Universidad de las Ciencias Informáticas


Facultad 3

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas.

# Título: Desarrollo del Módulo Gestión de Fondos Documentales para su digitalización del sistema para la obtención de objetos digitales con valor legal (DIGILEX).

**Autor:**

Marcelo Adrián del Rio Rodríguez

**Tutores:**

Mcs. Yarina Amoroso Fernández

Ing. Angel Miguel Morciego Lezcano

La Habana. Junio, 2017

## DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____
Autor

Marcelo Adrián del Rio Rodríguez

_____
Tutor

Ing. Angel Miguel Morciego Lezcano

_____
Tutor

Msc. Yarina Amoroso Fernández

**AGRADECIMIENTOS**

*A todos mis familiares y amigos que hicieron posible este momento.*

**DEDICATORIA:**

*A mi madre.*

**RESUMEN**

El presente trabajo se desarrolla en el proyecto investigativo “Sistema para la obtención de documentos digitales con valor legal “como parte del Centro de Gobierno Electrónico el cual pretende generar objetos digitales con valor legal partiendo de diferentes tipos documentales mediante una solución de software para así mejorar la calidad de los servicios ofrecidos a los ciudadanos por parte del Gobierno Electrónico. Tiene como objetivo general, desarrollar un módulo de gestión de uno o varios fondos documentales del sistema DIGILEX, que permita el proceso de digitalización. Se hace un estudio de los conceptos teóricos fundamentales relacionados a la investigación como la digitalización de documentos, gestión documental, fondo documental y documento con valor legal, la tipología documental, así como la clasificación e importancia de esta. Incluye un amplio bosquejo sobre los sistemas existentes para la digitalización de documentos, tanto nacionales como internacionales además se abordan los elementos más importantes sobre metodologías, lenguajes de modelado de software, marco de trabajo y herramientas para el desarrollo, tales como: AUP-UCI, Visual Paradigm, UML, NetBeans, PostgresSQL, entre otros. Se presentan los distintos diagramas de clases del diseño y los diagramas de paquetes; los cuales ayudan al mejor entendimiento sobre cómo funciona el sistema. Se hace alusión a las métricas aplicadas para evaluar el diseño, así como las estrategias de pruebas llevadas a cabo para lograr una mejor calidad y a las conclusiones.

**PALABRAS CLAVES**

Digitalización de documentos, Gestión documental, Fondo documental, Documento con valor legal y la tipología documental.

# Índice General

# Índice de Figuras

# Índice de Tablas

## Introducción

La adopción de las nuevas tecnologías y en particular de aquellas relacionadas con la información (TIC) ha sido de gran utilidad para las entidades a todos los niveles de la sociedad, facilitando la posibilidad de acceder, organizar y comunicar información relevante con mayor eficiencia. A nivel gubernamental la aplicación de las tecnologías de la información y la comunicación (TIC) al funcionamiento del sector público, con el objetivo de incrementar la eficiencia, la transparencia y la participación ciudadana, define el surgimiento de Gobierno Electrónico.

El Gobierno Electrónico es una nueva forma de gobierno que a través de su enfoque innovador sitúan las TIC como un elemento de apoyo para mejorar los servicios y la información ofrecida a los ciudadanos, las empresas y al propio Gobierno, lo que implica alcanzar mayores niveles de eficacia y eficiencia en el quehacer gubernamental, mejorando los procedimientos del gobierno , aumentando la calidad de los servicios públicos y facilitando la coordinación entre las diferentes instancias de gobierno.

La gestión electrónica de documentos con valor legal es un proceso global, corporativo e integral del proceso documental de una organización siendo un requisito indispensable dentro del Gobierno Electrónico, ya sean documentos generados por sistemas informáticos, hasta ejemplares impresos. El traslado de documentos impresos a documentos digitales es un proceso muy complejo que requiere de un tratamiento especial para que conserven su valor legal, esto trae consigo que se necesite más grado de complejidad a la hora de realizar el proceso que solo utilizar el scanner como único dispositivo de digitalización de documentos.

El Centro de Gobierno Electrónico (CEGEL), perteneciente a la Universidad de las Ciencias Informáticas (UCI) ha desarrollado, en el marco del Convenio Intergubernamental Cuba-Venezuela, sistemas para la obtención de objetos digitales con valor legal como: DigiPRO, DigiPyrus, DigiDAP, CDA entre otros, obteniendo resultados satisfactorios. Aunque cuentan con una función principal de gestionar solo la información para las entidades que fueron creados, todos estos sistemas contienen implementados, requisitos imprescindibles para lograr obtener documentos electrónicos con información legal.

El proyecto investigativo “Sistema para la obtención de documentos digitales con valor legal “correspondiente al centro CEGEL, tiene como objetivo desarrollar una solución tecnológica que permita agilizar la construcción de sistemas para la generación de

objetos digitales con valor legal con la automatización de los procesos de un Centro de Digitalización, propiciando la disminución del tiempo de desarrollo y comercialización de los mismos, la modernización e informatización de la gestión de objetos digitales con valor legal, así como la preservación y protección de fondos documentales. La solución está dividida en módulos, los cuales se especializan en los procesos de cada área del Centro de Digitalización.

Actualmente la gestión de un fondo documental para la obtención de objetos digitales con valor legal, posee un conjunto de deficiencias que fundamentan la problemática como son:

- Por lo general son entregados por el cliente, un gran volumen de documentos sin organización o clasificación previa.
- Falta de un mecanismo óptimo informatizado que posibilite la entrada y gestión de varios fondos documentales.
- Los documentos digitalizados, no poseen una ubicación asignada con respecto a los documentos físicos, lo que conlleva que: se tiene que invertir tiempo y esfuerzo en buscar dentro de todos los documentos físicos al que pertenece el documento digitalizado.
- Variedad en la plataforma tecnológica de desarrollo, en la mayoría de los casos utilizando tecnologías privativas que pueden ser remplazadas por tecnologías libres.
- Los softwares existentes que posibilitan la gestión de un fondo documental para su digitalización tienen escasas posibilidades de reutilización.

La solución de estos elementos garantiza el correcto funcionamiento del sistema, por lo que es necesario organizar y procesar los documentos, permitiendo así que los módulos puedan interactuar entre sí y se comuniquen los componentes de la solución, de manera automática.

Por lo antes planteado se hace necesario el desarrollo del proceso de preparación y digitalización de los fondos documentales para el sistema DIGILEX por lo que queda formulado como **problemática a resolver.**

¿Cómo facilitar la gestión de los fondos documentales para el proceso de digitalización del sistema DIGILEX?

Por tanto, el **Objeto de estudio** de este trabajo se encuentra enfocado al proceso de gestión de digitalización de los fondos documentales y el **Campo de acción** se encuentra enmarcado en el proceso de almacenar y preparar un fondo documental para la obtención de objetos digitales con valor legal.

**El Objetivo general** es desarrollar un módulo de gestión de fondos documentales del sistema DIGILEX, que permita el proceso de digitalización.

De este último se derivan **Objetivos específicos** los cuales están dados por:

- Elaborar el marco teórico de la investigación para fundamentar las bases de la solución.
- Desarrollar el módulo de gestión de un fondo documental para el sistema DIGILEX.
- Validar la solución propuesta aplicando diferentes métodos y métricas de validación de software.

Para dar cumplimiento a los objetivos planteados se desarrollaron las siguientes **tareas de investigación.**

- Realización de un estudio del estado del arte de los diferentes conceptos relacionados con el tema, así como de los sistemas existentes para la digitalización de documentos.
- Estudio de la metodología, lenguajes y herramientas para darle solución al problema planteado.
- Análisis y modelado de los procesos del negocio.
- Levantamiento de las funcionalidades de los procesos de negocio.
- Estudiar la Arquitectura y los patrones a utilizar.
- Diseño de la estructura y comportamiento de los componentes de los procesos identificados.
- Diseño del modelo de datos.
- Diseño del diagrama de componente del sistema.
- Validación del diseño propuesto.
- Implementación de los componentes diseñados.
- Validación de la implementación realizada.
- Validación de las variables de la investigación.

Señalando como **idea a defender**:

Con el desarrollo del módulo de gestión de los fondos documentales del sistema DIGILEX se contribuirá a facilitar la gestión de los mismos para el proceso de digitalización de dicho sistema.

**Métodos de investigación:**

**Métodos Teóricos:**

- **Histórico – Lógico:** permitió la realización del estudio de la evolución de los sistemas informáticos, para la gestión de documentos que se enmarcan en la digitalización de documentos. Realizando énfasis en las características más importantes que han surgido y que son aplicables en el diseño de la solución propuesta.
- **Análisis- Síntesis:** este método viabilizó la realización del estudio teórico de la investigación y el análisis de la gestión de un fondo documental para el proceso de digitalización, permitiendo que se sintetizaran los elementos más importantes de estos para el análisis de los documentos y la bibliografía con el objetivo de obtener información sintetizada y detallada relacionada con el objeto de estudio.

**Método Empírico:**

- **Entrevista:** este método se utilizó para esclarecer diversos puntos de vista sobre el tema de la investigación y recopilar datos concretos del cliente, con la finalidad de adquirir información necesaria que posibilitó el avance eficaz de la investigación y el logro de los objetivos propuestos. Se materializó la entrevista informal, individual y no estructurada.

**Estructura capitular:**

**Capítulo 1 Fundamentos Teóricos:** en este capítulo se abordan elementos importantes sobre las aplicaciones para la digitalización de documentos, se presentan las definiciones de digitalización de documentos, así como algunos elementos directamente relacionados con esta, fondos documentales y tipos documentales. Se realiza un estudio crítico sobre las herramientas existentes destinadas a la digitalización de documentos a nivel nacional e internacional. Además, se describe la metodología, las tecnologías, las herramientas y el lenguaje a utilizar para la construcción del módulo.

**Capítulo 2 Descripción y diseño del módulo:** en este capítulo se exponen los principales artefactos que se generan en la actividad de modelado realizada al módulo a desarrollar, como son: la descripción de los requisitos funcionales y no funcionales, definir la arquitectura que va a sustentar al sistema, el diseño del diagrama del despliegue del sistema, y el modelo de datos con el objetivo de tener una mejor comprensión, documentación del sistema y establecer las bases para la implementación del módulo.

**Capítulo 3 Validación del módulo:** Se describen las diferentes pruebas realizadas al sistema con el objetivo de verificar que el mismo cumple con los requerimientos planteados.

# Capítulo 1: Fundamentación teórica

## 1.1 Introducción

El total conocimiento sobre algunos conceptos que se correspondan a los diferentes procesos documentales y de digitalización de documentos, es de vital importancia para lograr una mayor visión sobre el tema de la investigación tratada. También es necesario representar varias herramientas en el ámbito nacional e internacional, sus principales características y cuáles de estas son de utilidad para tratar el problema a resolver. Por lo antes planteado a continuación se quedarán reflejada unas series de conceptos y se registran las herramientas y metodologías que dan un paso a la solución de los objetivos de investigación propuestos.

## 1.2 Definición de términos

Se tratarán diferentes conceptos teóricos que enriquecerán la investigación, facilitando una mejor obtención de información relacionado a la elaboración del módulo de preparación para el Sistema para la obtención de documentos digitales con valor legal.

### 1.2.1 Documento

Escrito en papel u otro tipo de soporte con que se prueba o acredita una cosa, como un título, una profesión, un contrato (Julián Pérez Porto, 2009).

Muchas son las clasificaciones que se realizan de los documentos, no obstante, una de las más frecuentes es aquella que tiene como criterio fundamental para desarrollarse el soporte en el que se encuentran los mismos. De ahí que básicamente se establezcan dos grandes grupos: documentales textuales, que son los que se realizan en papel, y documentos no textuales, que son aquellos que utilizan cualquier otro tipo de soporte para guardar una información concreta (Julián Pérez Porto, 2009).

#### 1.2.1.1 Documento digital

Cualquier conjunto de información que conforme una unidad significativa independiente, registrada en un soporte electrónico, constituye también un documento. Cualquier unidad significativa independiente de información registrada en un diskette, en un CD Rom o en un disco duro, es un documento. Sólo que en este caso no es ya un documento impreso o un documento audiovisual sino un “Documento Digital“ (Julián Pérez Porto, 2009).

### 1.2.2 Fondo documental

Los fondos constituyen la mayor agrupación documental, existente en un archivo y corresponden al conjunto de documentos, de cualquier formato o soporte, producidos orgánicamente y/o reunidos y utilizados por una persona particular, familia u organismo en el ejercicio de sus actividades (DIBAM, 2017; El pensante ©, 2017).

### 1.2.3 Gestión documental

La gestión documental se extiende al ciclo de vida completo de los documentos desde su producción hasta la eliminación final y su envío al archivo para su conservación permanente. Está dirigido a asegurar una documentación adecuada, evitar lo no esencial, simplificar los sistemas de creación y uso del papeleo. Además, mejora la forma de cómo se organizan y se recuperan los documentos, proporciona el cuidado adecuado y almacenamiento de los documentos en los centros de archivo y asegurar la ordenación adecuada de los documentos que no se necesitan por mucho tiempo en la conducción de los asuntos del momento. (Fernández Valdés, 2008).

Se entiende por consiguiente que la gestión documental permite administrar el flujo de documentos de todo tipo en una organización. Además, permite la recuperación de información desde ellos, determinando el tiempo que los documentos deben guardarse, eliminar los que ya no sirven y asegurar la conservación de los documentos más valiosos (Fernández Valdés, 2008).

### 1.2.4 Tipología documental

Por su parte, dentro de las Ciencias de la Información, se conoce como Tipología Documental a los distintos formatos y tipos de documentos que una entidad específica elige y desarrolla, para ser aplicados en las tareas de producción o recepción documental, a fin de generar a su vez una dinámica homogénea para todos los departamentos y fuentes de alimentación de archivos. En consecuencia, la Tipología Documental puede ser entendida como un producto directo de la Gestión de Archivos, pues, aunque tiende a ser universal, cada Empresa o entidad es la responsable de elegir sus propios formatos, según sus necesidades (El pensante ©, 2017).

Estas unidades documentales están clasificadas como simples o complejas:

- **Simples**: están formadas por un solo tipo documental, cuyo contenido mantiene una unidad de información. Ejemplos: el oficio, la carta, el memorando, un libro de registro, un libro de caja, recibo, cualquier otra forma de documento.
- **Complejas**: se conforma por dos o más tipos documentales que se sustentan entre sí y cuyo contenido mantiene una unidad de información. Se le conoce

comúnmente como “expediente”. Ejemplos: el expediente de un estudiante, trámites para licencias, cualquier otro trámite administrativo.

Así mismo, resulta pertinente revisar las distintas tipologías documentales que pueden encontrarse de forma general en las diferentes entidades administrativas, para las cuales suelen ser concebidos como tipos de documentos, aquellos pertenecientes a la siguiente clasificación:

- **De acuerdo al contenido:** En este primer ítem, contemplado también como el primer criterio de clasificación en lo que a tipología documental se refiere, se encuentran los documentos que son discriminados según el contenido que presentan, entre los que se encuentran los Documentos Imperativos (Leyes, Decretos, Sentencias, entre otros documentos producidos por las entidades que representan algún tipo de autoridad civil, eclesiástica o económica) así también como los Documentos Informativos; los Documentos Sustantivos y Documentos Facilitativos (El pensante ©, 2017).
- **De acuerdo a la tradición documental**: existe también un criterio de clasificación de tipología documental, el cual se basa en la tradición del documento, es decir, a qué momento de la evolución documental pertenece. En este sentido, los distintos tipos de documentos pueden diferenciarse según si se trata de un documento Original, una Copia, un Borrador, o incluso una Minuta (El pensante ©, 2017).
- **De acuerdo al nivel de accesibilidad**: igualmente, la capacidad o forma en que una persona accede al documento marca también una clasificación dentro de los distintos tipos de documentos. De esta manera, según este criterio, los documentos pueden considerarse como Documentos que requieren ser publicados; de libre acceso; confidenciales; no confidenciales; público o privado (El pensante ©, 2017).
- **De acuerdo a su clase**: finalmente, la Archivística fija también los distintos soportes materiales como criterio de clasificación de la Tipología Documental. En consecuencia, se puede decir que los documentos se distinguen según su clase en Documentos textuales, no textuales, iconográficos, sonoros, audiovisuales, magnéticos, materiales y digitales (El pensante ©, 2017).

**Importancia de la tipología documental**

En la gestión documental el trabajo con los tipos de documentos es sumamente importante debido a que se pueden clasificar los documentos para saber cómo serán tratados y se identifica el tipo de documento que será archivado. Esto proporcionará la eficiencia de la recuperación de la información para la toma de decisiones, así como la

protección de la misma, la estabilidad y continuidad administrativa, ya que se lleva a cabo un buen control sobre estos tipos. Conocer los tipos de documentos permite, además ordenar y organizar toda la documentación producida en una empresa. Esto favorecerá a un mayor desenvolvimiento en la organización permitiendo que la documentación con la que se trabaja sea más entendida por las personas que hacen uso de ella (El pensante ©, 2017).

### 1.2.5 Digitalización de documentos

La digitalización de documentos se define como la captura de una imagen física, mediante escáner o cámara digital, que una vez convertida en imagen electrónica puede ser almacenada y procesada por una computadora. Sus resultados están determinados por la resolución (densidad de puntos, o píxeles que tiene una imagen) y por la distribución luminosa en el documento, ya sea en sus niveles de grises o tonalidades de color. Por tanto, se entiende que la digitalización de documentos es el proceso de convertir un documento a una imagen que pueda ser reconocida por un computador (LinkedIn Corporation © , 2017).

### 1.2.6 Documento con valor legal

Un documento de archivo puede tener diferentes valores. El valor legal es el que tienen todos los documentos que sirven de testimonio ante la ley. Sirve para documentar las obligaciones legales y proteger los derechos de las personas y las instituciones. Algunos de los documentos con valor legal son: un testamento, una tesis doctoral y el registro de miembros de una asociación. Por consiguiente, se entiende por documento con valor legal como un documento que sirve para darle valor jurídico a un hecho, ya sea un acta, carta o escrito (El pensante ©, 2017).

## 1.3 Sistemas existentes

En la actualidad existen herramientas que resuelven el gran problema de digitalizar los documentos, los cuales están enfocados mayormente a las necesidades de empresas e instituciones que trabajan con gran cantidad de papeles. Muchas no incluyen las funcionalidades relacionadas con digitalizar documentos con valor legal y gestionar los tipos documentales. A continuación, se expone el análisis de los sistemas homólogos.

### 1.3.1 En el Mundo

**Aspen**

Aspen es un completo sistema de gestión documental con funcionalidades avanzadas que permite el almacenamiento de los documentos del histórico de una empresa y su consulta vía web. DocPath ha convertido a Aspen en un software de gestión documental

de gran envergadura, al incorporar todo un conjunto de funcionalidades avanzadas que van mucho más allá del mero almacenamiento. De este modo, la solución proporciona todas las prestaciones que necesitan los usuarios para una óptima gestión de sus documentos (DocPath Corp, 2017).

- **Características principales**

A las opciones de almacenamiento y recuperación, Aspen añade una serie de funcionalidades avanzadas que hacen de este software una potente solución de gestión documental (DocPath Corp, 2017).

1. **Generación de documentos en tiempo real** – DocPath Aspen permite almacenar los datos de negocio, los formularios y sus recursos asociados en las diferentes versiones que utiliza su organización, de forma totalmente independiente, para su recuperación posterior en tiempo real. los almacena y los indexa en una base de datos.
2. **Método de distribución controlada –** La solución de gestión documental ofrece un método de distribución controlada de la documentación, puesto que permite tener bajo control toda la información que está en manos de terceros y conocer el uso que se hace de ella.
3. **Sistema de digitalización de documentos impresos –** El software de gestión documental de DocPath integra, asimismo, capacidades de captura para la digitalización masiva de documentos. Mediante esta funcionalidad, Aspen escanea los documentos firmados por el cliente para pasarlos a formato digital, los almacena y los indexa en una base de datos.

- **Ventajas**

1. **Generación eficiente en tiempo real** del documento solicitado, a partir de los datos de negocio y los formularios almacenados en las diferentes versiones que utiliza su organización.
2. **Mejora el sistema de distribución** de la documentación, al permitir el acceso inmediato de los usuarios a través de la web, en un entorno totalmente controlado y con trazabilidad de acciones realizadas.
3. **Sistema ágil de búsqueda** basado en índices personalizados.
4. **Posibilita restringir el uso** sobre determinados documentos, con parámetros de seguridad a la hora de acceder, modificar, guardar o imprimir.

5. **Sistema de digitalización** para el reconocimiento de documentos impresos mediante la lectura de un código de barras e indexación automática en el gestor documental.
6. Permite realizar **diferentes acciones sobre los documentos**, como consultar, imprimir o enviar por email.
7. **Fácilmente integrable** con el resto de aplicaciones presentes en su organización.
8. **Ahorros significativos** de espacio de almacenamiento, de tiempo a la hora de realizar búsquedas, y de costos asociados a la impresión y distribución.

**DDS (Document Digitization System)**

DDS es una herramienta útil para todo tipo de negocio que tenga que almacenar gran cantidad de documentos; ya que le brinda una forma fácil y sencilla de digitalizar y almacenar su información. Este programa le ahorra tiempo y espacio a la hora de buscar un documento ya que además de ser un programa para uso de digitalización también le sirve para la búsqueda de documento digitalizados ya que cuenta con un motor de búsqueda “Search Engine” para adquirir su información en solo un paso. DDS es un programa “Open Source” (Código fuente, abierta) que se puede modificar de acuerdo a sus necesidades. Además, cuenta con un sistema de seguridad independiente donde usted asigna cada empleado a distintos permisos. DDS es compatible con cualquier formato de digitalización y para cualquier tipo de scanner de documentos.

DDS almacena una referencia principal para que sea fácil clasificar los documentos como, por ejemplo: Número de Cliente, Número de Record, Nombre de Cliente, etc.; esta información puede ser modificada de acuerdo al tipo de negocio que lo valla a emplear (General technology Corporation, 2017).

- **Características principales**

**Contiene un campo donde el usuario** escribirá el nombre que asignará al documento procesado.

**Contiene un campo donde el usuario** seleccionará la carpeta donde quiere almacenar el documento**.**

**EasyScan**

EasyScan es una utilidad que puedes usar con cualquier escáner, para digitalizar todo tipo de documentos en papel. Su objetivo es crear un estándar en lo que se refiere a

programas de escaneado, dado que cada uno de ellos incorpora utilidades distintas y no siempre fáciles de usar.

El programa incorpora una interfaz con diversas opciones de configuración: brillo, contraste, resolución, zoom, etc. También puede guardar varias hojas para imprimirlas después todas al mismo tiempo.

EasyScan tiene además soporte para varios idiomas instalando diferentes paquetes de lenguaje, entre ellos, el de español (SOFTONIC INTERNACIONAL S.A, s.f.).

### 1.3.2 En Cuba

En Cuba el Consejo de Estado a partir de agosto de 2001 optó por decretar leyes que establecen las normas y principios que rigen la actividad archivística en el territorio nacional. Comenzaron entonces a crearse y extenderse centros u organizaciones con el fin de dar soluciones informáticas a la gestión documental, sobresaliendo entre ellos el Centro de Gobierno Electrónico con reconocidos resultados alcanzados en el desarrollo de sistemas para la digitalización de fondos documentales con valor legal.

**DigiPyrus**

El sistema de Digitalización DigiPyrus, desarrollado por el Centro de Gobierno Electrónico (CEGEL) de la Universidad de las Ciencias Informáticas (UCI), permite digitalizar los Tomos de los Registros y Notarías de la República Bolivariana de Venezuela, contribuyendo a agilizar el acceso a la información que a diario se tramita en cada una de sus oficinas. Este sistema surge debido a la demanda de una gestión más eficiente y segura de los trámites que se realizan con los documentos archivados en las Oficinas de los Registros y Notarías Públicas. De ahí, que se haya propuesto presentar una solución que cumpla con el objetivo de dotar a esta entidad de un sistema que contenga las funciones, para realizar el proceso de digitalización de todos los documentos archivados y de esta forma obtener el fondo digital de cada una de estas (La Rosa Montes.Dayana R. A., 2007).

**Procesos**

Los procesos que se informatizan (La Rosa Montes.Dayana R. A., 2007) mediante este sistema son:

- **Recepción de Documentos:** Verificación y registro de los tomos provenientes del Almacén Temporal
- **Preparación de documentos:** Ejecución de las operaciones establecidas para garantizar que los tomos cumplan con los parámetros necesarios para el proceso de digitalización.

- **Escaneo de Documentos:** Conversión de los tomos físicos en objetos digitales multipáginas.
- **Control de la Calidad:** Inspeccionar el 100% del objeto digital multipáginas proveniente del área de Digitalización de Documentos.
- **Encuadernación de Documentos**: Recuperación y preservación de las condiciones iniciales de los tomos digitalizados.
- **Devolución de Documentos:** Registro de salida de los tomos que han sido digitalizados en el proceso.
- **Asociación de Metadatos:** Identificación de los datos de los objetos digitales multipáginas para la asociación de sus metadatos.
- **Exportación:** Exportación de las imágenes digitales multipáginas de los documentos del tomo al Centro de Datos del SAREN.

**DigiDaP**

DigiDAP forma parte de la Solución Tecnológica Integral para la automatización y modernización de la División de Antecedentes Penales de la República Bolivariana de Venezuela, como el subsistema que informatiza los procesos del Centro de Digitalización para el fondo documental de dicha institución. Su objetivo fundamental es garantizar la obtención de objetos digitales con valor legal a partir de la digitalización del fondo para mejorar los servicios de inscripción y certificación de antecedentes penales a los ciudadanos de la nación venezolana (González Valdés, 2011).

**Procesos**

En el modelo de referencia para la digitalización de fondos documentales de antecedentes penales (González Valdés, 2011) (Borges Piedra, 2011).

- **Recepción y Devolución de Documentos:** Recepcionar expedientes al Centro de digitalización, devolver desde este al archivo y eliminar expedientes que por errores se decidan quitar del proceso.
- **Preparación de Documentos:** Preparar los expedientes y garantizar que tengan las condiciones necesarias para pasar por el escáner en la próxima área, evaluar la calidad de las unidades documentales para determinar si proceden o no en el proceso y registrar los folios a digitalizar de cada una y emitir notas de preparación en caso de que sea necesario para esclarecer la omisión de folios a digitalizar.
- **Digitalización de Documentos:** Digitalizar las unidades documentales contenidas en los expedientes, rectificar errores en los trámites que hayan sido regresados en el proceso y mejorar la calidad visual de los documentos digitales
- **Asociación de Metadatos:** Asociar metadatos a las unidades documentales contenidas en los expedientes, identificar posibles coincidencias documentales

para eliminar unidades documentales duplicadas en el archivo, identificar posibles asociaciones entre unidades documentales en expedientes y rectificar errores de metadatos en los trámites que hayan sido regresados en el proceso.

- **Control de Calidad:** Revisar legalmente los trámites realizados en el Centro de Digitalización, emitir la respuesta correspondiente del trámite e imprimir salidas del sistema.
- **Otorgamiento:** Otorgar trámites revisados legalmente, certificando la calidad del proceso realizado, así como otorgar firma manuscrita a las salidas impresas del sistema y la firma digital a los documentos digitales, obtener reportes estadísticos sobre el proceso y localizar trámites dentro del proceso y consultar el estado de los mismos.
- **Encuadernación de Documentos**: Encuadernar y foliar las unidades documentales dentro de los expedientes para su asentamiento en archivo.

**CDA**

CDA es la solución tecnológica desarrollada para el Centro de Digitalización de Alfabéticas para el Servicio Administrativo de Identificación, Migración y Extranjería. Para su desarrollo se utilizó como marco de trabajo el utilizado por la solución DigiPyrus por lo que las características arquitectónicas y tecnologías de desarrollo utilizadas son similares. Su objetivo fundamental es extraer tarjetas alfabéticas para convertirlas en metadatos. Esta solución está compuesta por varios módulos, algunos de estos son (Sistema de identificación, migración y extranjería., 2011):

- **Almacén Temporal:** permite la gestión de inventario de los recursos que entran y salen del Centro de Digitalización, así como del control de recibo y despacho de las Tarjetas de Alfabética desde las Oficinas de Identificación hacia cada Centro de Digitalización.
- **Preparación**: permite la creación de los lotes de documentos en el sistema, se registran los documentos rechazados por malas condiciones para entrar en el proceso de digitalización y se realiza la corrección de los documentos rechazados por mala preparación o códigos de barra ilegible.
- **Digitalización**: permite la digitalización de los lotes de documentos o de documentos rechazados.
- **Firma digital:** permite la emisión de la firma digital de los documentos, certificando que la información obtenida es verídica.

**Tabla 1: Comparación de los sistemas estudiados.**

| Solución | Licencia | Gestión de un fondo documental para su digitalización. | Digitalizan un solo tipo documental |
|---|---|---|---|
| **Aspen** | Si | Si | Si |
| **DDS** | Si | Si | Si |
| **EasyScan** | Si | No | Si |
| **DigiPyrus** | Sí | Sí | Si |
| **DigiDAP** | Sí | Sí | Si |
| **CDA** | Sí | Sí | Si |

### 1.3.3 Valoración Crítica.

Sobre el análisis de las aplicaciones para la digitalización de documentos existentes a nivel internacional se tiene, que no resultan soluciones factibles a tener en cuenta, pues estos fueron desarrollados con software propietario, lo que implica gastos muy elevados en licencias y mantenimiento. El Aspen identifica los documentos con códigos de barra único y el DDS le da la posibilidad al usuario de identificarlos con un nombre que él introduzca en el sistema. Sobre el estudio a nivel nacional, al igual que los sistemas internacionales, estos fueron desarrollados con software propietarios. Pero no se pueden descartar estas aplicaciones porque DigiPyrus, DigiDAP y CDA realizan funciones cercanas a nuestro problema a resolver, ya que tienen sus módulos de almacén y preparación de documentos, pero son sistemas hechos a la medida, o sea, destinados a digitalizar un solo tipo documental. Además, CDA gestiona los documentos y los agrupa en lotes, lo que sería una solución factible para nuestro módulo. Con lo que se puede concluir que las funcionalidades de CDA como crear lotes, asociar documentos al lote e identificar los documentos físicos y digital con un código de barra que sea identificador único en nuestra factoría de digitalización, son una solución viable para el desarrollo de nuestro problema a resolver.

## 1.3 Preparación de documentos

### 1.4.1 Preparación del material para digitalizar.

Los recursos necesarios para llevar a cabo esta tarea son el personal encargado del proceso de digitalización, utensilios para manipular de forma correcta el documento (guantes, saca presillas, pinzas, etc.), un computador y el sistema DIGILEX.

Para cumplir con los objetivos de calidad se requiere preservar la integridad física del documento original, proporcionando el material en las condiciones necesarias para una eficiente manipulación y por último controlar la salida y entrada de documentos a las áreas.

El principal resultado esperado de esta área es lograr que el material quede correctamente registrado en el sistema DIGILEX y en condiciones aptas para ser digitalizado.

Se han englobado en 7 pasos generales las tareas a llevar a cabo en el proceso de preparación de documentos. Estas son:

- Recibir envío del fondo documental.
- Diagnóstico de los documentos para su digitalización.
- Separar y organizar los documentos para el proceso de digitalización.
- Clasificar el tipo documental de los documentos que pueden pasar al proceso de digitalización.
- Ingresar al sistema DIGILEX los documentos con un identificador único.
- Entregar los documentos al departamento de digitalización.
- Recibir los documentos una vez terminado el proceso de digitalización para prepararlos y entregarlos al cliente.

**Tarea 1: Recibir envío del fondo documental.**

Al recibir el fondo documental se comprueba que la cantidad de documentos entregados sea la acordada en el contrato, luego se deben ubicar en un Almacén, con seguridad, con el objetivo de mantener la integridad de los documentos.

**Tarea 2: Diagnóstico de los documentos para su digitalización.**

Los documentos deben revisarse para asegurar que estos presentan condiciones aptas para ser digitalizados. Esta actividad es muy importante cuando los documentos tienen un nivel de antigüedad considerable por lo que están dañados, o cuando están unidos en forma de libros y necesitan ser separados. Se utilizan varias técnicas para asegurar la calidad del proceso de preparación como el planchado, la limpieza, el desengrapado, plasticar los documentos originales cuyas hojas sean difíciles de manipular para evitar su maltrato en el escáner, siempre que las condiciones tecnológicas lo permitan, entre otros. Se puede realizar antes de pactar el contrato con el cliente.

**Tarea 3: Separar y organizar los documentos para su digitalización.**

Corresponde al funcionario decidir si los documentos originales pasan directamente a los digitalizadores o deberán ser previamente fotocopiados. Tal decisión debe estar

basada en aspectos como: tipo de escáner (plano o con alimentador) con el que se cuenta, ritmo de producción con el cual se quiera trabajar (volumen alto o bajo de producción), preservación del buen estado físico de los documentos y calidad de la imagen.

Debe controlarse la entrada de documentos originales al proceso de digitalización con el objetivo de evitar pérdidas de material por manipulación durante el proceso.

Se debe hacer posible la rápida localización de los documentos físicos: cada expediente o documento debe tener asignada una ubicación y no otra, de forma que su búsqueda sea ágil y siempre encuentre la respuesta justa.

El arreglo u organización de un archivo engloba una serie de tareas que, aunque se hallan estrechamente relacionadas, conviene distinguir:

La elaboración de un cuadro de clasificación que dé cabida a todos los tipos documentales y refleje la estructura dada al archivo. Clasificar los fondos consiste en definir y concretar los grandes grupos que forman el archivo (documental, 2012):

**a)** Las secciones, división que se establece a partir de las principales actividades desarrolladas por la asesoría (contable, fiscal, laboral, mercantil, auditoría, etc.).

**b)** Las series documentales, conjuntos de documentos producidos como resultado de un mismo procedimiento administrativo o jurídico.

Así, en una asesoría, la documentación de los clientes se agrupará en primer lugar por áreas de negocio y después por tipos de servicio.

La ordenación de los documentos dentro cada serie documental empleando el criterio más adecuado para cada tipo de documento. Esta operación, en tanto complemento de la clasificación, consiste en poner en orden (cronológico, alfabético, numérico…) las unidades documentales (documentos o expedientes) dentro de las series constituidas.

La instalación física de los documentos en el depósito de archivo, mediante cajas archivadoras u otro sistema, que va a posibilitar la localización de los documentos a través del correspondiente código de archivo o signatura topográfica.

**Tarea 4: Clasificar el tipo documental de los documentos que pueden pasar al proceso de digitalización.**

Debe clasificarse el tipo documental de los documentos que van a pasar al proceso de digitalización facilitando la captura de Metadatos.

**Tarea 5: Ingresar al sistema DIGILEX los documentos con un identificador único.**

Se recomienda introducir los documentos en el sistema, a través de un identificador único, con el objetivo de asociar el documento físico con el objeto digital.

**Tarea 6: Entregar los documentos al departamento de digitalización**

Los documentos preparados (clasificados y con sus códigos de barra) y listos para comenzar a ser digitalizados pasan al área de digitalización

**Tarea 7: Recibir los documentos una vez terminado el proceso de digitalización para prepararlos y entregarlos al cliente.**

Los documentos cuando terminan el proceso completo de digitalización son devueltos al área de preparación, donde se organizan para ser entregados al cliente de la misma manera en que llegaron al centro de digitalización.

## 1.5 Herramientas y tecnologías utilizadas

### 1.5.1 Metodología de desarrollo de software

Una metodología de desarrollo de software no es más que un conjunto de procedimientos, técnicas, herramientas y un soporte documental para los desarrolladores de un sistema. La selección de una adecuada metodología de desarrollo de software, es una muy importante decisión ya que todo desarrollo de sistemas se torna difícil y riesgoso si no se lleva un control mediante una está, así mediante un buen resultado mantener satisfechos a los clientes o usuarios. Según lo anteriormente planteado, una metodología es un conjunto de componentes que especifican (Varela):

- Cómo dividir un proyecto en etapas.
- Qué tareas se llevarán a cabo en cada etapa.
- Qué salidas se producen y cuándo deben producirse.
- Qué restricciones se aplican.
- Qué herramientas van a hacer utilizadas.
- Cómo se gestiona y controla el proyecto

Existen diferentes tipos de metodologías de desarrollo de software los cuales pueden ser clasificados en tradicionales o ágiles.

**Metodologías Tradicionales**

Las metodologías tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo del software, con el objetivo de conseguir un software más eficiente y predecible. Para ello, se realiza primeramente la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto. Este planteamiento está basado en el resto de disciplinas de ingeniería, a pesar de que

el software no pueda considerarse como la construcción de una obra clásica de ingeniería (Figueroa, Solís, & Cabrera, 2008).

**Metodologías Ágiles**

Las metodologías ágiles aportan nuevas técnicas y métodos de trabajo para el desarrollo de cada etapa de un software. En general esta metodología hace un balance entre los procesos y el esfuerzo, ya que tratan de centrarse en las cuestiones necesarias sin perderse en las burocráticas y se basa principalmente en satisfacer al cliente mediante tempranas y continuas entregas de software donde este y los desarrolladores del producto se encuentran en constante comunicación ya que el primero forma parte del equipo de desarrollo (Canós, 2003).

Para orientar el desarrollo del módulo, se decidió emplear una metodología ágil debido a que al equipo de desarrollo se encuentra caracterizado por:

- Presentar un equipo de desarrollo muy pequeño y por ende con pocos roles.
- Énfasis en la entrega frecuente de funcionalidades desarrolladas.
- Optimizar el tiempo y evitar la demora en la generación de artefactos.
- Propiciar el trabajo conjunto del cliente con el equipo de desarrollo.

**AUP – UCI**

La metodología a emplear es una variación de la metodología AUP (Proceso Unificado Ágil), en unión del modelo CMMI-DEV v1.3 (Sánchez, 2015), ya que está definida por el proyecto que desarrolla el módulo. Además, genera los artefactos que permiten obtener la documentación necesaria para una mejor comprensión de la herramienta a desarrollar.

Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan un negocio muy bien definido. El cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos. Se recomienda en proyectos no muy extensos, ya que una HU no debe poseer demasiada información. Todas las disciplinas antes definidas (desde Modelado de negocio hasta Pruebas de Aceptación) se desarrollan en la Fase de Ejecución, de ahí que en la misma se realicen Iteraciones y se obtengan resultados incrementales, ver Figura 1. En una iteración se repite el flujo de trabajo de las disciplinas, Requisitos, Análisis y diseño, Implementación y Pruebas internas. De esta forma se brinda un resultado más completo para un producto final de manera creciente. Para llegar a lograr esto, cada requisito debe tener un completo desarrollo en una única iteración (Sánchez, 2015).

Esta metodología cuenta de tres fases: inicio, ejecución y cierre por las que deben transitar durante el desarrollo de las actividades productivas. La presente investigación elige la fase de ejecución, en la que se ejecutan las actividades requeridas para desarrollar el software, incluyendo el ajuste de los planes del proyecto considerando los requisitos y la arquitectura.

Durante el desarrollo se obtienen los requisitos, se elabora la arquitectura y el diseño, se implementa y se libera el producto.

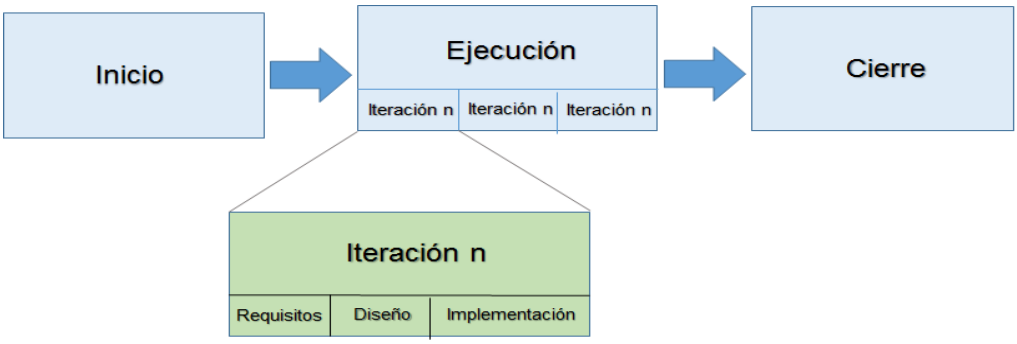



**Figura 1: Fases de AUP-UCI.**

De las siete disciplinas que propone la metodología se eligen cinco, las cuales son:


Para modelar el desarrollo de la solución, se opta por el cuarto escenario, el cual define que son para proyectos que no modelen negocio solo pueden modelar el sistema con HU (Historia de Usuarios).

### 1.5.2 El Lenguaje Unificado de Modelado UML

El Lenguaje Unificado de Modelado (por sus siglas en inglés UML) es el lenguaje de modelado más conocido y utilizado en la actualidad para el desarrollo de software. UML sirve para el modelado completo de sistemas complejos, tanto en el diseño de los sistemas de software como para la arquitectura de hardware donde se ejecuten. Permite visualizar, construir y documentar los elementos que forman un sistema de software orientado a objetos. Este lenguaje indica cómo escribir y leer los modelos, no dice como crearlos, pues eso es objetivo de la metodología que se utilice (Larman, UML y patrones, 1999). Se seleccionó UML en su versión 2.1 para modelar los artefactos seleccionados durante el proceso de desarrollo del software.

### 1.5.3 Herramienta de modelado

Las herramientas CASE comprenden un conjunto de programas de diferentes tipos empleados para ayudar a las actividades del proceso del software como el análisis de requisitos, el modelado de sistemas, la depuración y las pruebas (Sommerville, 2005).

#### 1.5.3.1 Visual Paradigma para UML

Como herramienta CASE se opta por el Visual Paradigm para UML en su versión 8.0, ya que es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

### 1.5.4 Lenguaje de Programación

Un lenguaje de programación es un lenguaje que puede ser utilizado para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente (Microsoft, 2005).

#### 1.5.4.1 Java

Java es un lenguaje de programación orientado a objetos ideado por Sun Microsystem[1]. Es un lenguaje de propósito general por lo que con él se podría crear cualquier tipo de aplicación. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria (Fernandez, 2005).

Como parte de las tendencias de lenguajes de programación actuales, Java presenta los siguientes beneficios:

- Es independiente de la plataforma de desarrollo.
- Existen dentro de su librería, clases gráficas que permiten crear objetos visuales comunes altamente configurables y con una arquitectura independiente de la

[1] Sun Microsystems fue una empresa informática que se dedicaba a vender estaciones de trabajo, servidores, componentes informáticos, software (sistemas operativos) y servicios informáticos.

plataforma. Java permite a los desarrolladores aprovechar la flexibilidad de la Programación Orientada a Objetos en el diseño de sus aplicaciones

- El manejo de las bases de datos es uniforme, es decir, transparente y simple.
- El sistema de Java es seguro ya que tiene muchas funciones de seguridad integradas, que garantizan la seguridad del código que se está ejecutando.

Para el trabajo de diploma se decide usar Java en su versión 1.8 para la implementación del módulo, ya que es un lenguaje orientado a objetos, distribuido, seguro, multitarea y de gran rendimiento.

### 1.5.5 Entorno de Desarrollo Integrado (IDE)

Un IDE es un entorno o marco de trabajo para la mayoría de los lenguajes de programación, contiene un editor de código, un compilador y un depurador, entre otros (Duarte, 2015).

Netbeans es un IDE disponible para Windows, Mac, Linux y Solaris. Permite el rápido y fácil desarrollo de aplicaciones Java de escritorio, móviles y aplicaciones web, además proporciona una gran herramienta para JavaFX, PHP, JavaScript y Ajax, Ruby y Ruby en Rails, Groovy y Grails y C/C ++ (Gimeno, 2011).

A continuación, algunas ventajas de este IDE (Gimeno, 2011):

- Es un IDE multilenguaje y adaptable.
- Es software libre y gratuito.
- Intuitivo y fácil de utilizar.
- Se puede desarrollar todo tipo de aplicaciones.
- Es poderoso y extensible.
- Fácil integración con el servidor de aplicaciones Glassfish.

Para desarrollar el módulo propuesto se escogió NetBeans en su versión 8.1 como entorno de desarrollo integrado debido a las características mencionadas anteriormente.

### 1.5.6 Sistema de Gestor de Base Datos SGBD – PostgreSQL

Un sistema gestor de base de datos (SGBD) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para añadir, borrar, modificar y analizar los datos. Los usuarios pueden acceder a la información usando herramientas específicas de interrogación y de generación de informes, o bien mediante aplicaciones al efecto. Estos sistemas también proporcionan métodos para mantener la integridad de los datos, para administrar el acceso de usuarios a los datos y para recuperar la información si el

sistema se corrompe. Permiten presentar la información de la base de datos en variados formatos. La mayoría incluyen un generador de informes. También pueden incluir un módulo gráfico que permita presentar la información con gráficos y tablas (Ramírez, 1999).

Como SGBD se opta por PostgreSQL 8.0, ya que es orientado a objetos de software libre, publicado bajo la licencia BSD2. Permite que mientras un proceso escriba en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último que se ejecutó. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases de datos, eliminando la necesidad del uso de bloqueos explícitos. Postgres ofrece una potencia adicional sustancial al incorporar conceptos adicionales básicos en una vía en la que los usuarios pueden extender fácilmente el sistema como clases, herencia, tipos y funciones.

Utiliza un modelo cliente-servidor y multiproceso en vez de multihilo para garantizar la estabilidad del sistema. La aplicación de escritorio pgAdmin III se utiliza como interfaz gráfica para hacer más fácil la administración del usuario con la base de datos, además se puede ejecutar en varias plataformas (Momjian, 2001).

### 1.5.7 Mapeo Objeto-Relacional

En la actualidad, casi todas las aplicaciones están diseñadas para usar la Programación Orientada a Objetos (POO), y las bases de datos más utilizadas son del tipo relacional, es decir, que solo permiten guardar tipos de datos primitivos. Este problema trae consigo que no se pueda guardar directamente en las tablas los objetos que genera la aplicación, por lo que se hace necesario realizar el Mapeo de Objeto-Relacional (ORM), ya que es el encargado de realizar conversiones y simula que la base de datos es orientada a objeto (Amador, 2013).

**ORM:** es una técnica de programación para convertir datos entre el lenguaje de programación orientado a objetos utilizado y el sistema de base de datos relacional utilizado en el desarrollo de una aplicación (Amador, 2013).

Para el mapeo de la base de datos se opta por el Eclipse Link en su versión 2.5.0 , ya que es un ORM de código abierto y el marco de persistencia de objetos está apoyado en un entorno de Java. Soporta varias fuentes y formatos de datos incluyendo las

2 Berkeley Software Distribution (Distribución de Software Berkeley)

bases de datos relacionales, no relacionales, los servicios XML (Lenguaje de Marcas Extensible), JSON[3] y se puede utilizar en cualquier plataforma Java.

### 1.5.8 XEGFORT

XEGFORT es un marco de trabajo que facilita la reutilización de componentes en sistemas de corte empresarial. Es fácil de utilizar y configurar. Algunos de los elementos fundamentales con los que cuenta este marco de trabajo son la seguridad, la configurabilidad y la adaptabilidad, convirtiéndolo en un software altamente reusable, además garantiza la confiabilidad e integridad de la información que se intercambia a través de la red entre los clientes y el servidor de aplicaciones, facilitando la autenticación de los clientes (Lenis Matos Rodriguez, 2012).

Entre las principales funcionalidades con las que cuenta XEGFORT es la fortaleza de las contraseñas de las cuentas de los usuarios, contando con una encriptación de las mismas, evitando el robo de estas, que implicaría comprometer la seguridad del sistema. También se lleva un control del tiempo de inactividad máximo permitido sin que un operador interactúe con la aplicación reduciendo el riesgo de accesos que no estén autorizados a interactuar con el sistema (Lenis Matos Rodriguez, 2012).

Entre otras funcionalidades con las que cuenta son: la gestión de las sesiones de los usuarios, la gestión de usuarios, el control de trazas y excepciones y otras funcionalidades de administración del sistema.

Está diseñado para la creación de aplicaciones con arquitectura Cliente-Servidor utilizando la plataforma JEE (Java Platform, Enterprise Edition). Por esta razón XEGFORT consta de dos partes, una que se encuentra en el lado del cliente facilitando la creación de la capa de presentación y la comunicación con los objetos remotos desplegados en el servidor y una segunda que se encuentra en el lado del servidor donde se encuentra la lógica de negocio (Lenis Matos Rodriguez, 2012).

Para el desarrollo del módulo se opta por el marco de trabajo XEGFORT en su versión 1.2.2, ya que es el marco de trabajo definido por el proyecto.

## 1.6 Arquitectura de software

La arquitectura de software se enfoca principalmente en la estructura general del software y las formas en que la estructura proporciona una integridad conceptual para un sistema. En su forma más simple, la arquitectura es la estructura u organización de los componentes del programa (módulos), la manera en que estos componentes

3 JSON (JavaScript Object Notation): formato de texto ligero para el intercambio de datos.

interactúan, y la estructura de datos que utilizan los componentes. En sentido más amplio es una representación que permite, analizar la efectividad del diseño para cumplir los requerimientos establecidos, considerar alternativas arquitectónicas en una etapa en la que hacer cambios al diseño todavía es relativamente fácil y reducir los riesgos asociados con la construcción del software (Pressman R. S., 1988).

A continuación, se fundamentarán los diferentes estilos de arquitectónicos propuestos en la solución.

### 1.6.1 Arquitectura Cliente-Servidor

Esta arquitectura consiste fundamentalmente en un cliente que realiza peticiones a otro programa (Servidor) el cual le da respuesta. La interacción cliente-servidor es el soporte de la mayor parte de la comunicación por redes. Ayuda a comprender las bases sobre las que están construidos los algoritmos distribuidos.

Según Somerville, cuando un sistema se organiza como un conjunto de servicios y servidores asociados, más clientes que acceden y usan los servicios, se está en presencia del modelo arquitectónico cliente-servidor, el cual está compuesto por los siguientes componentes:

- Un conjunto de servidores que ofrecen servicios a otros subsistemas.
- Un conjunto de clientes que llaman a los servicios ofrecidos por los servidores.
- Una red que permite a los clientes acceder a estos servicios. Esto no es estrictamente necesario, ya que los clientes y los servidores podrían ejecutarse sobre una misma máquina. (Sommerville, 2005).

### 1.6.2 Arquitectura en Capas

Esta arquitectura se basa fundamentalmente en una organización jerárquica donde cada capa proporciona servicios a la inmediatamente inferior y se sirve de las prestaciones que le brindan la inmediatamente superior, al dividir un sistema en capas, cada capa puede tratarse de forma independiente, sin tener que conocer los detalles de las demás. La división de un sistema en capas facilita el diseño modular, en la que cada capa encapsula un aspecto concreto del sistema y permite además la construcción de sistemas débilmente acoplados, lo que significa que, si se minimiza las dependencias entre capas, resulta más fácil sustituir la implementación de una capa sin afectar al resto del sistema. Los sistemas o arquitecturas en capas constituyen uno de los estilos que aparecen con mayor frecuencia mencionados como categorías mayores del catálogo, o por el contrario, como una de las posibles encarnaciones de algún estilo más envolvente (Lezcano., 2009).

Las principales ventajas del estilo de arquitectura basado en capas son (Microsoft | Achitecture, 2010):

- **Abstracción:** las capas permiten cambios que se realicen en un nivel abstracto.
- **Aislamiento:** la arquitectura de capas permite aislarlos cambios en tecnologías a ciertas capas para reducir el impacto en el sistema total.
- **Rendimiento:** distribuir las capas entre múltiples sistemas (físicos) puede incrementar la escalabilidad, la tolerancia a fallos y el rendimiento.
- **Mejoras en pruebas:** la capacidad de realizar pruebas se beneficia de tener interfaces bien definidas para cada capa, así como de la habilidad para cambiar a diferentes implementaciones de las interfaces de cada capa.

## 1.7 Patrones de diseño

Los patrones de diseño son herramientas que proveen facilidades para crear un software reutilizable de buena calidad. Cada patrón describe un problema que ocurre repetidamente en nuestro entorno, y describe el núcleo de la solución a ese problema, de tal forma que ésta pueda ser usada varias veces, sin hacer el mismo trabajo dos veces (ROJAS, 2010).

**Patrones DAO**

El patrón Objeto de Acceso a Datos (DAO por sus siglas en inglés) suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos. Consiste básicamente en una clase que es la que interactúa con la base de datos. En una aplicación, hay tantos DAOs como tablas en la base de datos. Los métodos de esta clase dependen de la aplicación y de lo que queramos hacer (ROJAS, 2010).

**Patrones DTO**

Los DTO (Data Transfer Object) o también denominados VO (Value Object) son utilizados por DAO para transportar los datos desde la base de datos hacia la capa de lógica de negocio y viceversa (Larman, UML y patrones, 1999).

**Patrones GRASP**

Los Patrones Generales de Software para la Asignación de Responsabilidades (GRASP por sus siglas en inglés) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

- **Experto**: El patrón experto en información se utiliza con frecuencia en la asignación de responsabilidades; es un principio de guía básico que se utiliza

continuamente en el diseño de objetos. Este permite conservar el encapsulamiento, ya que los objetos se valen de su propia información para llevar a cabo las tareas. El comportamiento se distribuye entre clases que cuentan con la información requerida, alentando con ello definiciones de clase sencillas y más cohesivas que son más fáciles de comprender y mantener. Así se brinda soporte a una alta cohesión.

- **Creador:** El patrón guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento lo cual supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización (Larman, UML y patrones, 1999).
- **Controlador:** El patrón controlador es un objeto que no pertenece a la interfaz de usuario, es un manejador artificial de todos los eventos del sistema que define el método de su operación. A menudo genera un bajo acoplamiento y permite un mayor potencial de los componentes reutilizables. En (Larman, UML y patrones, 1999) se plantea que los objetos externos de conexión y la capa de presentación no deberían tener la responsabilidad de llevar a cabo los eventos del sistema.
- **Bajo acoplamiento**: Consiste en tener las clases lo menos relacionadas entre sí, para que, en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión en el resto de las clases. Esta característica permite potenciar la reutilización y disminuye la dependencia entre las clases (Larman, UML y patrones, 1999).
- **Alta cohesión:** en la perspectiva del diseño orientado a objetos, la cohesión (o, más exactamente, la cohesión funcional) es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme (Larman, UML y patrones, 1999).

**Patrones GoF**

Se encargan de solucionar problemas de composición de clases y objetos, creación de instancias, integración y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan para el diseño básico del sistema (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

- **Fachada:** la clase definida que ofrece una interfaz común con un conjunto heterogéneo de interfaces. Las interfaces heterogéneas pueden ser un conjunto de funciones, un esquema, un grupo de otras clases o un subsistema.

## 1.8 Métricas de diseño

Las métricas de software proporcionan una forma cuantitativa para valorar la calidad de los atributos internos de producto y, por tanto, permiten valorar la calidad antes de construir el producto. Las métricas proporcionan la comprensión necesaria para crear modelos efectivos de requerimientos y diseño, código sólido y pruebas amplias que le permite a los desarrolladores tener una visión profunda de la eficacia del proceso del software y de los proyectos que dirigen, utilizando el proceso como un marco de trabajo. Se reúnen los datos básicos de calidad y productividad. Estos datos son entonces analizados, comparados con promedios anteriores, y evaluados para determinar las mejoras en la calidad y productividad. Las métricas son también utilizadas para señalar áreas con problemas, de manera que se puedan desarrollar los remedios y mejorar el proceso del software (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

Las métricas están diseñadas para evaluar los siguientes atributos de calidad (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002):

- **Responsabilidad:** Consiste en la responsabilidad asignada a una clase en un marco de modelado de un dominio o concepto, de la problemática propuesta.
- **Complejidad de implementación:** Consiste en el grado de dificultad que tiene implementado un diseño de clases determinado.
- **Reutilización:** Consiste en el grado de reutilización presente en una clase o estructura de clase, dentro de un diseño de software.
- **Acoplamiento:** Consiste en el grado de dependencia o interconexión de una clase o estructura de clase, con otras, está muy ligada a la característica de Reutilización.
- **Complejidad del mantenimiento:** Consiste en el grado de esfuerzo necesario a realizar para desarrollar un arreglo, una mejora o una rectificación de algún error de un diseño de software. Puede influir indirecta, pero fuertemente en los costes y la planificación del proyecto.
- **Cantidad de pruebas:** Consiste en el número o el grado de esfuerzo para realizar las pruebas de calidad (Unidad) del producto (componente, módulo, clase, conjunto de clases, etc.) diseñado.

### 1.8.1 Tamaño operacional de clase (TOC)

Está dado por el número de métodos asignados a una clase y evalúa los siguientes atributos de calidad (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002):

- **Responsabilidad:** Un aumento del TOC implica un aumento de la responsabilidad asignada a la clase.
- **Complejidad de implementación:** Un aumento del TOC implica un aumento de la complejidad de implementación de la clase.
- **Reutilización:** Un aumento del TOC implica una disminución del grado de reutilización de la clase.

### 1.8.2 Relaciones entre clases (RC)

Las relaciones entre las clases están dadas por el número de relaciones de uso que tenga una clase con otras, o sea el número de dependencias que una clase tiene con otra. Mediante la cual se calcula el Acoplamiento, la Complejidad de mantenimiento, la Reutilización y la Cantidad de pruebas a fin de inspeccionar la efectividad del diseño, existiendo una relación directa con los tres primeros e inversa con el último antes mencionado (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

- **Acoplamiento:** Un aumento del RC implica un aumento del Acoplamiento de la clase.
- **Complejidad de mantenimiento:** Un aumento del RC implica un aumento de la complejidad del mantenimiento de la clase.
- **Reutilización:** Un aumento del RC implica una disminución en el grado de reutilización de la clase.
- **Cantidad de pruebas:** Un aumento del RC implica un aumento de la Cantidad de pruebas de unidad necesarias para probar una clase.

## 1.9 Pruebas de software

El aumento de la complejidad de los sistemas de software propicia el aumento de la necesidad de garantizar su calidad. Las pruebas del sistema son una técnica que ayuda a garantizar la calidad del software, partiendo de un conjunto de actividades que pueden planearse por adelantado y realizarse de manera sistemática. Por esta razón, durante el proceso de software, debe definirse una plantilla para la prueba del software: un conjunto de pasos que incluyen métodos de prueba y técnicas de diseño de casos de prueba específicos (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

### 1.9.1 Estrategia de prueba

La Estrategia de Prueba de software es un conjunto de actividades que, durante el proceso de desarrollo de software, describen los pasos que hay que llevar a cabo en un proceso de prueba, que incluyen: la planificación, el diseño de casos de prueba, la ejecución y los resultados, teniendo en cuenta el esfuerzo y recursos que se van a requerir. La estrategia de pruebas se hace con el objetivo de que el producto de software que se encuentre en desarrollo, reúna con todos los requisitos planteados por el cliente mediante la lógica del negocio (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

### 1.9.2 Niveles de Prueba

Las pruebas son aplicadas con diferentes objetivos y en diferentes escenarios. Por lo que se agrupan por niveles como:

**Nivel 1: Pruebas de unidad**

Es el nombre que reciben los procedimientos de pruebas locales a un módulo del sistema. Por definición dichas pruebas cubren la funcionalidad propia del módulo tanto con los métodos de pruebas caja blanca como de caja negra. Combinando ambos métodos se obtiene una mayor fiabilidad del producto final (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

**Nivel 2: Pruebas de integración**

Su principal objetivo es asegurar que los componentes en el modelo de implementación funcionen correctamente una vez integrados, de manera ascendente y descendente, para ejecutar un caso de uso (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

**Nivel 3: Pruebas del sistema**

Las pruebas del sistema, también llamadas como pruebas funcionales es una serie de diferentes pruebas cuyo propósito principal es ejercitar por completo el sistema basado en computadora. Aunque cada prueba tenga un propósito diferente, todas funcionan para verificar que los elementos del sistema se hayan integrado de manera adecuada y que se realicen las funciones asignadas (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

**Nivel 4: Pruebas de aceptación**

Es la prueba final que se hace antes del despliegue del sistema. Su objetivo es verificar que el software esté listo y que puede ser usado por los usuarios finales para ejecutar

las funciones y tareas por lo cual fue construido (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

### 1.10 Conclusiones del capítulo

El estudio de los conceptos más esenciales asociados a la investigación permitió un mejor entendimiento sobre el negocio para su modelado.

Los sistemas Aspen, DDS y EasyScan estudiados tienen una aproximación amplia a la solución deseada pero no pueden ser parte de la propuesta porque son sistemas costosos y no garantizan la obtención de objetos digitales con valor legal.

Los sistemas DigiPyrus, DigiDAP y CDA contribuyen aportes significativos a las funcionalidades y características arquitectónicas que deben estandarizar para así facilitar la gestión de los fondos documentales para el proceso de digitalización del sistema DIGILEX.

El proceso de desarrollo estará guiado por una variación de la metodología AUP establecida previamente por el equipo de desarrollo a partir de las necesidades propias del entorno del proyecto de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI.

El análisis sobre las tecnologías de desarrollo a utilizar determinó la selección como herramienta CASE para la obtención de los elementos del diseño el Visual Paradigm en su versión 8.0, el UML 2.1 como lenguaje de modelado, el lenguaje de programación será Java en su versión 1.8, el Entorno de Desarrollo Integrado NetBeans 8.0 por el soporte relacionado a la Plataforma NetBeans que sustentará la propuesta de solución y como marco de trabajo XEGFORT en su versión 1.2.2, ya que es el marco de trabajo definido por el proyecto.

Con el estudio de los diferentes tipos de arquitecturas y patrones se determinó el análisis y diseño a emplear en la implementación de módulo.

El estudio de las métricas de diseño de software y los diferentes tipos de pruebas posibilito una mayor visión para obtener un producto confiable.

## Capítulo 2: Características y diseño del módulo

### 2.1 Introducción

En este capítulo se realiza una propuesta de solución del módulo para darle cumplimiento al objetivo general mediante la descripción detallada del proceso que se encuentra involucrado en el dominio y fuertemente asociado a nuestro campo de acción, teniendo en cuenta las fases que presenta la metodología ágil de desarrollo de software

AUP-UCI. Con la descripción de la historia de usuario y el diseño del módulo mediante el diagrama de clases se facilita una mayor documentación y comprensión del módulo a desarrollar. Además, se definen los patrones de diseño, la arquitectura y el modelo de datos.

## 2.2 Modelado del negocio

La modelación del negocio consiste en comprender los procesos de negocio de una organización que se desean informatizar, con el objetivo de garantizar que el software desarrollado va a responder las necesidades del cliente. En este proceso el equipo de desarrollo de software debe tener un contacto directo con el cliente y se puede auxiliar de su documentación como: la misión, visión, objetivos, políticas, estrategias de la entidad, sus procesos de negocio, su estructura, organización (Hernández Aguilar, 2010).

Para comprender el contexto del negocio a informatizar en la aplicación DIGILEX, se elabora un modelo conceptual, en el que se evidencia los conceptos, sus características y las relaciones existentes entre ellos.

### 2.2.1 Modelo conceptual

El Modelo conceptual es una representación visual de los objetos y conceptos (o partes esenciales) que se requieren en la modelación de un problema determinado y las relaciones que subsisten entre ellos (Guibert Estrada, 2011). En la Figura 2 se muestra el Modelo conceptual que representa los conceptos del contexto del negocio referente a la gestión documental.

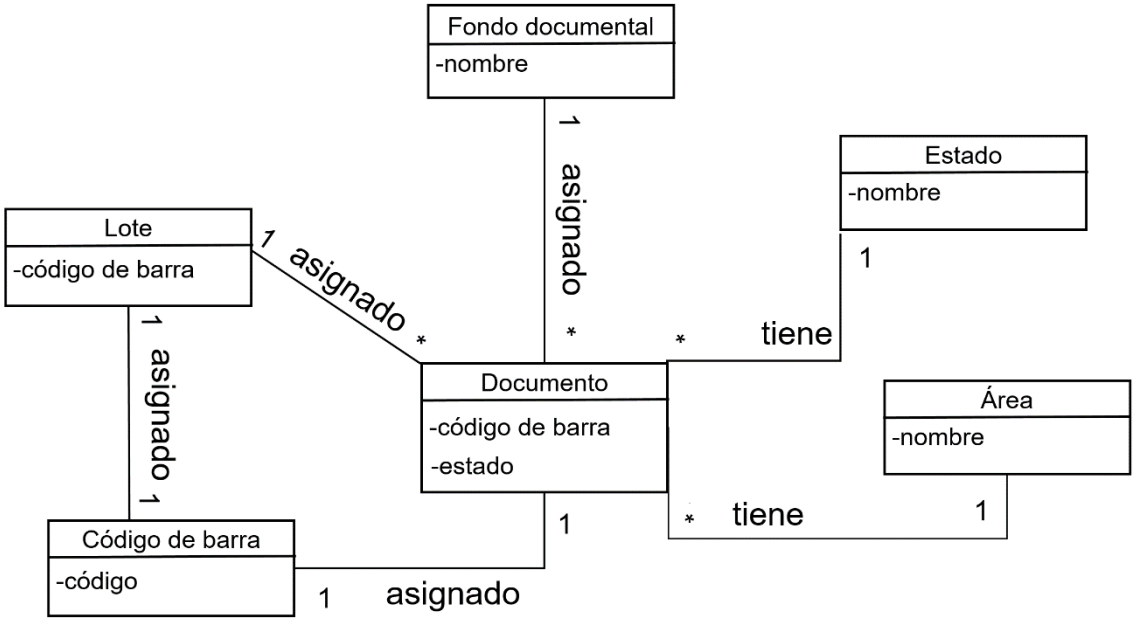



**Figura 2: Modelo conceptual para la gestión documental.**

Los conceptos del contexto del negocio anteriormente modelado son los siguientes:

- **Documento:** contiene el área, código de barra y estado que poseen los documentos.
- **Estado:** contiene el estado por el cual atraviesa un documento en el sistema.
- **Fondo documental:** contiene el formato o soporte de un documento.
- **Área:** contiene el área en cual se encuentra el documento.
- **Lote:** es el que contiene los documentos.
- **Código de barra:** es asignado al documento y lote.

Para comprender el contexto del negocio se elaboró el modelo conceptual, en el que se definen los conceptos asociados a la gestión de fondos documentales para su digitalización para el sistema DIGILEX, así como las características de estos conceptos y las relaciones existentes entre ellos. El modelo conceptual elaborado se utiliza como base para la definición de las funcionalidades y restricciones que debe cumplir el sistema.

## 2.3 Especificación de los requisitos de software

La especificación de los requisitos de software fue una tarea realizada por los analistas del proyecto en la fase de levantamiento de requisitos los cuales se corresponden con la descripción completa del comportamiento del sistema a desarrollar .Incluye un conjunto de tareas que ayudan a comprender cuál será el impacto del software sobre el negocio, qué es lo que quiere el cliente y como interactuarán los usuarios finales con el software (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

### 2.3.1 Requisitos funcionales (RF)

En el contexto de esta investigación, se entiende por requisitos funcionales a las condiciones o capacidades que el sistema debe cumplir. A continuación, se enumeran los requisitos funcionales identificados en el desarrollo de la solución:

**RF1**–Crear lote.

**RF2**–Listar lotes.

**RF3**–Editar Lotes.

**RF4**–Eliminar lote.

**RF5**–Crear documento.

**RF6**–Listar documentos.

**RF7**–Editar documento.

**RF8**–Eliminar documento.

**RF9**–Generar códigos de barras.

**RF10**–Registrar datos generales del envío.

**RF11**–Registrar personas responsables del envío.

### 2.3.2 Requisitos no funcionales (RNF)

Los requisitos no funcionales son aquellos que no se refieren directamente a las funcionalidades específicas que proporciona el sistema sino a las propiedades y restricciones del sistema, como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. Estos requisitos requieren de mayor atención en algunas ocasiones que los requisitos funcionales ya que son normalmente a los que debe apuntar la arquitectura y si estos no son cumplidos, el software puede no funcionar o el cliente simplemente no aceptar el producto.

**Requerimientos de Usabilidad**

**RNF1.** El sistema debe ser usado por personas que tengan conocimientos básicos de informática y deberá permitir el trabajo en equipo de forma colaborativa, de forma tal que varios usuarios trabajen sobre el mismo proyecto posibilitando así que la información pueda integrarse sin dificultad ni esfuerzo.

**Requerimientos de Portabilidad**

**RNF2.** Las herramientas podrán ser usadas bajo cualquier sistema operativo de Windows o cualquier distribución de Linux.

**Requerimientos de Apariencia o Interfaz Externa**

**RNF3.** El sistema debe ofrecer una apariencia profesional con una interfaz amigable y fácil de operar.

**RNF4.** Se debe mantener la línea de diseño establecida, con la uniformidad y representatividad de la solución.

**RNF5.** Todas las interfaces deben estar en idioma español.

**Requerimientos de Software**

**RNF6**. Se requiere tener instalado el PostgreSQL en su versión 9.4, JDK en su versión 1.8.6.9, Glassfish Server en su versión 4.0 y el NetBeans como IDE de desarrollo en su versión 8.1, para el funcionamiento de la aplicación.

**Requerimientos de Hardware**

**RNF7**. La PC del usuario debe requerir como mínimo de RAM 1 GB y un disco duro que posea como mínimo 5 GB para almacenar cierta cantidad de información.

**RNF8.** El servidor debe requerir como mínimo 4 GB de RAM y 500 GB de disco duro para guardar los objetos digitales generados.

## 2.4 Historias de Usuario (HU)

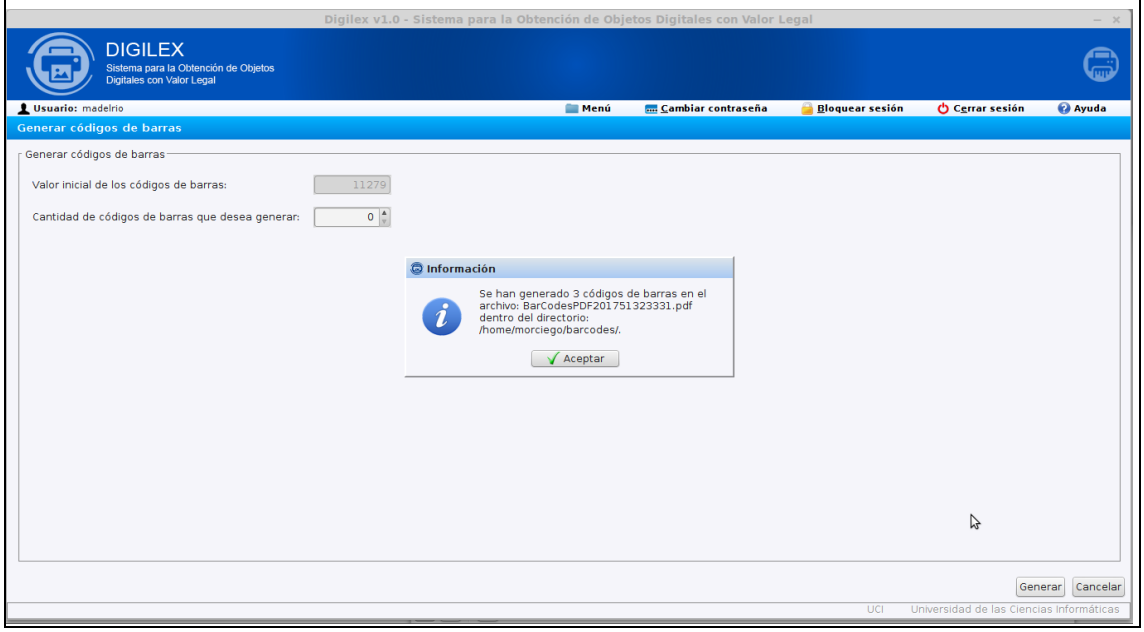
Las historias de usuario son utilizadas como herramienta para dar a conocer los requerimientos del sistema al equipo de desarrollo. Son pequeños textos en los que el cliente describe una actividad que realizara el sistema; la redacción de los mismos se realizara bajo la terminología del cliente, no la del desarrollador, de forma que sea clara y sencilla, sin profundizar en detalles. Las historias de usuarios también son utilizadas para estimar el tiempo que el equipo de desarrollo tomara para realizar las entregas. En una entrega se puede desarrollar una o varias historias de usuario, esto depende del tiempo que demore la implementación de cada una de las mismas (Echeverry Tobón, 2007).

De los requisitos funcionales que se especificaron se realizaron sus historias de usuarios. A continuación, se muestra un ejemplo de la historia de usuario del requisito funcional Generar códigos de barra.

**Tabla 2: Historia usuario Generar códigos de barra.**

| **Número:** 2 | **Nombre del requisito:** Generar códigos de barra |
|---|---|
| **Programador:** Marcelo Adrián del Rio Rodríguez | **Iteración Asignada:** 1 |
| **Prioridad:** Alta | **Tiempo Estimado:** 16 h |
| **Riesgo en Desarrollo:** N/A | **Tiempo Real:** 40 h |
| **Descripción:**<br>Para generar el código de barra el usuario debe seleccionar la cantidad de código de barra que desee generar. Luego el usuario debe dar clic en el botón Generar y se debe mostrar una ventana de confirmación. | |
| **Código del lote:** (*código de barra del lote*), longitud de 13 caracteres, permite solo letras [a-Z] y números [0-9]. | |
| **Observaciones:** N/A | |

**Prototipo elemental de interfaz gráfica de usuario:**




## 2.5 Técnicas de validación de requisitos

Con el objetivo de demostrar que los requerimientos previamente definidos cumplen con las expectativas del cliente, se aplicaron las técnicas de validación de requisitos:

- **Revisión de requisitos:** se realizan reuniones para localizar errores en el documento. Donde se agregaron requisitos y se modificaron otros.
- **Generación de casos de prueba de aceptación:** para validar los requisitos funcionales de la solución, se diseñan casos de pruebas de aceptación para cada una de las Historias de Usuario.
- **Prototipos:** algunas propuestas se basan en obtener de la definición de requisitos prototipos que, sin tener la totalidad de la funcionalidad del sistema, permitan al usuario hacerse una idea de la estructura de la interfaz del sistema con el usuario. Esta técnica tiene el problema de que el usuario debe entender que lo que está viendo es un prototipo y no el sistema final (Pressman R. S., Ingeniería del Software, 2010.).

## 2.6 Análisis y diseño del sistema.

### 2.6.1 Arquitectura del módulo

Atendiendo al análisis realizado en la fundamentación teórica de la investigación (capítulo1), se definió para el módulo una combinación del estilo Cliente/Servidor, distribuida en n capas.

Estas capas son Presentación, Negocio Cliente, Negocio Servidor y Acceso a Datos.

Haciendo uso del estilo cliente/servidor existe un cliente que presenta llamadas a un servidor de aplicaciones mediante el protocolo RMI-IIOP y este le retorna el resultado de la llamada para que se encargue de la presentación de los datos en una interfaz gráfica de usuario.

La **capa de presentación:** es la que interactúa con el usuario, mostrando los datos, que obtiene en la capa negocio cliente. Además, controla los eventos que se generan en la misma. Solo se relaciona con la capa negocio cliente, ya que cada capa solo se comunica con otra vecina a ella.

La **capa de negocio cliente**: gestiona la lógica de negocio relacionada en la parte del cliente, y el acceso a las funcionalidades de los componentes del servidor. La lógica de negocio está implementada sobre un concepto que agrega a la arquitectura, que es el concepto de gestor de negocio del cliente, que no es más que la agrupación de funcionalidades que se refiere a una misma entidad en un gestor con su nombre.

La **capa de negocio servidor**: es responsable de presentar una fachada a todos los componentes que se encuentran en el servidor a modo de gestores de negocio, físicamente alojada en un servidor de aplicaciones al cual se accede de forma remota. Los **gestores de negocio servidor** presenta una interfaz y su implementación contiene la lógica de negocio de las funcionalidades. Además, esta capa hace llamada a la capa de acceso a datos y se lo retorna a la capa de negocio cliente.

La **capa de acceso a datos**: recibe los datos de la capa de negocio servidor y los puede consultar, persistir, actualizar y eliminar en la base de datos, mediante mecanismos que ofrecen las especificaciones utilizadas, que se realizan en conjunto con las entidades persistentes.

Las **entidades persistentes**: representan una relación entre la base de datos y los objetos de la programación, donde cada entidad se corresponde con una tabla.

La **base de datos**: se encuentra ubicada físicamente en el SGBD, en ella se encuentran los datos almacenados listos para ser consultados y actualizados, también se encuentran los mecanismos para facilitar el trabajo con los datos como las funciones y secuencias.

Las **entidades del dominio**: está implícita a través de todas las capas de la arquitectura.

### 2.6.2 Patrones del diseño

Para diseñar el módulo se emplearon un conjunto de patrones de diseño, los cuales son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software. A continuación, se describen los patrones empleados:

**Patrones DAO**

Consiste básicamente en las clases que interactúan con la base de datos. El uso de este patrón se evidencia por ejemplo en la clase PResponsableEntregaDAO (ver figura 3).

**Patrones DTO**

Son utilizados por las clases DAO, para transportar los datos desde la base de datos hasta los gestores de negocio del servidor y viceversa. El uso de este patrón se puede observar en la clase PResponsableEntregaDAO donde el método registrarRespEntrega adquiere por parámetro un objeto de tipo EDResponsableEntrega (ver figura 3).

```
public class PResponsableEntregaDAO {
    @PersistenceContext(unitName = "digiPU")
    EntityManager em;
    EDResponsableEntregaConverter respEntregaConverter;

    public void registrarRespEntrega(EDResponsableEntrega respEntrega) {

        ResponsableEntrega pRespEntrega = respEntregaConverter.convertir(respEntrega);

        em.persist(pRespEntrega);

    }
}
```

**Figura 3: Evidencia de los patrones DAO y DTO.**

**Patrones de diseño GRASP**

**Experto:** se utilizará para que cada objeto realice la funcionalidad de acuerdo a la información que domine, ya que esta se vale de su propia información para realizar las tareas encomendadas. Se evidencia como ejemplo en la clase EDResponsableEntrega (ver figura 4).

```java
public class EDResponsableEntrega implements Serializable {
    private Integer idRespEntrega;
    private String nombre;
    private String primerApellido;
    private String segundoApellido;
    private String CI;
    private String cargo;
    public EDResponsableEntrega() {
        this.idRespEntrega = -1;
        this.nombre = "";
        this.primerApellido = "";
        this.segundoApellido = "";
        this.CI = "";
        this.cargo = "";
    }

    public EDResponsableEntrega(Integer idRespEntrega, String nombre,
    String primerApellido, String segunoApellido, String CI, String carg
        this.idRespEntrega = idRespEntrega;
        this.nombre = nombre;
        this.primerApellido = primerApellido;
        this.segundoApellido = segunoApellido;
        this.CI = CI;
        this.cargo = cargo;
    }
```

**Figura 4: Evidencia del patrón experto.**

**Creador:** son las clases que tienen la responsabilidad de instanciar objetos de otras clases. El uso de este patrón se evidencia en la clase EDLoteConverter.

```java
public class EDLoteConverter {

    @Inject
    EDCodigoBarraConverter codigoBarraConverter;
    @Inject
    EDAreaConverter areaConverter;
    @Inject
    EDEstadoConverter estadoConverter;
    @Inject
    EDDocumentoConverter documentoConverter;
    @Inject
    PDocumentoDAO documentoDAO;
```

**Figura 5: Evidencia del patrón creador.**

**Controlador:** es utilizado para todas las clases implicadas en la capa de presentación, posee responsabilidades de controlar el flujo de eventos mediante las actividades correspondientes. Este patrón se evidencia en la clase AccAlmacen, que es el que se encarga de buscar los atributos y del acceso a la base de datos para obtener finalmente un resultado.

```
public void onBtnTerminar() {
    if ((formulario.jTOficinaProcedencia.getText().equals("")) && (formulario.jSEstante.getValue().hashCode() == 0)
            && (formulario.jSCantidadFolios.getValue().hashCode() == 0)) {
        GestorMensajes.MensajeAviso("Debe llenar todos los campos obligatorios.");
    } else if (formulario.jTOficinaProcedencia.getText().equals("")) {
        GestorMensajes.MensajeAviso("Debe introducir la procedencia del envío.");
    } else if (formulario.jSEstante.getValue().hashCode() == 0) {
        GestorMensajes.MensajeAviso("Debe ubicar en un estante el envío.");

    } else {
        EDAlmacen almacen = new EDAlmacen();
        almacen.setCantCajas((int) formulario.jSCantidadFolios.getValue());
        almacen.setMunicipio(formulario.jTMunicipio.getText());
        almacen.setObservaciones(formulario.jTObservaciones.getText());
        almacen.setOficinaProcedencia(formulario.jTOficinaProcedencia.getText());
        almacen.setProvincia(formulario.jTProvincia.getText());
        almacen.setUbicarEstante((int) formulario.jSEstante.getValue());

        try {
            gestor.guardarDatosGeneralesEnvio(almacen);
            GestorMensajes.MensajeInformacion("Los datos se han adicionado correctamente.");
            limpiarComponentes();
        } catch (Exception ex) {
            GestorExcepcion.reportarExcepcion(ex);
        }
    }
}
```

**Figura 6: Evidencia del patrón controlador.**

**Alta Cohesión:** la principal característica de este patrón es asignar responsabilidades de modo que la cohesión siga siendo alta. La información que almacena una clase debe de ser coherente y debe estar en la medida de lo posible relacionada con la clase. El patrón se evidencia en cada de una de las clases del módulo, de tal forma que se elimina la sobrecarga de responsabilidades.

**Bajo Acoplamiento:** su principal característica es mantener las clases más independientes entre sí y con la menor cantidad de relaciones; la cual posibilita que, en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización y disminuyendo la dependencia entre las clases. El patrón se evidencia en cada una de las clases diseñadas.

### 2.6.3 Estándares de codificación

Se definen estándares de codificación porque un estilo de programación homogéneo en un proyecto permite que todos los participantes lo puedan entender en menos tiempo y

que el código en consecuencia sea mantenible (Calleja, Carmen. Estándares de codificación, 2009).

El estándar de codificación empleado en el desarrollo del módulo fue definido por el equipo de desarrollo. A continuación, se muestran algunas pautas de dicho estándar.

**1)** Todas las nomenclaturas a utilizar se definirán en idioma español, excepto las clases entidades de dominio que terminan en Converter ya que son así definidas por XEFORG.

**2)** Los nombres de los paquetes serán con minúscula y las clases serán con mayúscula, en caso de ser un nombre compuesto las siguientes palabras se escribirán de igual forma (ver figura 7).

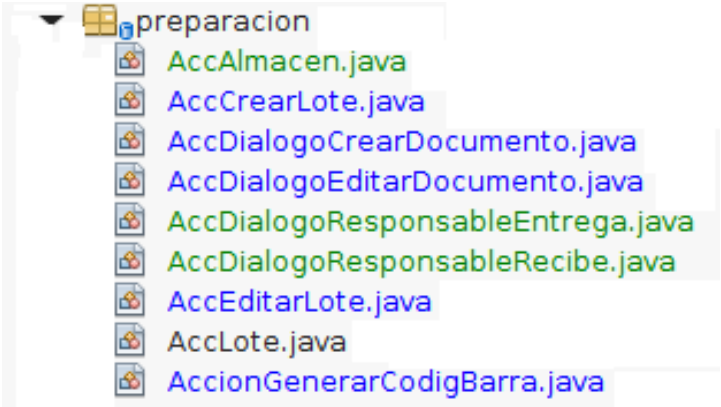



**Figura 7: Estándar de codificación 2.**

**3)** Los nombres de los métodos serán con minúscula, en caso de ser un nombre compuesto las siguientes palabras se escribirán con mayúscula (ver figura 8).

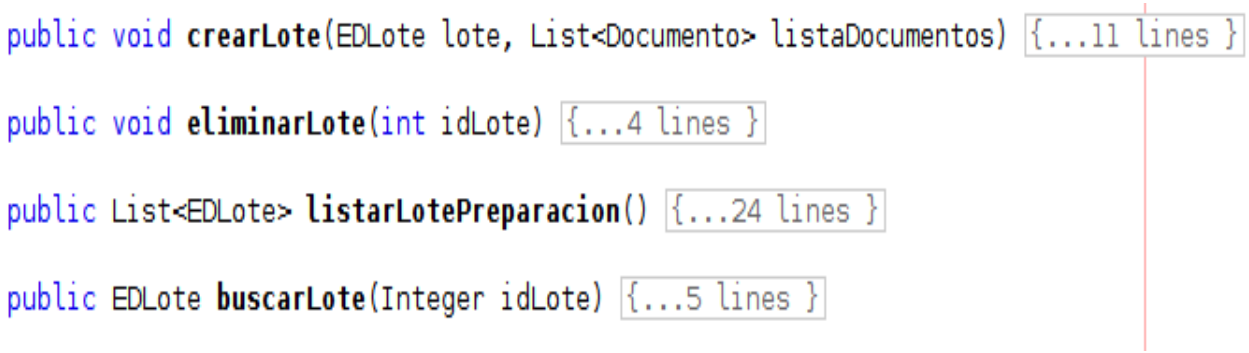



**Figura 8: Estándar de codificación 3**

**4)** Las clases gestoras de negocio comienza con el prefijo Gestor y luego el nombre de la clase.

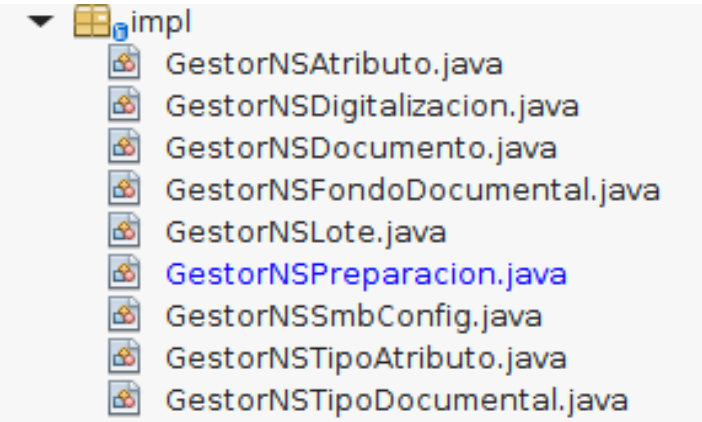



**Figura 9: Estándar de codificación 4.**

**5)** Las clases de acceso a datos comienzan con el prefijo P, el nombre de la clase y luego con el subfijo DAO.


**Figura 10: Estándar de codificación 5.**

### 2.6.4 Diseño de la Base de Datos

Una base de datos correctamente diseñada le proporciona acceso a información actualizada y precisa. Dado que un diseño correcto es esencial para lograr los objetivos en trabajar con una base de datos, dedique tiempo necesaria para obtener información sobre los principios de un buen diseño tenga sentido. Al final, que es mucho más probable que acabe con una base de datos que satisfaga sus necesidades y fácilmente puede acomodar el cambio (Microsoft, 2005).

La estructura de la base de datos del módulo de gestión de un fondo documental del sistema DIGILEX se muestra mediante el modelo de datos. Este modelo fue elaborado utilizando la técnica de modelado de datos Entidad-Relación, mostrando las entidades de datos, sus atributos asociados y las relaciones entre las entidades, a continuación, la descripción de cada una de sus clases:

La base de datos está compuesta por las tablas public.documento, public.area, public.rechazos, public.lote, public.codigo_barra, public.atributo,

public.tipo_documental, public.atr_tipo_doc, public.doc_tipo_doc, public.fondo_documental, public.estado.

public.area: Guarda el área en que se encuentra el documento (Preparación, Digitalización, Metadatos y Firma).

Public.estado: Contiene el estado que tiene el documento (Pendiente, en proceso, finalizado y rechazado).

Public.fondo_documental: almacena los nombres de todos los fondos documentales registrados en el sistema.

Public.documento: almacena todo lo referente al documento (área, código de barra, estado, fondo documental al que pertenece, lote al que pertenece y la imagen digitalizada).

Public.lote: Contiene todos los atributos referentes al lote (el código de barra).

Public.codigo_barra: Almacena los códigos de barras.

Public.tipo_documental: guarda los tipos documentales asociados a un fondo documental.

Public.doc_tipo_doc: almacena el valor de la relación que hay entre un documento y el tipo documental asociado a ese documento.

Almacén: almacena los datos generales del envío.

ResponsableEntrega: almacena los datos del personal encargado de entregar el fondo documental.

ResponsableRecibe: almacena los datos del personal encargado de recibir el fondo documental.

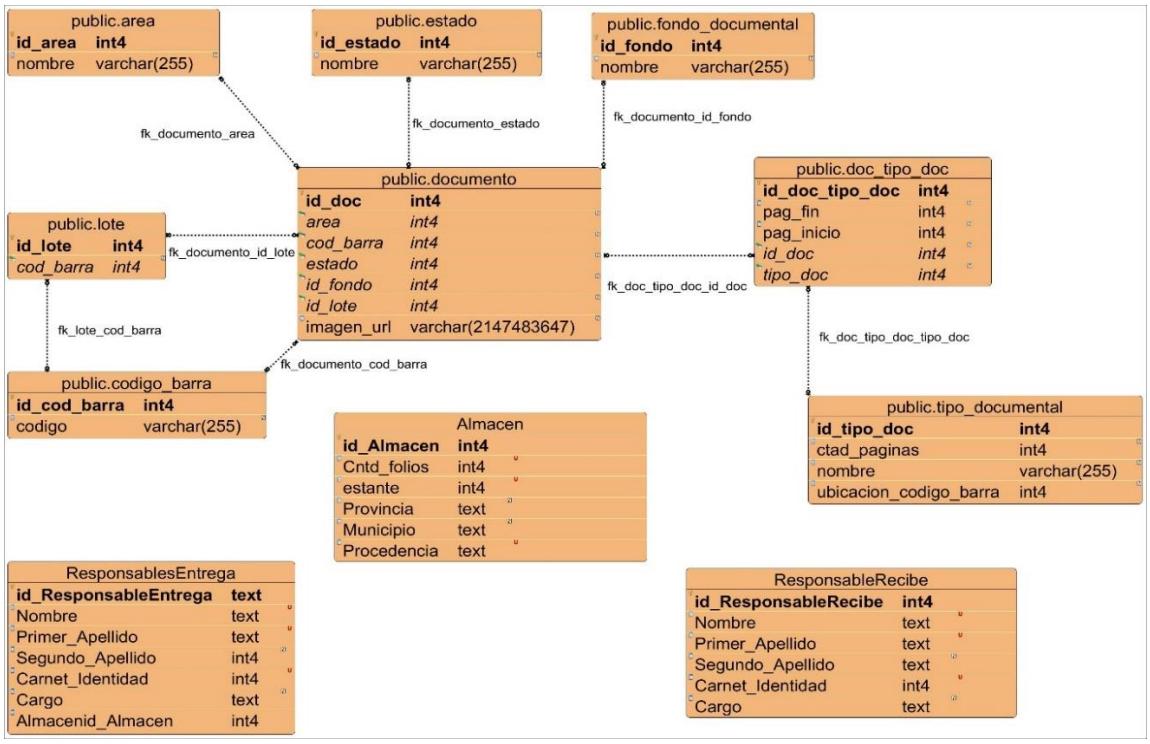



**Figura 11: Modelo de datos.**

### 2.6.5 Diagrama de paquetes

El siguiente diagrama de paquetes muestra las agrupaciones lógicas y las dependencias relacionadas en la HU Crear documento facilitando una descomposición de la jerarquía lógica del módulo.

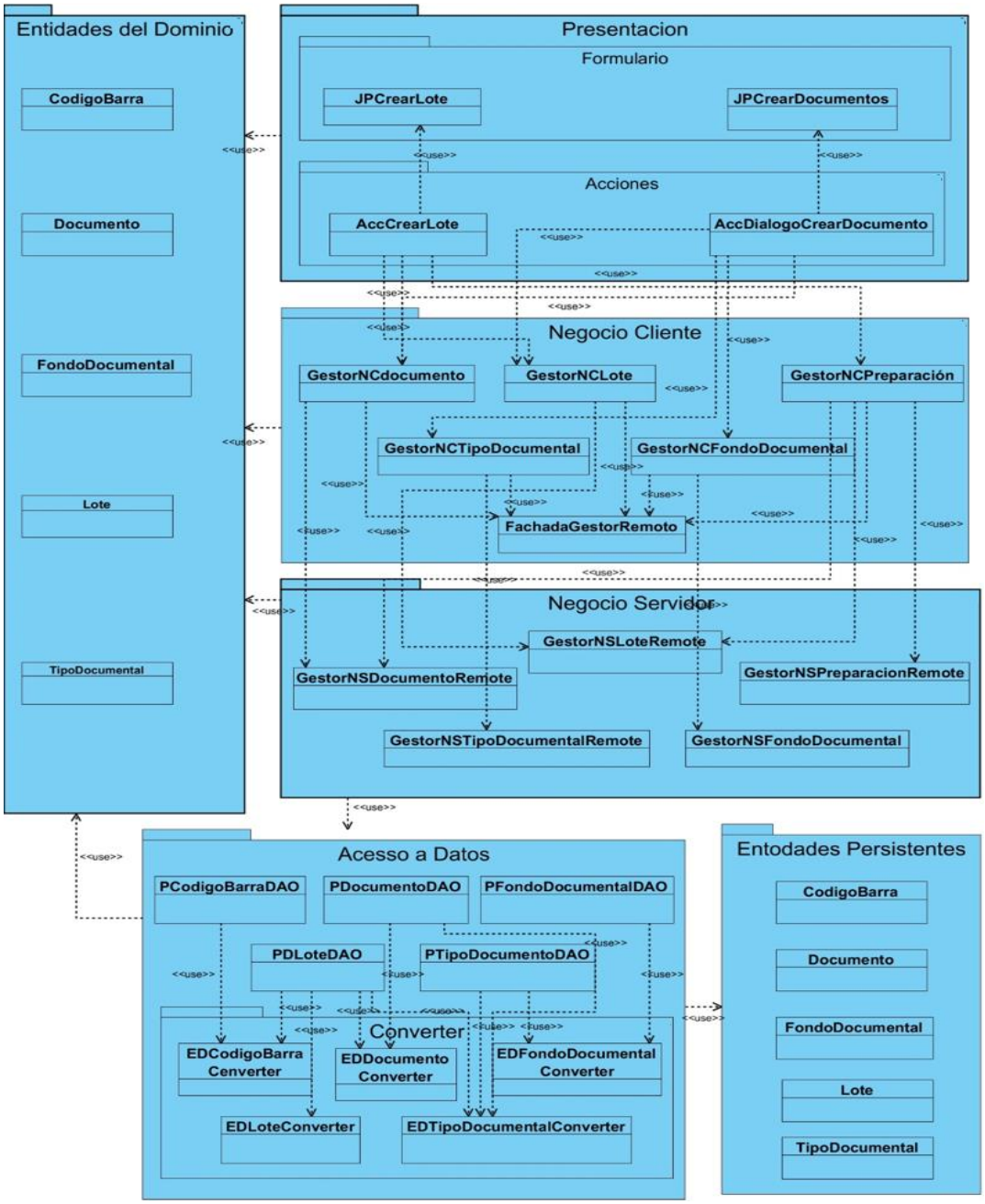



**Figura 12: Distribución de paquete de la HU Crear Documento**.

## 2.7 Verificación del diseño

### 2.7.1 Tamaño operacional de la clase (TOC)

Para evaluar el diseño de la solución, se decide aplicar la métrica TOC a un total de 27 clases, aplicando el criterio de evaluación de las métricas (Tabla 3) y el instrumento de evaluación de evaluación de las métricas (Tabla 4) para un promedio de procedimientos de 5,592592593 se obtuvieron los siguientes resultados.

**Tabla 3: Criterio de evaluación de las métricas TOC.**

| Atributo | Categoría | Criterio |
|---|---|---|
| Responsabilidad | Baja | <=Promedio |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | > 2* Promedio |
| Complejidad de implementación | Baja | <=Promedio |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | > 2* Promedio |
| Reutilización | Baja | > 2* Promedio |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | <=Promedio |

**Tabla 4: Instrumento de evaluación de las métricas TOC.**

| Clase | Cantidad de Procedimientos | Responsabilidad | Complejidad | Reutilización |
|---|---|---|---|---|
| AccCrearLote | 10 | Media | Media | Media |
| ACCDialogoCrearDocumentos | 4 | Baja | Baja | Alta |
| AccDialogoEditarDocumento | 4 | Baja | Baja | Alta |
| AccEditarLote | 11 | Media | Media | Alta |
| AccLote | 7 | Media | Media | Media |
| AccGenerarCodigoBarra | 2 | Baja | Baja | Alta |
| AccAlmacen | 8 | Media | Media | Media |

| | | | | |
|---|---|---|---|---|
| AccDialogoResponsableEntrega | 5 | Baja | Baja | Alta |
| AccDialogoResponsableRecibe | 5 | Baja | Baja | Alta |
| GestorNCPreparacion | 6 | Baja | Baja | Alta |
| GestorNCLote | 7 | Media | Media | Media |
| GestorNCDocumento | 7 | Media | Media | Media |
| EDCodigoBarraConverter | 7 | Media | Media | Media |
| EDDocumentoConverter | 5 | Baja | Baja | Alta |
| EDLoteConverter | 4 | Baja | Baja | Alta |
| EDAlmacenConverter | 2 | Baja | Baja | Alta |
| EDResponsableEntregaConverter | 2 | Baja | Baja | Alta |
| EDResponsableRecibeConverter | 2 | Baja | Baja | Alta |
| PCodigoBarraDAO | 3 | Baja | Baja | Alta |
| PDocumentoDAO | 10 | Media | Media | Media |
| PloteDAO | 13 | Media | Media | Media |
| PResponsableEntregaDAO | 1 | Baja | Baja | Alta |
| PResponsableRecibeDAO | 1 | Baja | Baja | Alta |
| PAlmacenDAO | 1 | Baja | Baja | Alta |
| GestorNSPreparacion | 5 | Baja | Baja | Baja |
| GestorNSLote | 9 | Media | Media | Media |
| GestorNSDocumento | 10 | Media | Media | Media |

La figura 13 muestra la representación de los resultados obtenidos en la herramienta agrupados en los intervalos definidos. El gráfico refleja que la mayoría de las clases tienen de 6 a 10 procedimientos. Este resultado demuestra que el funcionamiento general del módulo está distribuido equitativamente entre la mayoría de las clases, de esta forma se garantiza que, en caso de ocurrir algún cambio en el sistema de clases del módulo, estaría menos comprometido el correcto funcionamiento del mismo.

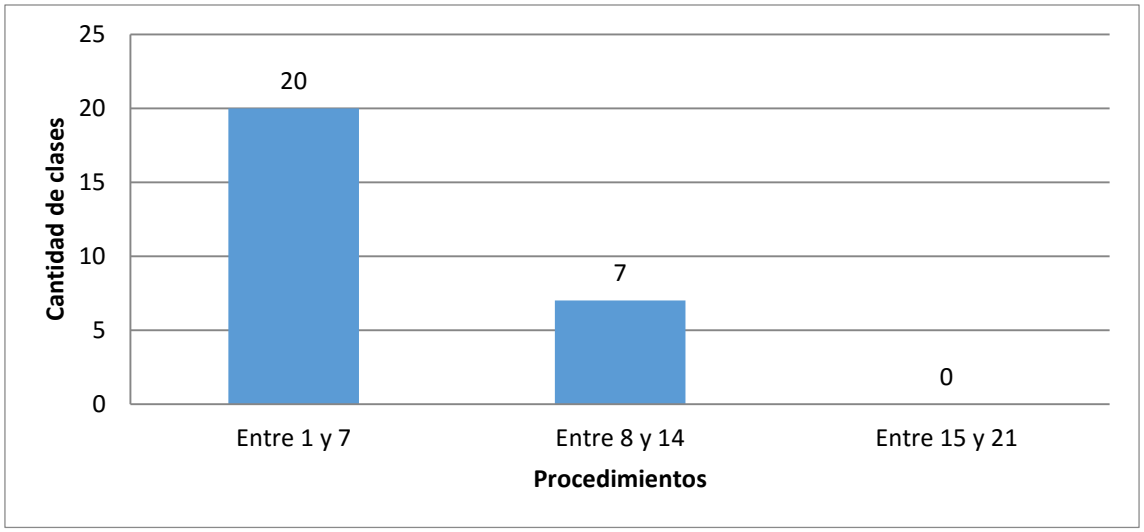



**Figura 13: Gráfica correspondiente a la representación de la evaluación de la métrica TOC.**

La siguiente figura muestra la representación en % de los resultados obtenidos en la herramienta agrupados en los intervalos definidos.

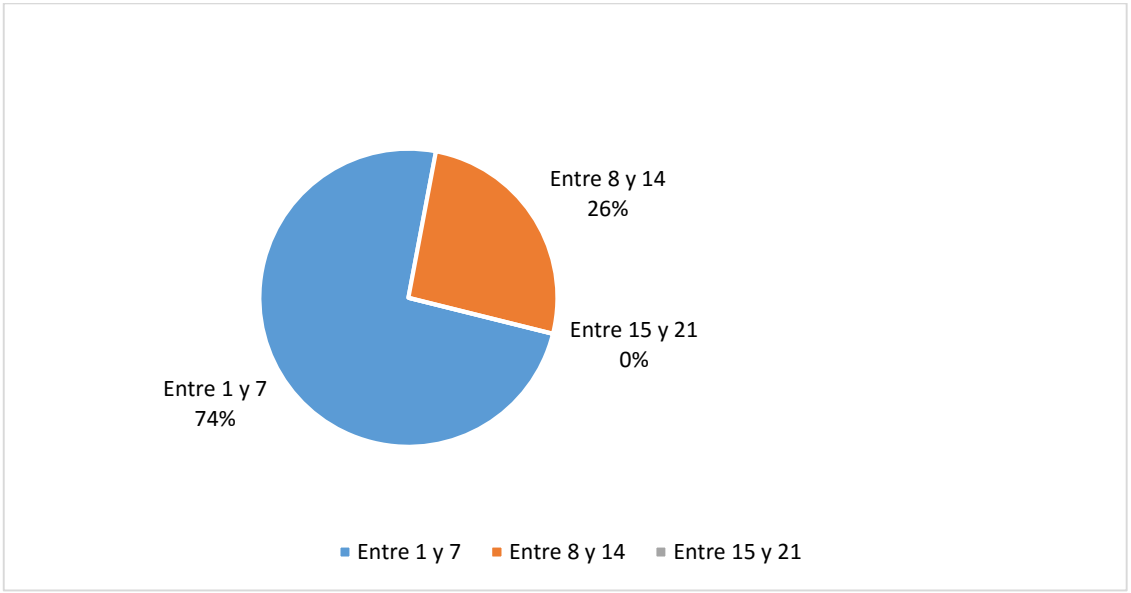



**Figura 14: Evaluación de la métrica TOC en por ciento.**

En la Figura 15 se puede observar el resultado de la evaluación de los atributos de la métrica TOC.

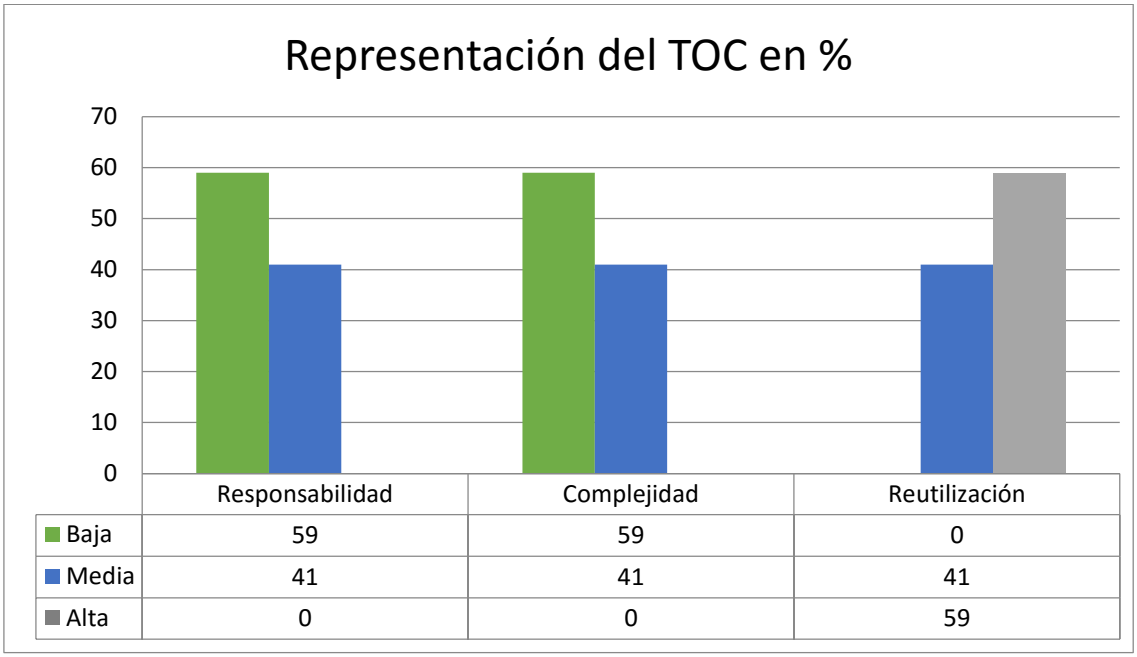



**Figura 15: Representación de los resultados de la métrica TOC.**

Haciendo un análisis de los resultados obtenidos en la evaluación de la herramienta de medición de la métrica TOC, se puede concluir que el diseño del módulo tiene una calidad aceptable teniendo en cuenta que el 59 % de las clases incluidas en este módulo, poseen una alta reutilización. Además, este mismo por ciento de clases poseen evaluaciones positivas en los atributos de calidad (Responsabilidad y Complejidad de Implementación) en el diseño propuesto, evidenciando que la mayoría de las clases no tienen tanta responsabilidad, no son tan complejas, y poseen un elevado grado de reutilización. Por lo que se concluye que los resultados obtenidos según esta métrica son positivos.

### 2.6.2 Relaciones entre clases RC

La métrica RC se aplica igualmente a un total de 27 clases, con un promedio de asociaciones de uso de 1.625, empleando el criterio de evaluación de las métricas (Tabla 5) y el instrumento de evaluación de evaluación de las métricas (Tabla 6) se obtuvieron los siguientes resultados.

**Tabla 5: Criterios de evaluación de las métricas RC.**

| Atributo | Categoría | Criterio |
|---|---|---|
| Acoplamiento | Ninguno | 0 |

| | | |
|---|---|---|
| | Bajo | 1 |
| | Medio | 2 |
| | Alto | >2 |
| Complejidad del mantenimiento | Baja | <=Promedio |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | > 2* Promedio |
| Cantidad de pruebas | Baja | <=Promedio |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | > 2* Promedio |
| Reutilización | Baja | > 2* Promedio |
| | Media | Entre Promedio y 2* Promedio |
| | Alta | <=Promedio |

**Tabla 6: Instrumento de evaluación de las métricas RC.**

| Clase | Cantidad de Relaciones de uso | Acoplamiento | Complejidad del Mantenimiento | Reutilización | Cantidad de Pruebas |
|---|---|---|---|---|---|
| AccCrearLote | 1 | Bajo | Baja | Alta | Baja |
| ACCDialogoCrearDocumentos | 2 | Medio | Media | Media | Media |
| AccDialogoEditarDocumento | 2 | Medio | Media | Media | Media |
| AccEditarLote | 3 | Alto | Media | Media | Media |
| AccLote | 3 | Alto | Media | Media | Media |
| AccGenerarCodigoBarra | 1 | Bajo | Baja | Alta | Baja |

| | | | | | |
|---|---|---|---|---|---|
| AccAlmacen | 1 | Bajo | Baja | Alta | Baja |
| AccDialogoResponsableEntrega | 2 | Medio | Media | Media | Media |
| AccDialogoResponsableRecibe | 2 | Medio | Media | Media | Media |
| GestorNCPreparacion | 1 | Bajo | Baja | Alta | Baja |
| GestorNCLote | 1 | Bajo | Baja | Alta | Baja |
| GestorNCDocumento | 1 | Bajo | Baja | Alta | Baja |
| EDCodigoBarraConverter | 1 | Bajo | Baja | Alta | Baja |
| EDDocumentoConverter | 1 | Bajo | Baja | Alta | Baja |
| EDLoteConverter | 1 | Bajo | Baja | Alta | Baja |
| EDAlmacenConverter | 1 | Bajo | Baja | Alta | Baja |
| EDResponsableEntregaConverter | 1 | Bajo | Baja | Alta | Baja |
| EDResponsableRecibeConverter | 1 | Bajo | Baja | Alta | Baja |
| PCodigoBarraDAO | 1 | Bajo | Baja | Alta | Baja |
| PDocumentoDAO | 3 | Alto | Media | Media | Media |
| PloteDAO | 4 | Alto | Alta | Baja | Alta |
| PAlmacenDAO | 2 | Medio | Media | Media | Media |
| PResponsableEntregaDAO | 2 | Medio | Media | Media | Media |
| PResponsableRecibeDAO | 2 | Medio | Media | Media | Media |
| GestorNSPreparacion | 2 | Medio | Media | Media | Media |

| | | | | | |
|---|---|---|---|---|---|
| GestorNSLote | 2 | Medio | Media | Media | Media |
| GestorNSDocumento | 1 | Bajo | Baja | Alta | Baja |

La Figura 16 muestra la representación de los resultados obtenidos en la herramienta agrupados en los intervalos definidos.

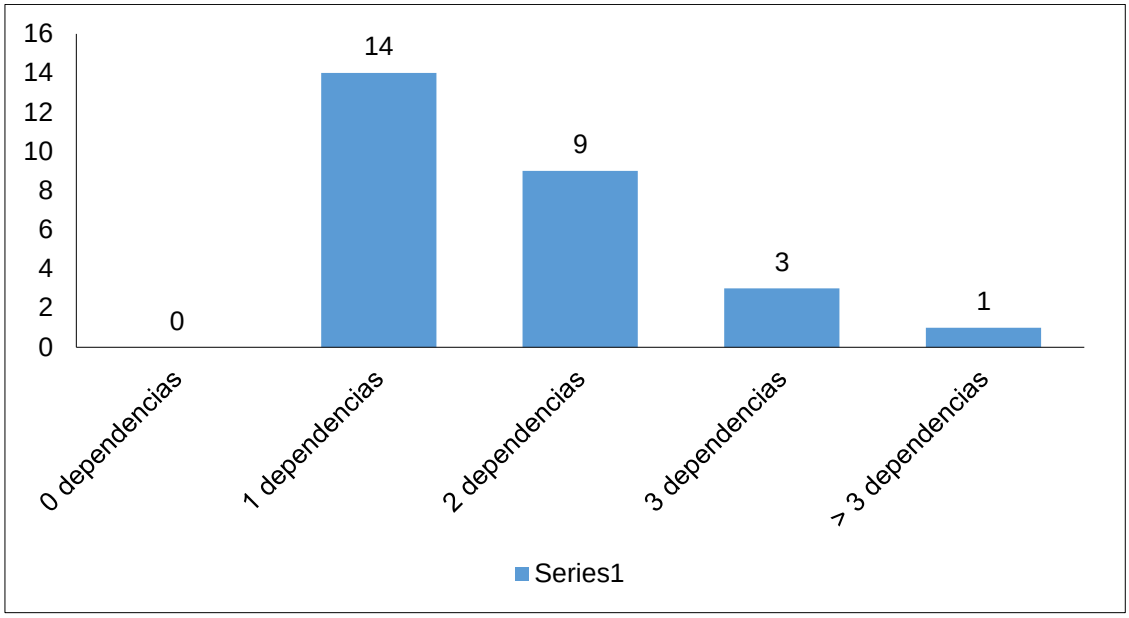
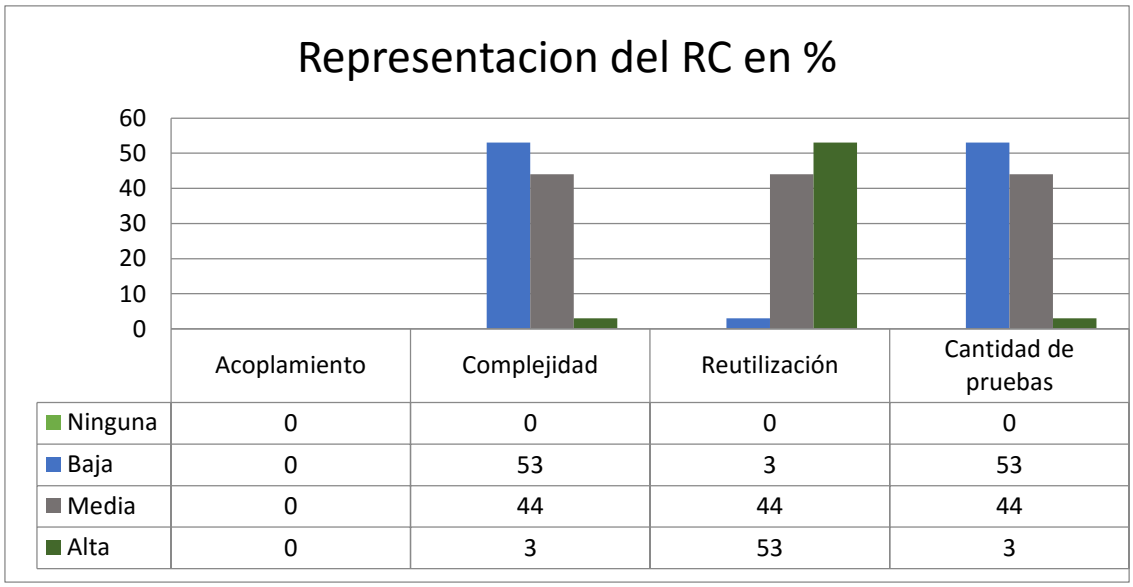



**Figura 16: Resultados de la aplicación de la métrica RC.**

En la Figura 17 se puede observar el resultado de la evaluación de los atributos de la métrica RC.




| | Acoplamiento | Complejidad | Reutilización | Cantidad de pruebas |
|---|---|---|---|---|
| Ninguna | 0 | 0 | 0 | 0 |
| Baja | 0 | 53 | 3 | 53 |
| Media | 0 | 44 | 44 | 44 |
| Alta | 0 | 3 | 53 | 3 |

**Figura 17: Evaluación de los atributos de la métrica RC.**

Haciendo un análisis de los resultados obtenidos en la evaluación de la herramienta de medición de la métrica RC, se puede concluir, que el diseño realizado tiene una buena calidad teniendo en cuenta que el atributo complejidad de mantenimiento posee un 53% de las clases con índices bajo, lo que facilita las tareas de corrección, modificación y mantenimiento de las clases. El número o el grado de esfuerzo para realizar las pruebas de calidad del producto diseñado es bajo dado que la métrica aplicada arroja un 53% de baja cantidad de pruebas fomentando así un alto índice de reutilización con un 53%.

### 2.7 Conclusiones del capítulo:

En el capítulo recién concluido se puede afirmar que:

Con el empleo de la metodología AUP – UCI, se permitió realizar un trabajo organizado y estructurado, generando los artefactos de la fase de ejecución.

Se identificaron los requisitos esenciales para el desarrollo del sistema, obteniéndose un total de 11 requisitos funcionales a tener en cuenta para el desarrollo del módulo.

El uso de la arquitectura cliente servidor distribuidas en n-capas y el empleo de patrones de diseño, garantizó obtener una solución de software factible con poca dependencia entre las clases.

La aplicación de métricas de diseño, demostró que las clases del diseño poseen bajo acoplamiento, que existe además una baja responsabilidad y complejidad de implementación y una alta reutilización en el diseño propuesto.

## Capítulo 3: Validación del módulo

En el ciclo de desarrollo del software la etapa de pruebas constituye un aspecto fundamental, pues permite validar y verificar la calidad que posee el producto desarrollado. Atendiendo al análisis realizado en la fundamentación teórica de la investigación (capítulo1) y a una de las disciplinas establecidas por la metodología AUP – UCI se aplicaron las pruebas unitarias, las funcionales y las de aceptación.

### 3.2 Verificación

La verificación se refiere al conjunto de tareas que garantizan que el software implementa correctamente una función específica (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

#### 3.2.1 Pruebas unitarias

Como prueba unitaria aplicada al módulo se realizó el método de caja blanca o white box. La prueba de caja blanca, es una filosofía de diseño de casos de prueba que usa

la estructura de control descrita como parte del diseño a nivel de componentes para derivar casos de prueba. Al usar los métodos de prueba de caja blanca, puede derivar casos de prueba que (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002):

**1)** garanticen que todas las rutas independientes dentro de un módulo se revisaron al menos una vez.
**2)** revisen todas las decisiones lógicas en sus lados verdadero y falso.
**3)** ejecuten todos los bucles en sus fronteras y dentro de sus fronteras operativas.
**4)** revisen estructuras de datos internas para garantizar su validez.

Para emplear dicha prueba se aplica la técnica de camino básico, la cual es una técnica de prueba de caja blanca que permite al diseñador de casos de prueba derivar una medida de complejidad lógica de un diseño de procedimiento y usar esta medida como guía para definir un conjunto básico de rutas de ejecución (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002) , tomándose como ejemplo el método onBtnTerminar de la clase AccCrearLote. El método seleccionado (figura 18) es uno de los más complejos, ya que es una de las principales funcionalidades que responden el objetivo general.

```
public void onBtnTerminar() {
    try {
        try {
            String codigoBarraStr = formulario.jTCodigoBarraLote.getText();//1
            GestorNCPreparacion gestorNCPreparacion = new GestorNCPreparacion();//2
            if (getFormulario().jTCodigoBarraLote.getText().equals("")) {//3
                GestorMensajes.MensajeAviso("Debe asignar un código de barra al lote.");//4
            } else if (gestorNCPreparacion.existeCodigoBarra(codigoBarraStr)) {//5
                EDCodigoBarra codigoB = gestorNCPreparacion.buscarCodBarra(codigoBarraStr);//6
                if (esCodigoBarraAsociado(codigoBarraStr)) {//7
                    GestorMensajes.MensajeAviso("El código de barra ya fue asignado a un lote o documento existente.");//8
                } else if (esCodigoBarraAsociadoDocumentoLote(codigoBarraStr)) {//9
                    GestorMensajes.MensajeAviso("El código de barra ya está relacionado con un documento del lote.");//10
                } else {//11
                    EDLote lote = new EDLote();   //12
                    lote.setCodigoBarra(codigoB);
                    lote.setDocumentoList((List<EDDocumento>) formulario.jTDocumentos.getFuenteDatos()); //13
                    new GestorNCLote().crearLote(lote);      //14
                    GestorMensajes.MensajeInformacion("El lote se guardó correctamente.");          //15
                    onBtnAnterior();                                                                 //16
                }
            }
            else {//17
                GestorMensajes.MensajeAviso("El código de barra " + codigoBarraStr + " asignado al lote no existe.");//18
            }
        } catch (Exception ex) {
            GestorExcepcion.reportarExcepcion(ex);
        }
    } catch (Exception ex) {
        GestorExcepcion.reportarExcepcion(ex);
    }
}//19
```

**Figura 18: Método onBtnTerminar.**

A continuación, se muestra los procedimientos para realizar la técnica de camino básico:

1) **Confeccionar el grafo de flujo:**

El gráfico de flujo muestra el flujo de control lógico (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002) que usa la notación ilustrada en la figura 19.

**Nodos:** Son círculos que representan una o más sentencias procedimentales.

**Aristas:** Son flechas que representan el flujo de control y son análogas a las flechas del diagrama de flujo.

**Regiones:** son las áreas delimitadas por aristas y nodos.

En la figura 19 se muestra el grafo de flujo obtenido del método onBtnTerminar.

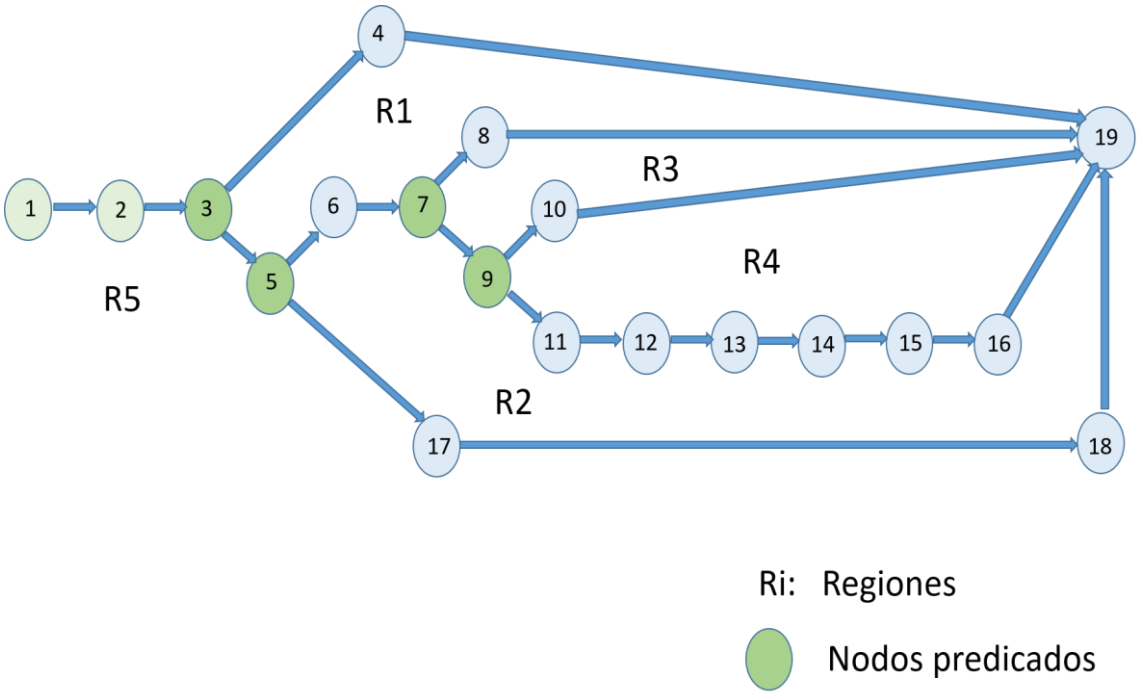



**Figura 19: Grafo de camino básico del método onBtnTerminar.**

**2) Calcular la complejidad ciclomática:**

Cuando se usa en el contexto del método de prueba de la ruta básica, el valor calculado por la complejidad ciclomática define el número de rutas independientes del conjunto básico de un programa y le brinda una cota superior para el número de pruebas que debe realizar a fin de asegurar que todos los enunciados se ejecutaron al menos una vez (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

La complejidad ciclomática tiene fundamentos en la teoría de gráficos y proporciona una medición de software extremadamente útil. La complejidad se calcula de tres formas las cuales deben dar el mismo resultado para comprobar que el cálculo sea correcto (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002) :

- El número de regiones del gráfico de flujo corresponde a la complejidad ciclomática.
- La complejidad ciclomática $V(G)$ para un gráfico de flujo $G$ se define como:

  $V(G)=E-N+2$

  Donde $E$ es el número de aristas del gráfico de flujo y $N$ el número de nodos del gráfico de flujo.
- La complejidad ciclomática $V(G)$ para un gráfico de flujo $G$ también se define como

  $V(G)=P+1$

  Donde $P$ es el número de nodos predicado contenidos en el gráfico de flujo $G$.

En el gráfico de flujo de la figura 19, la complejidad ciclomática puede calcularse usando cada uno de los algoritmos recién indicados:

1. El gráfico de flujo tiene cinco regiones.
2. V(G) = 22 aristas - 19 nodos +2 = 5.
3. V(G) =4 nodos predicado + 1= 5.

Por tanto, la complejidad ciclomática del gráfico de flujo en la figura 18 es 5.

**3) Determinar un conjunto básico de caminos linealmente independientes:**

El valor de V(G) proporciona la cota superior sobre el número de rutas linealmente independientes a través de la estructura de control del programa (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002). En el caso del procedimiento onBtnTerminar, se definen 5 caminos básicos:

**Camino básico #1:** 1, 2, 3, 4,19

**Camino básico #2:** 1, 2, 3, 5, 17, 18,19

**Camino básico #3:**1,2,3,5,6,7,9,11,12,13,14,15,16,19

**Camino básico #4:**1, 2, 3, 5, 6, 7, 9, 10,19

**Camino básico #5:**1, 2, 3, 5, 6, 7, 8,19

**4) Obtención de casos de prueba (CP):**

**Tabla 7: Casos de prueba del camino 5.**

| **Caso de prueba: Camino básico 5** | |
|---|---|
| **Objetivo de la prueba** | Comprobar si se muestra el mensaje “El código de barra ya fue asignado a un lote o documento existente”. |
| **Datos de entrada** | Código de barra. |
| **Procedimientos** | Escribir el código de barra asignado al lote. |
| **Salida esperada** | Mostrar mensaje. |

**Tabla 8: Casos de prueba del camino 3.**

| **Caso de prueba: Camino básico 3** | |
|---|---|
| **Objetivo de la prueba** | Comprobar si se creó el lote y se muestra el mensaje “El lote se guardó correctamente”. |

| | |
|---|---|
| **Datos de entrada** | Código de barra, lista de documentos. lote |
| **Procedimientos** | Escribir el código de barra asignado al lote. |
| **Salida esperada** | Se adiciona el lote y se muestra el mensaje. |

Una vez ejecutada la aplicación de la técnica camino básico al módulo se obtuvo un total de 58 casos de pruebas, garantizando que:

- Todas las rutas linealmente independientes se revisan al menos una vez.
- Se revisen todas las decisiones lógicas en sus lados verdaderos y falsos.
- Se ejecuten todos los bucles en sus fronteras y dentro de sus fronteras operativas.
- Se revisen estructuras de datos internos para garantizar su validez.

Por lo que se puede concluir que todos los casos de pruebas fueron probados satisfactoriamente demostrando que el código generado no presenta ciclos infinitos y no existe código innecesario en el sistema desarrollado.

### 3.2.2 Pruebas funcionales

Como método aplicado al módulo, se utiliza pruebas de caja negra. Estas son pruebas funcionales que se enfocan en los requerimientos funcionales del software lo cual le permiten al ingeniero de software derivar conjuntos de condiciones de entrada que revisarán por completo todos los requerimientos funcionales para un programa. Además, permiten detectar funcionamiento incorrecto o incompleto, errores en la interfaz, errores en la estructura de datos externa, problemas de rendimientos y errores de inicialización y terminación (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

Dentro de la prueba de caja negra se incluyen técnicas de pruebas las cuales son:

**Partición de Equivalencia:** es un método de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos de los que pueden derivarse casos de prueba (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

**Análisis de Valores Límites:** es una técnica de diseño de casos de prueba que complementan la partición de equivalencia la cual prueba la habilidad del

programa para manejar datos que se encuentran en los límites aceptables (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

**Grafos de Causa-Efecto:** permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

De las técnicas de pruebas anteriormente descritas que integran las pruebas de caja negra se seleccionó la técnica de partición de equivalencia, diseñando casos de prueba para cada uno de los requisitos funcionales del módulo. A continuación, se muestra un ejemplo de caso de prueba para la historia de usuario Crear Lote.

**EC1.1** Campos correctos: Escenario donde se introduce en el sistema los datos correctos.

**EC1.2** Campos incorrectos: Escenario donde se introduce en el sistema datos incorrectos.

**EC 1.3** Campos incompletos: Escenario donde existen campos incompletos.

**Tabla 9: Casos de prueba de la HU Crear lote.**

| Escenario | Código de barra del lote | Respuesta del sistema | Flujo central |
|---|---|---|---|
| **EC1.1** Campos correctos | V (HP00000149YRR) | "El lote se guardó correctamente." | Menú/Preparación/Gestionar lotes/Crear lote |
| **EC1.2** Campos incorrectos | I (0000RF667TYUG) | "El código de barra 0000RF667TYUG asignado al lote no existe." | |
| **EC 1.3** Campos incompletos | I (Vacío) | "Debe asignar un código de barra al lote." | |

La siguiente figura muestra la interfaz gráfica que refleja la respuesta del sistema en el escenario 1.1.

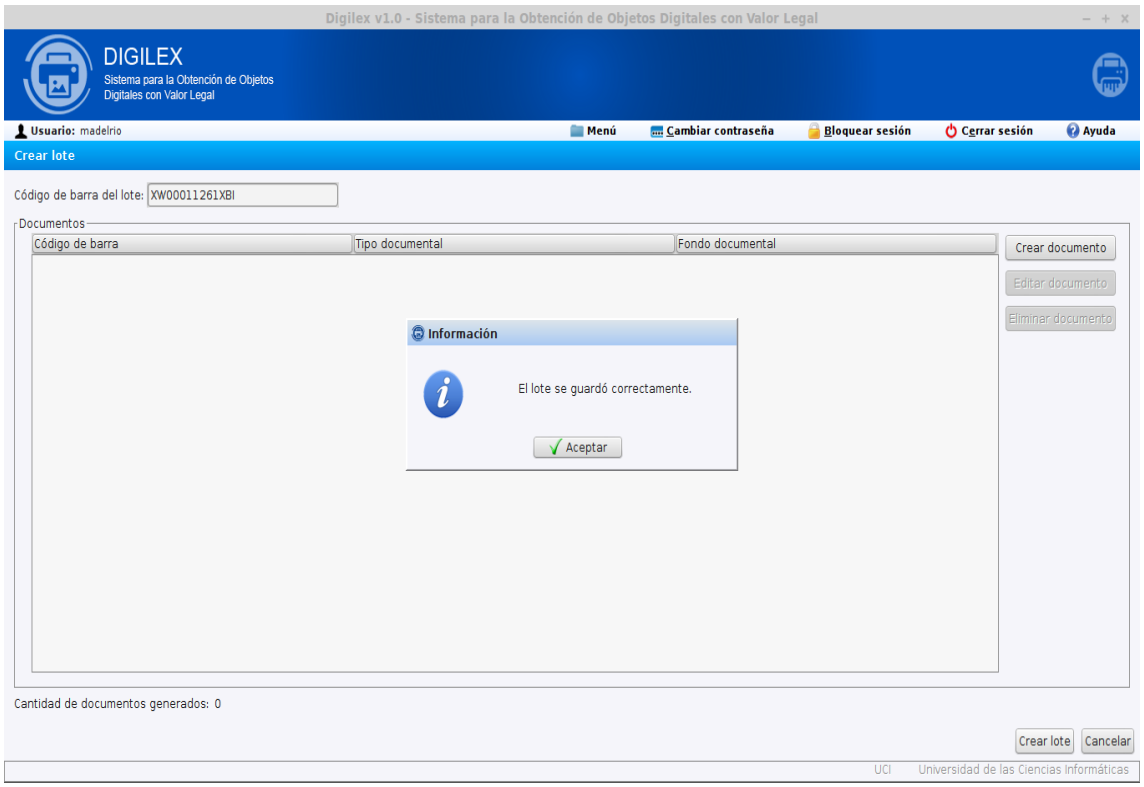



**Figura 20: Interfaz gráfica de la historia usuario Crear lote.**

Con el objetivo de comprobar que las funcionalidades del módulo del sistema se realizan correctamente y responden a las necesidades del cliente, aplicando la técnica anteriormente ejemplificada fueron revisados por el departamento de Calidad CEGEL todos los casos de pruebas elaborados, obteniendo los siguientes resultados.

### Resultado de las pruebas

Se realizaron 3 iteraciones revisando en cada una la aplicación contra los 11 casos de pruebas elaborados. Durante la primera iteración se obtuvo un total de 10 no conformidades de complejidad media, durante la segunda iteración se encontraron 3 no conformidades, de las cuales 2 de complejidad media y 1 de complejidad baja, y durante la tercera iteración no se hallaron no conformidades.

### No Conformidades (NC)

Según la norma ISO 9000:2005, una No Conformidad es un incumplimiento de un requisito del sistema, sea este especificado o no. Se conoce como requisito una necesidad o expectativa establecida, generalmente explícita u obligatoria (Asociación Española para la Calidad, 2017).

La no conformidad detectada es una ausencia de calidad, las cuales pueden estar dadas por faltas de ortografías o errores en el diseño del sistema que puedan afectar su uso.

En la figura 21 se muestra un gráfico con el número de no conformidades obtenidas en el sistema DIGILEX para un total de 11 casos de pruebas.

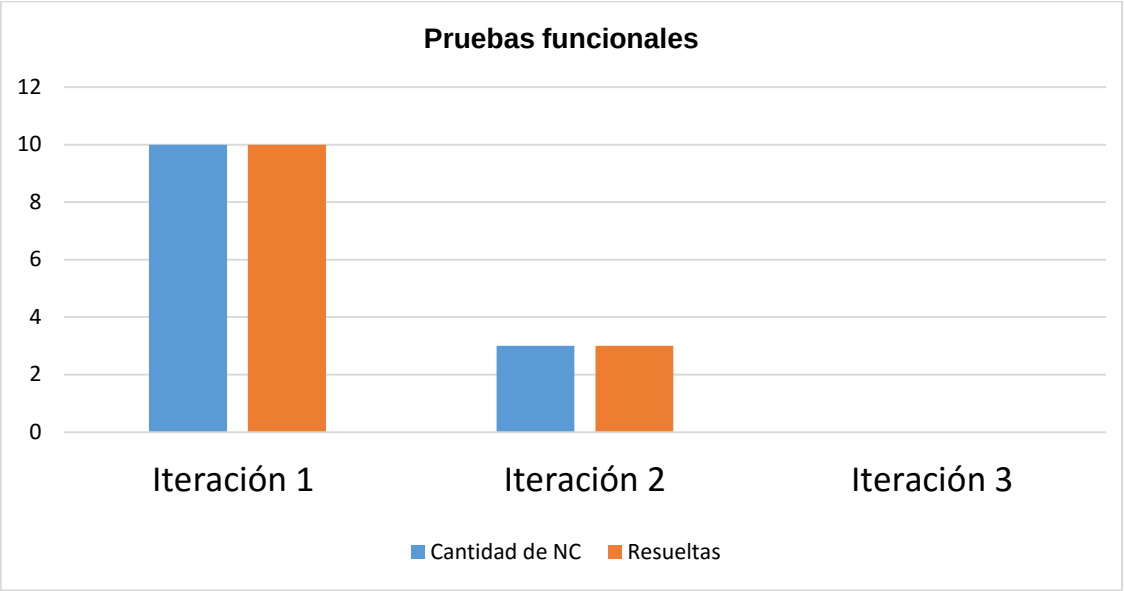



**Figura 21: Resultados de pruebas funcionales.**

Las 10 no conformidades encontradas en la primera iteración fueron:

1. Al introducir un código de barra asociado a otro documento, el sistema no muestra un mensaje de error y se agrega el documento con el código de barra repetido.
2. Cuando se asocia un documento al lote, no se guarda el Tipo Documental.
3. En el campo **Cantidad de códigos de barras a generar** se introducen letras y números y el sistema no muestra un mensaje de error.
4. Al eliminar un documento no se muestra un mensaje de confirmación.
5. Al editar un lote no se muestra un mensaje de confirmación.
6. Al editar un lote y asociarle nuevos documentos y volver a la interfaz **lista de lotes**, la cantidad de documentos no se actualiza.
7. Al seleccionar más de un documento para ser eliminados, solamente se elimina el primero.
8. Cuando se selecciona un documento en la lista de documentos creados de un lote no se activa el botón **Eliminar.**
9. El campo **Carné de Identidad** admite letras y la cantidad de caracteres debe ser de 11 dígitos.
10. Cuando se selecciona un lote en la lista de lotes creados no se activa el botón Eliminar.

Las 10 no conformidades fueron resueltas. En la siguiente figura se muestra la solución posterior a la no conformidad 4.

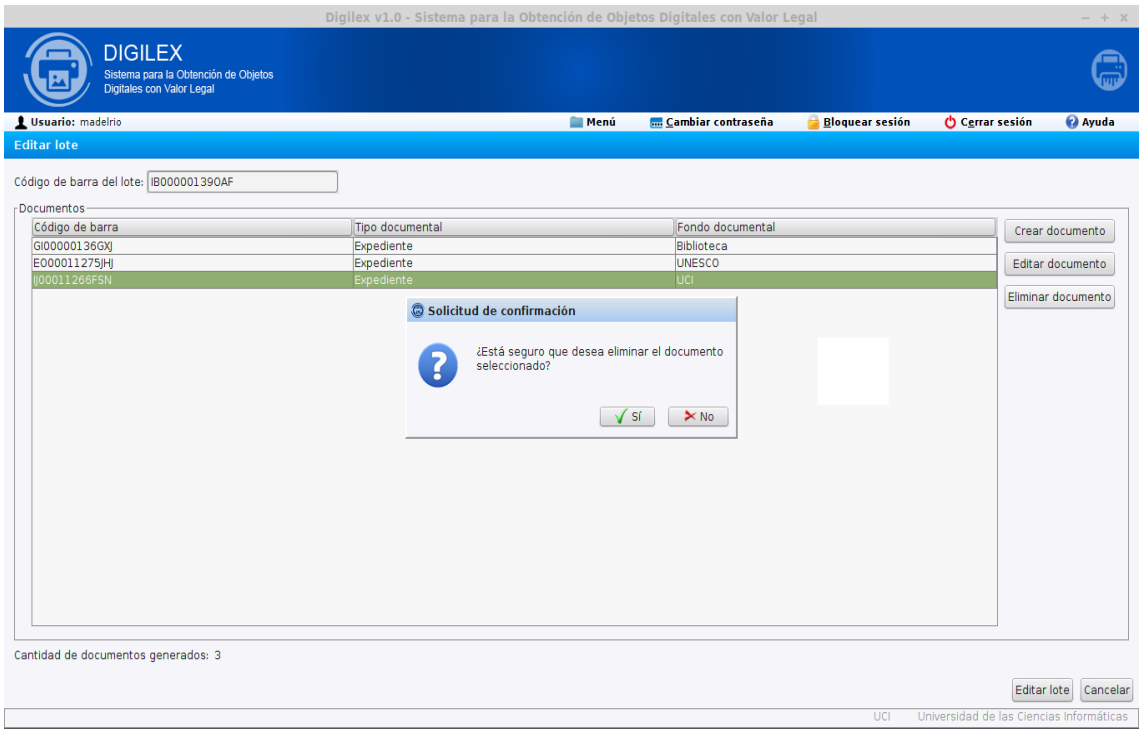



**Figura 22: Interfaz gráfica obtenida en la solución de la NC 4.**

Las 3 no conformidades obtenidas en la segunda iteración fueron:

1. No se muestra la ruta donde se guarda el pdf generado con los códigos de barras.
2. El mensaje que se muestra cuando se desea eliminar un lote la palabra "Esta" no tiene tilde.
3. En la interfaz de la lista de documentos de un lote, el contador **Cantidad de documentos generados** no se actualiza.

Las 3 no conformidades fueron resueltas. La siguiente figura muestra la solución posterior a la no conformidad 1.

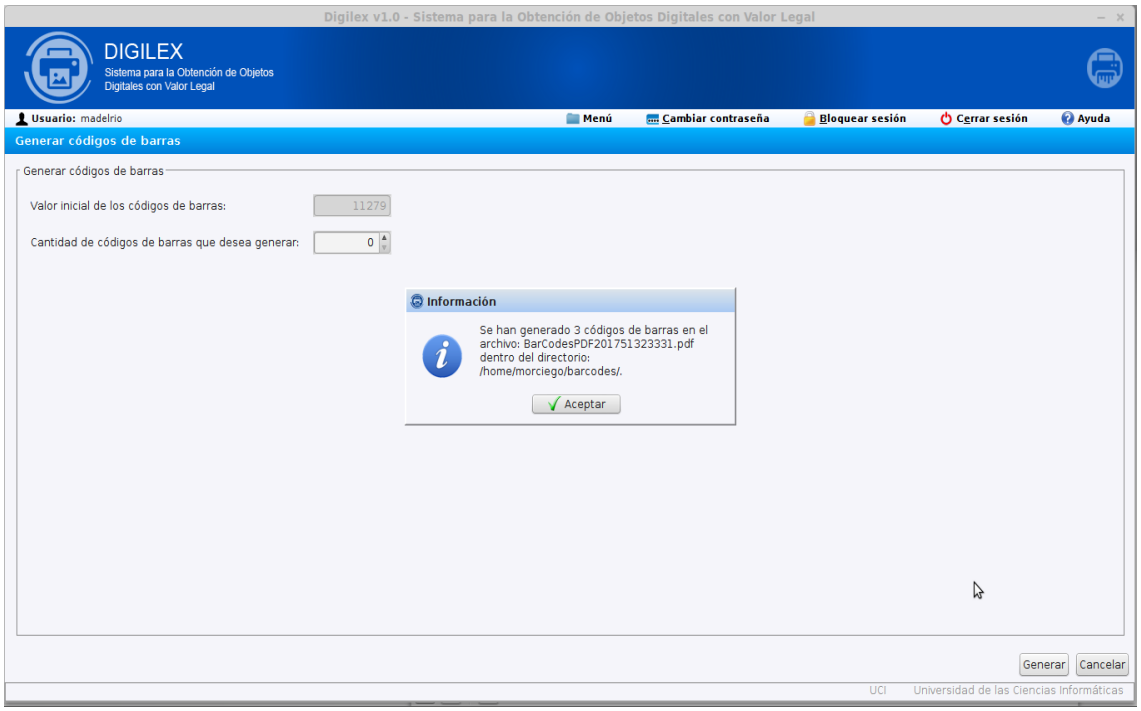



**Figura 23: Interfaz gráfica obtenida en la solución de la NC 1.**

En la tercera iteración no se encontró no conformidad permitiendo al departamento de calidad del centro emitir el documento correspondiente para la liberación del módulo.

## 3.3 Validación

La validación es un conjunto diferente de tareas que aseguran que el software que se construye siguiendo los requerimientos del cliente (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

### 3.3.1 Prueba de aceptación

La prueba de aceptación es generalmente desarrollada y ejecutada por el cliente o un especialista de la aplicación el cual le realiza una serie de pruebas específicas con la intención de descubrir errores antes de aceptar el software del desarrollador siendo conducida a determinar cómo el sistema satisface sus criterios de aceptación, validando los requisitos que han sido levantados para el desarrollo incluyendo la documentación y procesos de negocio. Está considerada como la fase final del proceso, para crear un producto confiable y apropiado para su uso (Pressman R. S., Ingenería del software: Un enfoque práctico, 2002).

**Tabla 10: Caso de prueba de aceptación Crear lote.**

| **Código de caso de prueba: 1** | **Nombre historia de usuario:** Crear Lote |
|---|---|
| **Nombre de la persona que realiza el caso de prueba:** Marcelo del Rio Rodríguez | |

| | | |
|---|---|---|
| **Descripción de la prueba:** El usuario puede crear un lote en el sistema. | | |
| **Condiciones de ejecución:** El usuario debe estar autenticado en el sistema. | | |
| **Entrada/Pasos de ejecución** | | **Resultados esperados** |
| **Acción** | **Entrada** | |
| El usuario debe seleccionar el botón **Crear lote**. El sistema muestra una interfaz, donde se mostrará el campo a llenar. El usuario escribe los datos del lote, una vez escritos los datos debe seleccionar el botón Crear lote. | Código de barra | Se adiciona un nuevo lote en el listado de los mismos. |
| **Evaluación de prueba: Satisfactoria** | | |

Con el objetivo de revisar la herramienta, teniendo en cuenta el CP definido, se realizó un encuentro con la jefa de proyecto de DIGILEX Ing. Anidey Machado Escudero en el cual se obtuvieron resultados satisfactorios y el software fue aceptado por el cliente.

## 3.4 Validación de las variables de la investigación

"Con la validez se determina la revisión de la presentación del contenido" (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2004).

Se realiza un cuasiexperimento como diseño para validar las variables de la investigación de forma cualitativa ya que este manipula deliberadamente, al menos, una variable independiente para observar su efecto y relación con una o más variables dependientes (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2004).

Pasos principales para aplicar el cuasiexperimento según (Hernández Sampieri, Fernández Collado, & Baptista Lucio, 2004):

1. Decidir cuantas variables dependientes e independientes se incluyen en el cuasiexperimento.
2. Desarrollar el instrumento o instrumentos para la recolección de datos que permita medir la(s) variable(s) dependiente(s).
3. Seleccionar una muestra de personas para el experimento.

Siguiendo los pasos anteriormente descrito se obtiene:

**Variable independiente:** desarrollo del módulo de gestión de los fondos documentales.

**Variable dependiente:** facilitar la gestión de los mismos para el proceso de digitalización de dicho sistema.

Como **instrumento para la recolección de datos** se utilizan cuestionarios. Un cuestionario consiste en un conjunto de preguntas con respecto a una o más variables a medir estas. Aplicando preguntas cerradas las cuales contienen categorías o alternativas de respuesta que han sido delimitadas, es decir, se presentan a los sujetos las posibilidades de respuesta y aquellos deben circunscribirse a éstas.

Empleando preguntas dicotómicas (dos alternativas de respuestas) a 4 especialistas de la aplicación perteneciente al centro CEGEL se obtienen los siguientes resultados.

¿Son organizados y clasificados los documentos para digitalizar?

Especialista #1- Si

Especialista #2- Si

Especialista #3- Si

Especialista #4- Si

¿El mecanismo posibilita la entrada y gestión de fondos documentales al sistema?

Especialista #1- Si

Especialista #2- Si

Especialista #3- Si

Especialista #4- Si

¿Los documentos digitalizados poseen una ubicación con respecto a los documentos físicos?

Especialista #1- Si

Especialista #2- Si

Especialista #3- Si

Especialista #4- Si

Por tanto se puede afirmar que el módulo gestión de fondos documentales para su digitalización del sistema para la obtención de objetos digitales con valor legal (DIGILEX)

posee una calidad aceptada por lo que se puede concluir que se cumplió lo planteado como idea a defender validándose así las variables de la investigación

## 3.5 Conclusiones del capítulo

En el capítulo recién concluido se puede afirmar aplicando las pruebas unitarias y funcionales, utilizando el método de prueba de caja blanca y de caja negra respectivamente, se obtuvo que no existe código innecesario además se elaboraron y aplicaron los casos de pruebas de aceptación a cada historia de usuario para dar validez y veracidad a la propuesta de solución. Además, se validaron las variables de la investigación científica cumpliendo con lo ideado. Mediante lo anteriormente citado, se arriba a la obtención de un prototipo funcional, en el cuál se evidencia un correcto funcionamiento de los requisitos planteados para darle solución al módulo del sistema.

## Conclusiones Generales

Al concluir el presente trabajo se puede afirmar que:

1. El estudio de diferentes conceptos relacionados con la gestión documental, el análisis de los sistemas existentes en el ámbito nacional e internacional y la definición de las tecnologías y herramientas definidas por el proyecto permitió elaborar el marco teórico de la investigación para fundamentar las bases de la solución.
2. La obtención de las historias de usuario definidas por los requerimientos del cliente, el análisis y diseño de sistema mediante patrones, estándares de codificación y diseño de la base de datos permitió desarrollar el módulo de gestión de un fondo documental para el sistema DIGILEX.
3. Mediante los resultados satisfactorios en cuanto a la alta reutilización de las clases empleando las métricas de diseño y el análisis a los casos de pruebas obtenidos aplicando diferentes métodos de pruebas se validó la solución propuesta.

## Recomendaciones

A partir de la investigación realizada se recomienda:

- Trabajar en nuevas versiones del sistema que permitan la mejora continua del mismo a partir de nuevos requerimientos del cliente.

## Bibliografía


ALBERT, S. (2010). *Manual de apoyo para recepción y devolución de documentos del centro de Digitalización.*

Amador, L. J. (2013). *Estudio comparativo de sistemas de mapeo objeto relacional desarrollados en plataformas Open Source." .* CITECSA .

Asociación Española para la Calidad. (2017). *QAEC Asociación Española para la Calidad*.

Borges Piedra, L. .. (2011). *Diseño e implementación de los módulos de Recepción y Devolución, Control de Calidad y Otorgamiento de Documentos del Centro de Digitalización para la División de Antecedentes Penales.* La Habana.

Borges, L. P., Lizama, Y. M., & Hernández, A. T. (2011). *Diseño e implementación de los módulos de Recepción y Devolución, Control de Calidad y Otorgamiento de Documentos del Centro de Digitalización para la División de Antecedentes Penales.* La Habana.

Calleja, M. A. (2009). *Carmen. Estándares de codificación.*

Calleja, M. A. (2009). *Carmen. Estándares de codificación.*

Canós, J. H. (2003). *Metodologías ágiles en el desarrollo de software. Metodologías Ágiles en el Desarrollo de Software,.*

DIBAM. (2017, enero 22). *Archivo Nacional de Chile*. Retrieved from http://www.archivonacional.cl/616/w3-article-10983.html

DocPath Corp. (2017). *DocPath® Aspen™ Sistema de gestión documental con funcionalidades avanzadas.*

DocPath Corp. (2017). *DocPath® Aspen™ Sistema de gestión documental con funcionalidades avanzadas.*

documental, e. m. (2012, 12 05). *clasificacion-y-ordenacion-de-los-documentos*. Retrieved from https://elmundodocumental.wordpress.com/2012/12/05/clasificacion-y-ordenacion-de-los-documentos/

Duarte, P. M. (2015). *Evaluación de la accesibilidad en software generado por un entorno de desarrollo integrado.* Retrieved from APA

Echeverry Tobón, L. M. (2007). *Caso práctico de la metodología ágil XP al desarrollo de software* . Doctoral dissertation, Universidad Tecnológica de Pereira.

El pensante ©. (2017). *el pensante*. Retrieved from el pensante: https://educacion.elpensante.com/el-valor-legal/

El pensante ©. (2017). *el pensante*. Retrieved from el pensante: https://educacion.elpensante.com/definicion-de-tipologia-documental/

Fernández Valdés, M. D. (2008). *Análisis conceptual de las principales interacciones entre la gestión de información, la gestión documental y la gestión del conocimiento.* Acimed.

Fernandez, O. (2005). *Introducción al lenguaje de programación Java-Una guía básica. .* España: Universidad Carlos III de Madrid.

Figueroa, R. G., Solís, C. J., & Cabrera, A. A. (2008). *Metodologías tradicionales vs. Metodologías ágiles.* Universidad Técnica Particular de Loja, Escuela de Ciencias en Computación.

General technology Corporation. (2017). *General technology*. Retrieved from General technology: http://www.arnetpr.com/argtcorp/programacion/dds/DDS.pps

Gimeno, J. M. ( 2011). *Introducción a Netbeans.*

González Valdés, S. .. (2011). *Diseño e implementación de los módulos de Preparación de Documentos, Digitalización de Documentos y Asociación de Metadatos del Centro de Digitalización para la División de Antecedentes Penales.*

González, S. V., Suárez , R. F., & Lizama , Y. M. (2011). *Diseño e implementación de los módulos de Preparación de Documentos, Digitalización de Documentos y Asociación de Metadatos del Centro de Digitalización para la División de Antecedentes Penales.*

Guibert Estrada, L. y. (2011). *Ingeniería de requisitos del software educativo "Mis Mejores Cuentos".* La Habana, Cuba.

Hernández Aguilar, V. F. (2010). *Protofase a la ingenieria de requisitos para facilitar la comprensión del negocio a informatizar en el desarrollo de software de gestión.* Perú. Retrieved from Protofase a la

Hernández Sampieri, R., Fernández Collado, C., & Baptista Lucio, P. (2004). *METODOLOGÍA DE LA INVESTIGACIÓN.* Mexico.

Julián Pérez Porto, M. M. (2009). *Definición.de*. Retrieved from Definición.de: http://definicion.de/documento/

La Rosa Montes.Dayana, R. A. (2007). *Análisis y Modelado del sistema para el proceso de digitalización de documentos de Registros y Notarias del MPPRIJ de la República Bolivariana de Venezuela.* La Habana.

La Rosa Montes.Dayana, R. A. (2007). *Análisis y Modelado del sistema para el proceso de digitalización de documentos de Registros y Notarias del MPPRIJ de la República Bolivariana de Venezuela.* La Habana.

Larman, C. (1999). *UML y patrones.*

Larman, C. (2003). *UML y Patrones Segunda edición,.* España: Pearson Prentice Hall.

Lenis Matos Rodriguez, D. S. (2012). *Propuesta de un mecanismo de seguridad para el marco de trabajo Kairos. .*

Lezcano., W. M. (2009). *Descripción de la arquitectura de software de la herramienta.* La Habana.

LinkedIn Corporation © . (2017). *slideshare*. Retrieved from slideshare: https://es.slideshare.net/camaldonado28/digitalizacin-de-documentos

M Ortiz, A. P. ( 2014 ). *Programación orientada a objetos con Java y UML.* Editorial Universitaria Abya-Yala. Retrieved from https://msdn.microsoft.com/es-es/library/xk24xdbe(VS.80).aspx

Microsoft | Achitecture. (2010). *Guía de Arquitectura N-Capas orientada al Dominio con .NET 4.0.*

Microsoft. (2005). *Microsoft*. Retrieved from https://msdn.microsoft.com/es-es/library/xk24xdbe(VS.80).aspx

Momjian, B. (2001). *PostgreSQL: i. ntroduction and concepts (Vol. 192).* New York: Addison-Wesley.

Pressman, R. S. ( 2010.). *Ingeniería del Software.*

Pressman, R. S. (1988). *Ingeniería del software.*

Pressman, R. S. (2002). *Ingenería del software: Un enfoque práctico.*

Pressman, R. S. (2002). *Ingenería del software: Un enfoque práctico.*

Ramírez, T. (1999). *Proyecto de Investigación.* Caracas: Panapo.

ROJAS, M. J. (2010). *Patrones de Diseño.*

Sánchez, T. R. (2015). *Metodología de desarrollo para la Actividad productiva de la UCI.*

Sistema de identificación, migración y extranjería. (2011). *Sistema para la Digitalización de Alfabética.*

SOFTONIC INTERNACIONAL S.A. (2017). *softonic*. Retrieved from https://easyscan.softonic.com/

SOFTONIC INTERNACIONAL S.A. (n.d.). *softonic*. Retrieved from https://easyscan.softonic.com/

Sommerville. (2005). *Ingenería del software.* Madrid: Pearson.

Varela, V. (n.d.). *Metodologia de desarrollo de sofware.*