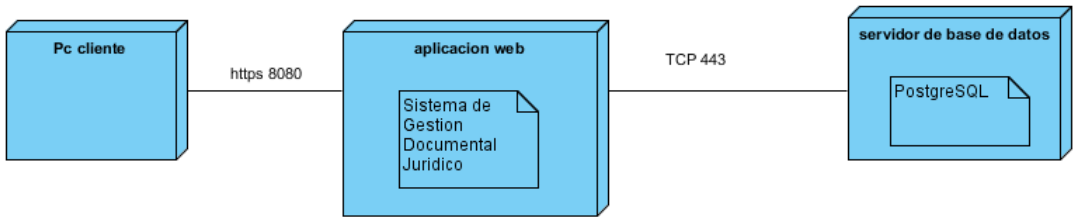


Figura 8: Diagrama de Despliegue.

3.3 Diagrama de componentes

Los diagramas de componentes UML representan las relaciones entre los componentes individuales del sistema mediante una vista de diseño estática. Pueden ilustrar aspectos de modelado lógico y físico.

En el contexto del UML, los componentes son partes modulares de un sistema independientes entre sí, que pueden reemplazarse con componentes equivalentes. Son auto contenidos y encapsulan estructuras de cualquier grado de complejidad. Los elementos encapsulados solo se comunican con los otros a través de interfaces. Los componentes no solo pueden proporcionar sus propias interfaces, sino que también pueden utilizar las interfaces de otros componentes, por ejemplo, para acceder a sus funciones y servicios. A su vez, las interfaces de un diagrama de componentes documentan las relaciones y dependencias en una arquitectura de software (IONOS 2020).

Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases del modelo de diseño, los mismos son creados, modificados o eliminados en el proceso de implementación y constituyen la versión del producto.

Las relaciones de dependencia se utilizan en los diagramas de componentes para indicar que un componente se refiere a los servicios ofrecidos por otro componente. Los distintos componentes pueden agruparse en paquetes según un criterio lógico y con vista a simplificar la implementación. Estos paquetes son estereotipados como <<subsistemas>>. Cada subsistema puede contener componentes y otros subsistemas.

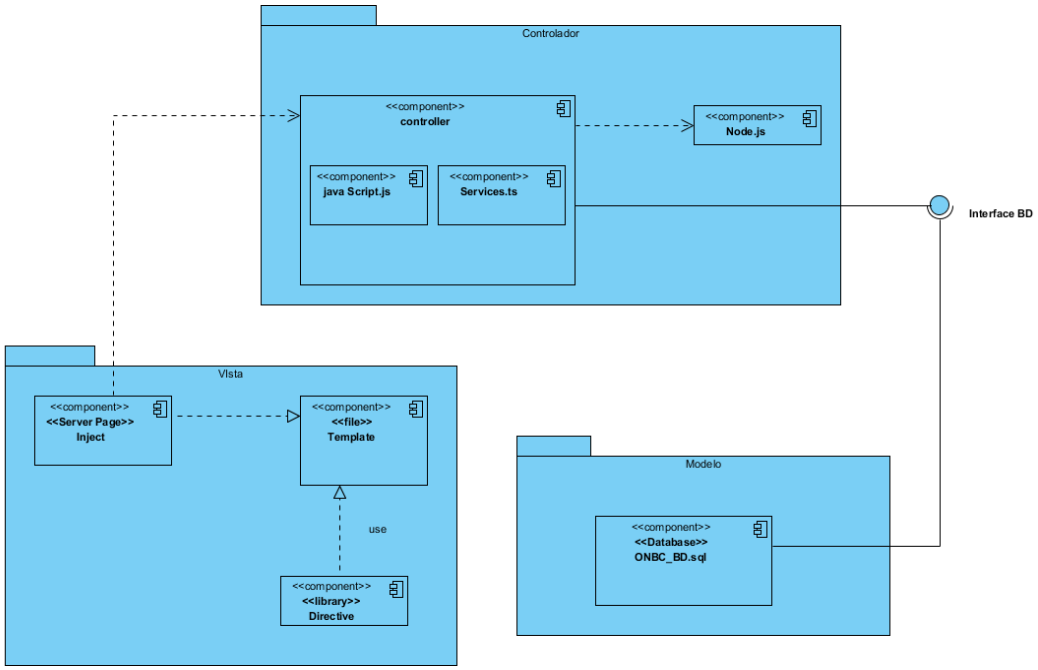


Figura 9: Diagrama de Componente.

3.4 Pruebas de software

Las pruebas de software son una serie de actividades que se realizan con el propósito de encontrar los posibles fallos de implementación, calidad o usabilidad de un programa u ordenador; probando el comportamiento del mismo. Una estrategia para las pruebas de software debe incluir pruebas a bajo nivel, que son necesarias para verificar que un pequeño segmento de código fuente se implementó correctamente, así como pruebas de alto nivel, que validan las principales funcionalidades del sistema a partir de los requerimientos del cliente (Ian Somerville 2005)

La estrategia de pruebas pudiera verse como un proceso en espiral, al igual que el proceso de desarrollo de software, comenzando desde la ingeniería del sistema, el levantamiento de requisitos, el diseño de los requerimientos y la implementación de los mismos. Al mismo nivel de cada una de estas etapas de desarrollo de software se realizan las pruebas del sistema, de validación, de integración y de unidad (Ian Somerville 2005). En la figura se muestra la representación descrita anteriormente:

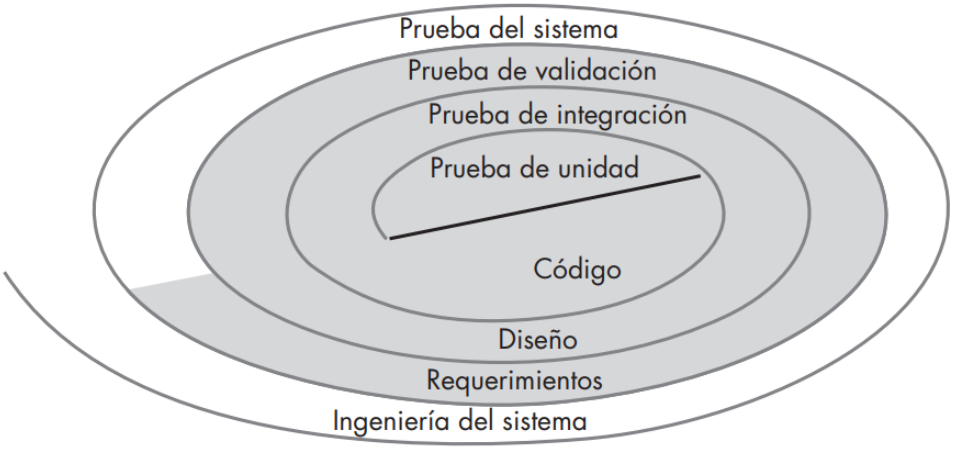


Figura 10: Estrategia de Pruebas (Ian Somerville 2005).

3.4.1 Pruebas del Sistema

Caja Negra

Las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software. O sea, la prueba de caja negra permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra intenta encontrar errores de las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación(PRESSMAN, Roger S 2010).

A continuación, se presenta el caso de prueba para las Historia de Usuario Crear usuario utilizando el método de caja negra bajo la técnica de partición de equivalencia.

Tabla 8: *Prueba de Sistema a la Historia de Usuario Crear usuario.*

Escenario	Descripción	Nombre	Correo	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Crear formulario crear usuario	El usuario introduce los datos correctamente.	Valor válido	Valor válido	Valor válido	Se adiciona una un nuevo usuario al sistema	El sistema verifica que no existan campos vacíos, guarda la

		Emilio	emilio@vcl.onbc			información y crear el usuario
EC 1.2 Datos incompletos	El usuario no introduce o selecciona datos requeridos.	Valor inválido	Valor inválido	N/A	Muestra un (*) indicando que existen datos obligatorios incompletos.	La aplicación muestra un mensaje de alerta que existen campos vacíos.
		Nada	Nada			
EC 1.3 Datos incorrectos	El usuario introduce datos de forma incorrecta.	Valor inválido	Valor inválido	N/A	Se analizan los campos incorrectos con un asterisco (*), al poner el puntero sobre el asterisco se visualizara el siguiente mensaje: "Caracteres incorrectos o sintaxis incorrecta".	El sistema muestra un mensaje indicando que los campos están llenados de forma incorrecta
		#$%&"()	#$%&"()			
EC 1.4 Cancelar operación	El usuario cancela la operación.	N/A	N/A	N/A	Se regresa a la interfaz crear usuario	El sistema regresa hacia la interfaz anterior

Caja Blanca

Pruebas de Caja Blanca: También suelen ser llamadas estructurales o de cobertura lógica. En ellas se pretende investigar sobre la estructura interna del código, exceptuando detalles referidos a datos de entrada o salida, para probar la lógica del programa desde el punto de vista algorítmico. Realizan un seguimiento del código fuente según se va ejecutando los casos de prueba, determinándose de manera concreta las instrucciones, bloques, etc. que han sido ejecutados por los casos de prueba.

En las pruebas de Caja Blanca se desarrollan casos de prueba que produzcan la ejecución de cada posible ruta del programa o módulo, considerándose una ruta como una combinación específica de condiciones manejadas por un programa.

Hay que señalar que no todos los errores de software se pueden descubrir verificando todas las rutas de un programa, hay errores que se descubren al integrar unidades del sistema y pueden existir errores que no tengan relación con el código específicamente (Martínez 2017).

En la siguiente imagen se observará un ejemplo de la prueba de seguridad caja blanca por pasos:

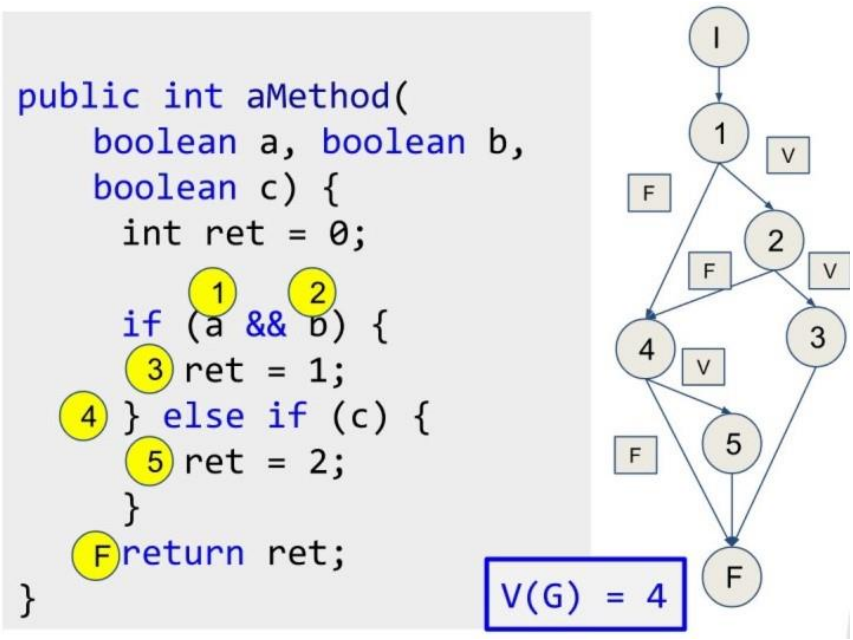


Figura 11: Método Caja Blanca. Fuente: Elaboración propia.

3.4.2 Pruebas de seguridad

Las pruebas de seguridad se podrían definir como el conjunto de actividades que se llevan a cabo para encontrar fallas y vulnerabilidades en aplicaciones web, buscando disminuir el impacto de ataques a ellas y pérdida de información importante. La seguridad en aplicaciones web busca asegurar la confidencialidad, disponibilidad e integridad de los datos y funciones que maneja el software, teniendo en cuenta el impacto que pueden tener fallas de seguridad según el contexto empresarial.

Como en todo proceso de calidad, se debe definir el alcance de las pruebas que se realizarán. Debido a que las pruebas de seguridad abarcan más que el producto, se deben tener en cuenta:

- ✓ Las personas
- ✓ El proceso
- ✓ El producto
- ✓ Políticas internas y normatividad (Silvia Margarita Díaz Díaz 2020).

Para llevar a cabo estas pruebas se elegirá a Nmap, dado que es una utilidad especializada en los sistemas con redes grandes y escanearlos de manera eficiente y en los paquetes de datos que serían dos cosas muy importantes a tener cuidado en la vulnerabilidad de nuestro sistema.

Nmap ('Network Mapper') es una utilidad (licencia) gratuita y de código abierto para el descubrimiento de redes y la auditoría de seguridad. Muchos administradores de sistemas y redes también lo encuentran útil para tareas como el inventario de la red, la gestión de programas de actualización del servicio y la supervisión del tiempo de actividad del host o del servicio. Nmap utiliza paquetes de IP sin procesar de formas novedosas para determinar qué hosts están disponibles en la red, qué servicios (nombre y versión de la aplicación) ofrecen esos hosts, qué sistemas operativos (y versiones de SO) están ejecutando, qué tipo de filtros de paquetes / firewalls están en uso y decenas de otras características. Fue diseñado para escanear rápidamente redes grandes, pero funciona bien contra hosts únicos. Nmap se ejecuta en todos los principales sistemas operativos de computadoras, y los paquetes binarios oficiales están disponibles para Linux, Windows y Mac OS X. Además del ejecutable clásico de línea de comandos Nmap, la suite Nmap incluye una GUI avanzada y un visor de resultados (Zenmap), una herramienta de depuración, redirección y transferencia de datos flexible (Ncat), una utilidad para comparar resultados de escaneo (Ndiff) y una herramienta de análisis de respuesta y generación de paquetes (Nping) (Sergio De Luz 2021).

3.4.3 Pruebas de rendimiento

Las pruebas de rendimiento son una clase de pruebas que se implementan y se ejecutan para caracterizar y evaluar las características relacionadas con el rendimiento del destino de la prueba, como los perfiles de tiempo, el flujo de ejecución, los tiempos de respuesta y la fiabilidad y los límites operativos. A lo largo del ciclo de vida del desarrollo de software (SDLC), se implementan Al principio de las iteraciones de la arquitectura, las pruebas de rendimiento se centran en identificar y eliminar los

cuellos de botella de rendimiento relacionados con la arquitectura. En las iteraciones de la construcción, los tipos adicionales de pruebas de rendimiento se implementan y ejecutan para ajustar el software y el entorno (optimización de los recursos y el tiempo de respuesta), y para verificar que las aplicaciones y el sistema manejan de manera aceptable las condiciones de estrés y de carga elevada, como grandes números de transacciones, clientes y/o volúmenes de datos (IBM 2006(IBM 2006).

Se va a proponer realizar las pruebas de carga y estrés dado el flujo alto de usuarios e información que se encontraran circulando en la aplicación web de manera simultánea. Según (Dotcom-Monitor 2020) una prueba de carga es una prueba planificada para realizar un número especificado de solicitudes a un sistema con el fin de probar la funcionalidad del sistema bajo niveles específicos de solicitudes simultáneas. Una prueba de carga garantiza que un sistema web pueda controlar un volumen de tráfico esperado y, por lo tanto, a veces se conoce como pruebas de volumen. El objetivo de una prueba de carga es demostrar que un sistema puede controlar el volumen esperado con una degradación del rendimiento mínima a aceptable. Los evaluadores deben definir el umbral de degradación del rendimiento aceptable como un valor que se considera aceptable para el usuario final para que los usuarios no reboten desde el sitio y una prueba de esfuerzo es una prueba diseñada para aumentar el número de solicitudes simultáneas en un sistema más allá de un punto donde el rendimiento se degrada, posiblemente incluso hasta el punto de fallo completo. Cuando una prueba de carga se agotará en el número de usuarios simultáneos, una prueba de esfuerzo básica seguirá aumentando la carga en el sistema hasta que los recursos se sobrecarguen. Esto lleva al sistema a un estado de error potencial para ver cómo lo maneja el sistema y si el sistema puede realizar una recuperación correcta.

Según(Microsoft Developer Network 2021), la metodología de las pruebas de rendimiento consiste en las siguientes actividades:

1. Identificar el entorno de pruebas. Identificar el entorno físico de pruebas y el entorno de producción, así como las herramientas y recursos de que dispone el equipo de prueba. El entorno físico incluye hardware, software y configuraciones de red. Tener desde el principio un profundo conocimiento de todo el entorno de prueba permite diseños de pruebas más eficientes. Facilita también la planificación y ayuda a identificar problemas en las pruebas en fases tempranas del proyecto. En algunas situaciones, este proceso debe ser revisado periódicamente durante todo el ciclo de vida del proyecto.

2. Identificar los criterios de aceptación de rendimiento. Determinar el tiempo de respuesta, el rendimiento, la utilización de los recursos y los objetivos y limitaciones. En general, el tiempo de

respuesta concierne al usuario, el rendimiento al negocio, y la utilización de los recursos al sistema. Identificar cuáles serían criterios de éxito de rendimiento del proyecto para evaluar qué combinación de la configuración da lugar a un funcionamiento óptimo.

3. Planificar y diseñar las pruebas. Identificar los principales escenarios, determinar la variabilidad de los usuarios y la forma de simular esa variabilidad, definir los datos de las pruebas y establecer las métricas a recoger. Consolidar esta información en uno o más modelos de uso del sistema a implantar, ejecutarlo y analizarlo.

4. Configurar el entorno de prueba. Preparar el entorno de prueba, las herramientas y recursos necesarios para ejecutar cada una de las estrategias, así como las características y componentes disponibles para la prueba. Asegurarse de que el entorno de prueba se ha preparado para la monitorización de los recursos según sea necesario.

5. Aplicar el diseño de la prueba. Desarrollar las pruebas de rendimiento de acuerdo con el diseño del plan.

6. Ejecutar la prueba. Ejecutar y monitorizar las pruebas. Validar las pruebas, los datos de las pruebas y recoger los resultados. Ejecutar pruebas válidas para analizar, mientras se monitoriza la prueba y su entorno.

7. Analizar los resultados, realizar un informe y repetirlo. Consolidar y compartir los resultados de la prueba. Analizar los datos, tanto individualmente como con un equipo multidisciplinario. Volver a priorizar el resto de las pruebas y a ejecutarlas en caso de ser necesario. Cuando todas las métricas estén dentro de los límites aceptados, ninguno de los umbrales establecidos haya sido rebasados y toda la información deseada se ha reunido, las pruebas han acabado para el escenario definido por la configuración.

Para realizar estas pruebas se utilizará la herramienta JMeter la cual la se ejecutará de la siguiente manera. En JMeter, un plan de pruebas es una jerarquía de componentes en forma de árbol (panel de control de la izquierda). Cada nodo del árbol es un componente. A su vez, un componente es una instancia de un tipo de componente en la que quizás se han configurado algunas de sus propiedades (en el panel de control de la derecha).

Los diferentes componentes de los que puede constar un plan de pruebas son:

- Test Plan. Es el tipo de componente que representa la raíz del árbol.
- *Thread Group*. Representa un grupo de usuarios. En JMeter cada *thread* es un usuario virtual.
- *Controllers (Sampler, Logic Controler*). Los *samplers* realizan peticiones contra la aplicación y los *logic controlers* establecen el orden en que se ejecutan éstos.
- *Config Element*. Establecen propiedades de configuración que se aplican a los *samplers* a los que afectan.
- *Assertion*. Comprueban condiciones que aplican a las peticiones que realizan contra la aplicación los *samplers* a los que afectan.
- *Listeners*. Recopilan datos de las peticiones que realizan los *samplers* a los que afectan.
- *Timer*. Añaden tiempo extra a la ejecución de las peticiones que realizan contra la aplicación los *samplers* a los que afectan
- Pre-*Processor element*. Realizan acciones o establecen configuraciones previas a la ejecución de los *samplers* a los que afectan.
- Post-*Processor element.* Realizan acciones o establecen configuraciones posteriormente a la ejecución de los *samplers* a los que afectan (Juan Francisco Sánchez 2018).

Conclusiones del capítulo

Luego de llevar a cabo la implementación de la propuesta de solución y elaborada la estrategia de pruebas para la misma se puede concluir que:

- ✓ Durante la confección del capítulo que recién concluye se abordaron cuestiones pertenecientes a la etapa de implementación y validación de la propuesta. Obteniéndose los diagramas de componentes correspondientes y el modelo de datos.
- ✓ Fueron identificados los estándares y estilos de codificación a utilizar para la implementación de los componentes definidos y la propuesta del aseguramiento de la seguridad del sistema, así como las pruebas de rendimiento.

CONCLUSIONES

Luego de realizada la presente investigación, se concluye lo siguiente:

- ✓ El análisis de los sistemas homólogos permitió definir las características que debe poseer la propuesta de solución.
- ✓ El análisis de las herramientas brindará la posibilidad de desarrollar la propuesta de solución bajo las mismas tecnologías en las que está soportada el Sistema de Gestión Documental Jurídico, facilitando así su posterior integración.
- ✓ Con la propuesta de solución se describe como la implementación de dicho sistema contribuiría a la gestión de la documentación utilizada por los abogados en la organización nacional de bufetes colectivos
- ✓ Las futuras ejecuciones de las pruebas, permitirán detectar y corregir deficiencias presentes del sistema, validando así la seguridad del sistema y el rendimiento a partir de las propuestas de pruebas de seguridad y de las pruebas de rendimiento.

RECOMENDACIONES

Para futuras investigaciones se recomiendan las siguientes acciones:

Implementar el Sistema de gestión documental jurídico para lograr su funcionamiento y emigrar el sistema que se encuentra en uso.

Realizar todas las pruebas de software propuestas en el presente trabajo para lograr el funcionamiento óptimo del Sistema de gestión documental jurídico.

REFERENCIAS

ALEJANDRO HERNANDEZ., 2017. Concepto de TIC: Tecnologías de la Información de la Comunicación. [en línea]. [Consulta: 12 junio 2021]. Disponible en: https://economiatic.com/concepto-de-tic/.

ALEXANDER MENZINSKY Y OTROS, 2018. *Historias de Usuario Ingeniería de Requisitos Ágil*. 2018. S.l.: s.n.

ALEXANDER NÁPOLES TERRY, 2016. GDA Xabal excriba 3.1 para la gestión documental - PDF Descargar libre. [en línea]. [Consulta: 28 junio 2021]. Disponible en: https://docplayer.es/23359607-Gda-xabal-excriba-3-1-para-la-gestion-documental.html.

ALLENDE, 2021. ▷ Miro, plataforma para crear esquemas y diagramas. *Creatividad.Cloud* [en línea]. [Consulta: 24 noviembre 2021]. Disponible en: https://www.creatividad.cloud/miro-plataforma-para-crear-esquemas-y-diagramas-visuales-de-forma-colaborativa-y-en-tiempo-real/.

ALVARO HUMBERTO CISNEROS, 2017. Conceptos básicos y generales de la programación. [en línea]. [Consulta: 29 junio 2021]. Disponible en: https://www.evirtualcisneros.xyz/articulos/8tecnologia/25-conceptos-basicos-y-generales-de-la-programacion.

ÁLVARO TOLEDO, 2018. CakePHP (Windows). *Uptodown.com* [en línea]. [Consulta: 30 junio 2021]. Disponible en: https://cakephp.uptodown.com/windows.

ARTHUR HICKEN, 2020. Estándares de codificación de software y pautas de programación. [en línea]. [Consulta: 26 noviembre 2021]. Disponible en: https://es.parasoft.com/blog/an-ounce-of-prevention-software-safety-security-through-coding-standards/.

ASALE, R.- y RAE, 2021. metodología | Diccionario de la lengua española. *«Diccionario de la lengua española» - Edición del Tricentenario* [en línea]. [Consulta: 28 junio 2021]. Disponible en: https://dle.rae.es/metodología.

COMUNIDAD BARATZ, 2015. Los 10 beneficios de la gestión documental en las organizaciones. *Comunidad Baratz,* [en línea]. [Consulta: 15 junio 2021]. Disponible en: https://www.comunidadbaratz.com/blog/los-10-beneficios-de-la-gestion-documental-en-las-organizaciones/.

CONSEJO DE ESTADO, 2009. *DECRETO-LEY No. 265/2009 "DEL SISTEMA NACIONAL DE ARCHIVOS DE LA REPÚBLICA DE CUBA* [en línea]. 5 mayo 2009. S.l.: Gaceta Oficial de la República de Cuba. Disponible en: http://www.archivo.lastunas.cu/Datos/Decreto-Ley-265.PDF.

CRAIG, LARMAN, 2003. *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* 2da edicion. Madrid: s.n. ISBN 84-205-3438-2.

CRUNCHBASE, 2008. Disponible en: https://techcrunch.com/2008/03/11/dropbox-the-online-storage-solution-weve-been-waiting-for/.

D R. HÉCTOR RAFAEL OROZCO AGUIRRE, 2019. *MODELO VISTA CONTROLADOR (MVC) Y WEB ARCHIVES (WARS)*. 2019. S.l.: s.n.

DAVID OCHOBITS, 2019. *Seafile: Nuestro propio servidor de almacenamiento* [en línea]. 23 mayo 2019. S.l.: s.n. Disponible en: https://colaboratorio.net/davidochobits/sysadmin/2019/seafile-nuestro-propio-servidor-de-almacenamiento/.

DERRIK DIENER, 2017. *https://www.addictivetips.com/ubuntu-linux-tips/set-up-your-own-cloud-storage-on-linux-next-cloud/* [en línea]. 29 mayo 2017. S.l.: s.n. Disponible en: https://www.agenda.si/fileadmin/datoteke/dokumenti/Alfresco_PDF/Flyer_Nextcloud_2016_angl.pdf.

DOTCOM-MONITOR, 2020. Pruebas de carga vs pruebas de estrés. *LoadView* [en línea]. [Consulta: 26 noviembre 2021]. Disponible en: https://www.loadview-testing.com/es/pruebas-de-carga-vs-pruebas-de-estres/.

DR. FRANCISCO JOSÉ GARCÍA PEÑALVO, 2018. *Ciencia de la Computación e Inteligencia Artificial* [en línea]. 2018. S.l.: s.n. Disponible en: https://repositorio.grial.eu/bitstream/grial/1228/1/07-rep.pdf.

ELIBETH EDUARDO GARCÍA, 2016. ➤ Dropbox alcanza los 500 millones de usuarios| CIOAL The Standard IT. [en línea]. [Consulta: 28 junio 2021]. Disponible en: https://thestandardcio.com/2016/03/08/dropbox-alcanza-los-500-millones-usuarios/.

FERNANDO MACHUCA, 2021. ¿Qué es Python? El lenguaje de programación del 2021. [en línea]. [Consulta: 28 junio 2021]. Disponible en: https://www.crehana.com/blog/web/que-es-python/.

FRANCISCO JAVIER PALAZÓN, 2017. Strato HiDrive. Disco duro on line pensado para uso personal y profesional. [en línea]. [Consulta: 28 junio 2021]. Disponible en: https://revistabyte.es/analisis/strato-hidrive/.

IAN SOMERVILLE, 2005. Ingeniería del software - Ian Sommerville - Google Libros. [en línea]. [Consulta: 5 octubre 2021]. Disponible en: https://books.google.es/books?hl=es&lr=&id=gQWd49zSut4C&oi=fnd&pg=PA1&dq=Ian+somerville+ultima+edicion&ots=s824usszri&sig=N-8lBUiPjkpul5c_ZaUpq6LToFw#v=onepage&q=Ian%20somerville%20ultima%20edicion&f=false.

IBM, 2006. Concepto: Prueba de rendimiento. [en línea]. [Consulta: 26 noviembre 2021]. Disponible en: https://cgrw01.cgr.go.cr/rup/RUP.es/LargeProjects/core.base_rup/guidances/concepts/performance_testing_37A31809.html.

INFOTECARIOS, 2021. Gestión de la Información vs Gestión del Conocimiento - Evaluando Software. [en línea]. [Consulta: 10 octubre 2021]. Disponible en: https://www.evaluandosoftware.com/gestion-de-la-informacion-vs-gestion-del-conocimiento/.

IONOS, 2020. Diagrama de componentes: modelado eficiente de sistemas con módulos de software. *IONOS Digitalguide* [en línea]. [Consulta: 27 noviembre 2021]. Disponible en: https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/diagrama-de-componentes/.

JIMÉNEZ, A.D. y PÉREZ, A.G., 2020. La gestión documental en función de la gobernanza de la información. El caso de la Universidad Central "Marta Abreu" de las Villas. *SAPIENTIAE*, vol. 6, no. 1, pp. 70-85. ISSN 2184-061X, 2183-5063.

JORGE CANO, 2020. Angular: Mucho más que un framework. *SG Buzz* [en línea]. [Consulta: 29 junio 2021]. Disponible en: https://sg.com.mx/revista/56/angular.

JOSÉ CARLOS SÁNCHEZ DOMÍNGUEZ y PATRICIA RODRÍGUEZ DAPENA, 2003. *Verificación de los requisitos no funcionales en el software crítico*. S.l.: s.n.

JUAN FRANCISCO SÁNCHEZ, 2018. Pruebas de rendimiento con JMeter. Ejemplos básicos. *SDOS* [en línea]. [Consulta: 26 noviembre 2021]. Disponible en: https://www.sdos.es/blog/pruebas-de-rendimiento-con-jmeter-ejemplos-basicos.

JUAN PABLO ALONSO VERANO, 2018. *LA GESTIÓN DOCUMENTAL COMO MODELO DE NEGOCIO.* [en línea]. 1 enero 2018. S.l.: s.n. Disponible en: https://ciencia.lasalle.edu.co/cgi/viewcontent.cgi?article=1048&context=maest_gestion_documental.

JULIÁN PÉREZ PORTO Y MARÍA MERINO, 2009. Definición de documento — Definicion.de. *Definición.de* [en línea]. [Consulta: 29 junio 2021]. Disponible en: https://definicion.de/documento/.

KRISTIN FJOLA TOMASDOTTIR Y OTROS, 2018. *The Adoption of JavaScript Linters in Practice: A Case Study on ESLint*. 19 septiembre 2018. S.l.: s.n.

LARAVEL, 2021. Installation - Laravel - The PHP Framework For Web Artisans. [en línea]. [Consulta: 29 junio 2021]. Disponible en: https://laravel.com/docs/8.x#why-laravel.

MARÍA ESTELA RAFFINO, 2020. Institución - Concepto, clasificación y características. [en línea]. [Consulta: 29 junio 2021]. Disponible en: https://concepto.de/institucion/.

MARTÍNEZ, E.S., 2017. Procedimiento para realizar pruebas de Caja Blanca. *Informática Jurídica*.

MARTÍNEZ GONZÁLEZ, B.A., ALFARO RIVERA, J.A. y RAMÍREZ MONTOYA, M.S., 2012. Procesos de gestión de información y construcción de conocimiento en la formación de investigadores educativos a través de ambientes a distancia. *Sinéctica*, no. 38, pp. 1-15. ISSN 1665-109X.

MDN CONTRIBUTORS, 2021. JavaScript | MDN. [en línea]. [Consulta: 28 junio 2021]. Disponible en: https://developer.mozilla.org/es/docs/Web/JavaScript.

MEGA, 2021. Documentation - MEGA. [en línea]. [Consulta: 30 junio 2021]. Disponible en: https://mega.nz/doc.

MELCHOR, 2019. *tutorial-diagramas-de-despliegue* [en línea]. 2019. S.l.: s.n. Disponible en: https://prezi.com/e_gpb7xev_im/tutorial-diagramas-de-despliegue/.

MICROSOFT DEVELOPER NETWORK, 2021. Pruebas de rendimiento web programadas - Visual Studio (Windows). .

MILTON RAFAEL VALAREZO PARDO Y OTROS, 2018. *COMPARACIÓN DE TENDENCIAS TECNOLÓGICAS EN APLICACIONES WEB*. 14 septiembre 2018. S.l.: s.n.

MOLINA HERNÁNDEZ, Y., GRANDA DIHIGO, A. y VELÁZQUEZ CINTRA, A., 2019. Los requisitos no funcionales de software. Una estrategia para su desarrollo en el Centro de Informática Médica. *Revista Cubana de Ciencias Informáticas*, vol. 13, no. 2, pp. 77-90. ISSN 2227-1899.

NEXTCLOUD GMBH, 2017. Wayback Machine. [en línea]. [Consulta: 28 junio 2021]. Disponible en: https://web.archive.org/web/20170922002126/http://ubucon.org/media/cms_page_media/32/nextcloud_talk.pdf.

NICK ANTONOPOULOS, LEE GILLAM, 2017. *Cloud Computing*. S.l.: s.n.

PMT, G., 2019. Metodología tradicional. *Gestión de Proyectos* [en línea]. [Consulta: 28 junio 2021]. Disponible en: https://pmtgrupoeafit.wixsite.com/gestion-proyectos/post/metodología-tradicional.

PRESSMAN, ROGER S, 2010. *Software engineering: a practitioner's approach*. S.l.: Palgrave Macmillan.

QUALITY DEVS, 2019. ¿Qué es Symfony? Y por qué es el mejor framework para Apps. *Quality Devs* [en línea]. [Consulta: 29 junio 2021]. Disponible en: https://www.qualitydevs.com/2019/08/05/que-es-symfony/.

SAS, E. -dea N., 2021. Adquiere Lucidchart en Colombia. [en línea]. [Consulta: 24 noviembre 2021]. Disponible en: https://www.e-dea.co/lucidchart-colombia.

SEALPATH, 2020. Protegiendo la información en sus tres estados - Blog Sealpath. *Sealpath* [en línea]. [Consulta: 10 octubre 2021]. Disponible en: https://www.sealpath.com/es/tres_estados_info/.

SERGIO DE LUZ, 2021. Mejores escáner de vulnerabilidades gratis para hacker ético. *RedesZone* [en línea]. [Consulta: 7 diciembre 2021]. Disponible en: https://www.redeszone.net/tutoriales/seguridad/mejores-escaner-vulnerabilidades-gratis-hacker/.

SILVIA MARGARITA DÍAZ DÍAZ, 2020. *Pruebas de seguridad en aplicaciones web como imperativo en la calidad de desarrollo del software* [en línea]. 2020. S.l.: s.n. Disponible en: https://www.unab.edu.co/sites/default/files/.

VANESSA ROSSELLÓ VILLÁN, 2019. *Las metodologías ágiles más utilizadas y sus ventajas dentro de la empresa* [en línea]. 15 marzo 2019. S.l.: s.n. Disponible en: https://www.iebschool.com/blog/que-son-metodologias-agiles-agile-scrum/.

VISUAL PARADIGM, 2021. Ideal Modeling & Diagramming Tool for Agile Team Collaboration. [en línea]. [Consulta: 29 junio 2021]. Disponible en: https://www.visual-paradigm.com/.

ANEXOS

Anexo 1: Encuesta realizada para identificar las deficiencias en el manejo de la información.

La entrevista se le fue realizada a trabajadores de abogacía de diferentes bufetes ubicados en la capital como el bufete colectivo de playa el bufete José Martí y el Bufete colectivo ubicado en 5ta y F
1- ¿Utiliza usted dispositivos USB o discos duros externos como medios de almacenamiento de contenidos de trabajo?
2- ¿Presenta con frecuencia problemas con estas tecnologías en cuanto a pérdida, roturas y otros motivos?
3- ¿Los problemas relacionados con estas tecnologías han afectado su desempeño laboral?
4- ¿En cuánto tiempo se vio afectado su trabajo debido a los inconvenientes descritos antes?