República de Cuba

Universidad de las Ciencias Informáticas

Facultad 2

"Sistema de Gestión de la Información para las Ópticas de Cuba"

Trabajo de Diploma

Presentado para optar por el título de Ingeniero en Ciencias Informáticas

Autores: Orquidia Ferrer Rodríguez

Rayner Bestard Díaz

Tutora: Ing. Anabel Betancourt de los Santos

Co Tutores:

Ing. Mirbel Agramonte Rojas

Ing. Ráiner Cárdenas Alvarez

"Año 55 de la Revolución"

La Habana, 2013.

Declaración de Autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Facultad 2 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los ____ días del mes de _____ del año 2013.

_____

Autor: Orquidia Ferrer Rodríguez

_____

Autor: Rayner Bestard Díaz

_____

Tutora: Ing. Anabel Betancourt de los Santos

_____

Co-Tutor: Ing. Ráiner Cárdenas Alvarez

_____

Co-Tutora: Ing. Mirbel Agramonte Rojas

Resumen

Los Sistemas de Gestión, en estos tiempos, prometen un impulso en las estrategias encaminadas a lograr la optimización de los procesos de negocio. Permitir un desarrollo perdurable en la gestión de políticas, procedimientos y procesos es la principal meta de estos sistemas.

El presente trabajo está encaminado a la solución de los problemas existentes en los procesos de gestión de información que tributen al adecuado funcionamiento de las ópticas de Cuba, a partir de un sistema que permita tener control de la información relacionada con las solicitudes, recursos y los reportes que se generan dentro de las ópticas. Para alcanzar este objetivo se plantean los principales conceptos que fundamenten la investigación, además se realiza una revisión de los antecedentes sobre la gestión de la información de las ópticas en el ámbito nacional e internacional.

Se expone además la fundamentación de las herramientas utilizadas y los elementos necesarios para la construcción del Sistema de gestión de la Información para las ópticas de Cuba la cual posee un conjunto de funcionalidades que contribuyen a la preservación y el control de la información de las ópticas en Cuba.

Palabras Claves: Control de la Información, Ópticas, Sistemas de Gestión.

Índice

Índice de Figuras

Índice de Tabla

Introducción

El Sistema de Salud Cubano encierra en sí todos los procesos que apoyan la ampliación de planes, programas y acciones que garantizan el logro de los objetivos enfocados a preservar la salud de nuestro pueblo. A dicho sistema pertenecen las ópticas, cuya misión fundamental es brindar un servicio especializado a la población en la venta de espejuelos, lentes de contacto, entre otros. Sin embargo el proceso que se realiza dentro de la óptica es más que la venta de espejuelos, allí se realizan la reparación de los mismos, así como la venta de artículos útiles como armaduras, estuches, franelas, cordones, galenos (espejuelos que vienen graduados desde la fábrica), entre otros. Se llevan a cabo también la rectificación de la medición de la vista, se realizan cortes de cristal y en algunas ópticas se preparan a los estudiantes en la tecnología optometrista.

La inmensa cantidad de órdenes que llegan diariamente a cada una de las ópticas, la existencia de materiales de trabajo a veces obsoletos y poco eficientes, y el faltante de materiales hacen que el retraso en la entrega del producto a los clientes sea el principal motivo de molestia para el mismo pues debe visitar una y otra vez la óptica hasta recibir el servicio completo.

Hoy alrededor del país existen una gran cantidad de ópticas donde utilizan diferentes materiales de trabajo; implicando que el cliente deba visitar varias en caso de no encontrar el material deseado en la óptica más cercana. Solamente existe un directorio en la página oficial de Infomed[1], en la cual se brindan datos de las ópticas y su dirección, la misma carece de un contacto directo a través del cual el cliente pueda constatar si el producto que necesita está en alguna de ellas. Actualmente el principal problema en las ópticas es la atención a las órdenes de graduaciones muy altas y aunque existen alternativas para esta problemática, lleva adjunta una demora o un viaje largo para el usuario.

Además toda la información se genera de forma manual y debido a esto la información existente se halla dispersa, haciendo que su manejo y/o procesamiento pueda resultar difícil y en algunos casos, puede llevar consigo a su extravío y/o destrucción, al no conocer la importancia que tienen para realizar estudios, auditorías, entre otros.

[1] Centro Nacional de Información de Ciencias Médicas, Ministerio de Salud Pública. Directorio de Instituciones. Biblioteca Virtual de Salud.

Dada la situación anterior se plantea como **problema a resolver:** ¿Cómo contribuir a la preservación y control de la información de las ópticas en Cuba?

Basado en lo anterior, se define como **objeto de estudio:** Procesos de gestión de información en las ópticas.

Para dar solución al problema se plantea como **objetivo general:** Desarrollar una aplicación que permita gestionar la información relacionada con los procesos de solicitud de servicios a la población en las ópticas cubanas.

Del objetivo general planteado se derivan los siguientes **objetivos específicos:**

1. Elaborar el marco teórico de la investigación identificando las principales tendencias, limitaciones y ventajas de software existentes para gestionar la información relacionada con las ópticas.

2. Definir y modelar los procesos fundamentales relacionados con la gestión de información de los recursos, solicitudes y reportes que se realizan en la óptica.

3. Obtener los requisitos funcionales y no funcionales del sistema.

4. Realizar el diseño del sistema.

5. Desarrollar un sistema que facilite la gestión de la información de las Ópticas de Cuba permitiendo un mejor control y preservación de la misma.

6. Validar el sistema desarrollado a partir de la realización de pruebas caja negra y de caja blanca.

Se define como **campo de acción**: Procesos de gestión de información relacionada con los recursos y las solicitudes que se manejan en las ópticas de Cuba.

Las **tareas de investigación** planificadas para dar cumplimiento a los objetivos planteados son:

1. Revisión de los antecedentes sobre la gestión de la información de recursos y solicitudes teniendo a las ópticas como entidad principal.

2. Caracterización de la situación existente en el mundo, en la región y en el país en particular sobre gestión de la información de recursos y solicitudes en las ópticas para definir la posición del investigador.
3. Análisis y selección de las herramientas, tecnologías, lenguajes y metodologías a utilizar para el diseño y construcción del sistema informático.
4. Análisis de la arquitectura del software.
5. Diseño del modelo de datos.
6. Elaboración de los diagramas de clases del diseño.
7. Implementación del sistema.
8. Análisis de los estándares de codificación y calidad definidos en la UCI a ser utilizados para desarrollar la solución.
9. Definición de las pruebas a emplear para validar el sistema.

Con el objetivo de realizar la investigación fue necesaria la utilización de los siguientes métodos científicos:

Métodos Teóricos:

Analítico-Sintético: permite el estudio y revisión de la bibliografía necesaria para tener un mejor entendimiento del problema a resolver.

Inductivo-Deductivo: permite llegar al planteamiento del objetivo, además de la extracción de las ideas fundamentales para la elaboración y fundamentación teórica del trabajo de diploma.

Histórico-Lógico: permite aplicar la lógica y determinar lo más factible para el desarrollo del sistema, a través de un estudio de las tendencias actuales en el desarrollo de sistemas de gestión de información.

Métodos Empíricos:

Observación: permite obtener las características de los procesos que tienen lugar en las ópticas, capturar los requisitos del sistema, además de la información necesaria para el desarrollo del mismo.

Entrevista: permite conocer las necesidades del cliente y obtener los requisitos del sistema a tener en cuenta para la solución de la investigación.

El documento consta de cinco capítulos, a continuación se brinda un resumen de los contenidos que abordan los mismos.

Capítulo 1: Fundamentación Teórica

En el presente capítulo se abordan los conceptos y definiciones fundamentales que son utilizados, se realiza un estudio de las tendencias actuales de los sistemas de gestión de la información para ópticas investigados y se analiza la plataforma de desarrollo sobre la cual se diseña e implementa el sistema.

Capítulo 2: Características del Sistema

Se describen las características del sistema a desarrollar, así como el objeto de automatización. Se realiza una descripción detallada de la propuesta de solución, el modelo de negocio, se especifican los requisitos funcionales y no funcionales del sistema y las descripciones de los casos de uso.

Capítulo 3: Diseño del Sistema

Aborda las actividades definidas para el diseño del sistema siguiendo la metodología RUP, se elaboran los diagramas de clases del diseño teniendo presente los patrones a utilizar y se especifica la arquitectura del sistema.

Capítulo 4: Implementación y Prueba del Sistema

En este capítulo se definen los diagramas de componentes y despliegue. Además se presentan los resultados de las diferentes pruebas realizadas al sistema implementado, para verificar que cumple con los requisitos y así comprobar sus funcionalidades.

Capítulo 5: Estudio de la Factibilidad

En este capítulo se realizará un análisis de la factibilidad del sistema para determinar si la solución propuesta es económica y viable para el desarrollo**.**

Capítulo I: Fundamentación Teórica

1.1 Introducción

Durante el desarrollo de este capítulo se expone el marco teórico en el que se desarrolla la investigación, creando una base sólida para los próximos capítulos. Para ello, se realiza un estudio de las tendencias actuales de algunas soluciones informáticas que automatizan la gestión de los procesos de las ópticas en el mundo, lo que sirve como entrada al análisis de los procesos que se desean automatizar. Se realiza también un estudio de la plataforma seleccionada donde se especifican las principales características de la metodología, los lenguajes y las herramientas que se utilizan en el desarrollo de la solución.

1.2 Conceptos fundamentales

Óptica

La Real Academia Española asocia el vocablo óptica como: *"Establecimiento donde se comercia con aparatos compuestos de lentes y espejos, que sirven para ver estampas y dibujos agrandados y como de bulto."* (1)

Información

"Comunicación o adquisición de conocimientos que permiten ampliar o precisar los que se poseen sobre una materia determinada." (2)

Gestión de Información

"La gestión de la información es un proceso que incluye operaciones como extracción, manipulación, tratamiento, depuración, conservación, acceso y/o colaboración de la información adquirida por una organización a través de diferentes fuentes y que gestiona el acceso y los derechos de los usuarios sobre la misma." (3)

La gestión de la información se establece como una disciplina transversal que aparece entrelazada en todas las diferentes capas de una organización, en todos los conceptos de administración como recursos humanos, marketing, finanzas, estrategia, operaciones, entre otros. (3)

Un sistema de gestión ayuda a lograr los objetivos de una organización mediante una serie de estrategias que incluyen la optimización de procesos, el enfoque centrado en la gestión de la información, entre otros. (4)

La implementación de un sistema de gestión brinda ventajas como:

- Control efectivo de las actividades de la organización.
- Integración de nuevas tecnologías y herramientas.
- Disponibilidad de mayor y mejor información para los usuarios en tiempo real.
- Elimina la barrera de la distancia trabajando con un mismo sistema en puntos distantes.
- Disminuye errores, tiempo y recursos.

Pero también presentan desventajas como:

- El tiempo que pueda tomar su implementación.
- La resistencia al cambio de los usuarios.
- Los problemas técnicos como fallas de hardware y/o software.
- La inadecuada implementación de las funcionalidades para el apoyo de actividades de la organización.

Los sistemas de gestión de información permiten la comprensión de cada organización desde un enfoque analítico, evaluador y creativo que permite develar las oportunidades que merezcan ser explotadas y contrarrestar las amenazas, además de establecer los factores que resulten críticos y buscar nuevas oportunidades. (5)

Por todo lo planteado, se hace necesario el desarrollo de un sistema capaz de facilitar los procesos de gestión de la información, por lo que con el objetivo de identificar las experiencias efectivas de las empresas que estén inmersas en el desarrollo de soluciones informáticas y así obtener una solución al problema existente, es necesario realizar un estudio sobre las tendencias actuales de los sistemas de gestión de información.

1.3 Tendencias actuales hacia los sistemas de gestión de información

La información es un bien económico que cuenta con dos características únicas: no merma al ser consumida, ni se pierde al ser trasmitida. A pesar de ello, su valor

económico ha ido ganando trascendencia respecto a la mano de obra y al capital, hasta llegar a ser un recurso estratégico en la mayoría de las organizaciones. (6)

1.3.1 Sistemas de gestión de información para ópticas

En la presente investigación fue llevado a cabo el estudio de varios sistemas de gestión de la información para ópticas como:

OCUCO

Es un sistema de gestión para óptica que ha desarrollado dos amplias gamas en productos de software en el mercado para consultas ópticas, oftalmológicas y optometristas, denominadas Acuitas Enterprise y Acuitas Optical. Los procesos de gestión de la información que manipulan son similares a los del sistema a desarrollar pero este software tiene un costo muy elevado ya que para su desarrollo fue invertido 4,5 millones de dólares y más de 14 millones en facturación hasta el año 2011, por lo que su obtención se torna difícil debido a su alto valor económico. (7)

OPTIVEN

Es un sistema para el control automatizado de los procesos de ventas, facturación, caja, exámenes optométricos u oftalmológicos por pacientes, órdenes de trabajo, inventario, entre otros. Ha sido diseñado como una herramienta amigable, de fácil manejo y mejora el desenvolvimiento de las labores de ventas y administración dentro de una óptica

Es el encargado de controlar de manera automática y ordenada las ventas, la facturación, la caja, el inventario de mercancía, el registro de datos personales y optométricos de los pacientes, la seguridad del sistema, los reportes, las compras, entre otros. (8)

Es una aplicación de escritorio, cuya principal desventaja radica en que sólo responde a las especificaciones y reglas del negocio para las ópticas venezolanas, ya que los procesos que se gestionan son diferentes a los propios de las ópticas cubanas. Para utilizarlo sería necesario realizar una personalización a dicho sistema y así adecuarlo a los procesos de Cuba.

OPTISOF

Es un sistema de gestión creado con la finalidad de dar soporte en la administración y control de ópticas, incluye graduaciones, cristales y monturas vendidas, todo en un vistazo de manera clara e intuitiva. Es un paquete integral de manejo de administración para ópticas con el que es posible llevar un extenso chequeo de todas las necesidades del negocio y facilita el control absoluto de todas las gestiones que se realicen en el establecimiento. (9)

Optisof contiene los siguientes módulos: (10)

- ***Módulo de Ventas:*** este incluye un listado de caja, ventas totales, por períodos, días, por producto, por vendedor, por proveedor y garantías. Permite la selección de las formas de pago, sea pagos aplazados, devoluciones y generación de tickets regalo.
- ***Módulo de Pacientes/Clientes:*** contiene la información del servicio de Optometría, audiometría, historias de pacientes, diagnósticos, agenda de citas y visitas de clientes.
- ***Módulo de Almacén:*** permite el control de inventario y existencias con rapidez. Posibilidad de generación e impresión de códigos de barras, aviso de existencias mínimas de materiales, con parámetros preestablecidos y la división del producto en stock por lentillas, monturas, lentes, entre otros.
- ***Módulo de compras:*** incluye funcionalidades para generar comprobantes de compra o pedido a proveedores, alta de entradas de artículos y el seguimiento de compras, por facturas, vencimientos, pagos e impagos.

Su desventaja fundamental radica en que es privativo y su costo asciende los 1.246,51 Euros por cada versión que se obtenga del sistema.

PYME MANAGER OPTICAS

Es un software especializado en la gestión, control y la administración dentro de las ópticas, desarrollado por la compañía Imagine Software, resulta muy sencillo de utilizar para lo cual no es necesario tener conocimientos avanzados sobre informática, lo cual lo hace recomendable para todo tipo de usuario.

Dispone de las siguientes opciones que permiten realizar múltiples acciones relacionadas con los centros Ópticos como el envío de mensajes de texto SMS de forma masiva integrado, el control y gestión de horarios y citas para revisiones, una base de datos

organizada mediante fichas sobre los clientes que incorpora toda la información necesaria para el control de cada caso, fichas de optometría y oftalmología, facturación rápida, caja del día y control de gastos. (11)

Es una solución de primer nivel completamente gratuita que puede reemplazar otras soluciones extremadamente caras que existen en el mercado de las ópticas. La mayor limitante radica en que es una aplicación de escritorio que sólo puede ser ejecutada sobre el sistema operativo Windows 7, la cual tiene una demanda muy alta de requisitos mínimos de hardware, tanto en el servidor como en las estaciones de trabajo, lo cual puede resultar un problema ya que las ópticas cubanas no cuentan en su totalidad con la infraestructura necesaria y en caso de tenerla puede que no llegue a tener los requisitos mínimos de hardware, por lo tanto es necesario desarrollar una solución que se ejecute con prestaciones bajas y que sea Web porque en caso de necesitar recursos sólo se necesitarían para el servidor.

Tras el estudio de las soluciones informáticas expuestas, se arroja como resultado que aunque todas llevan a cabo la informatización de los procesos de gestión de información para las ópticas, similares a los que se llevan a cabo en nuestro país, no pueden ser reutilizados ya que no se tiene acceso al código fuente de dichos sistemas por lo que la posibilidad de la reutilización queda descartada y no cuentan con las funcionalidades que se requieren para la gestión de información de las ópticas en Cuba, como la generación de reportes para el control de los datos, la visualización de las gráficas estadísticas, entre otros.

1.4 Metodología de desarrollo de software

Una metodología es definida como el *"conjunto de métodos que se siguen en una investigación científica o en una exposición doctrinal".* (12) *D*entro de un proyecto es la que define Quién debe hacer Qué, Cómo y Cuándo hacerlo. Decidir cuál metodología usar es de vital importancia pues es la encargada de guiar todo el proceso de desarrollo del software.

En la actualidad existen dos clasificaciones en lo que se refiere a procesos de desarrollo, los llamados métodos robustos o rígidos, y los métodos ágiles o ligeros. El primer método se encuentra centrado especialmente en el control del proceso, estableciendo

rigurosamente las actividades involucradas, los artefactos que se deben generar, así como las herramientas y notaciones que se usarán generando una abundante documentación que servirá a futuras versiones que contribuyan a lograr mejoras en el software. (13) El segundo método centra su atención en la integración del cliente al equipo de desarrollo, así como en la generación de códigos en un plazo a corto tiempo y establecer pequeños equipos de desarrollo preferentemente en parejas.

El Proceso Unificado de Rational conocido como RUP es la metodología seleccionada como guía a seguir durante el proceso de desarrollo del sistema.

1.4.1 Proceso Unificado de Rational (Rational Unified Process, RUP)

Los autores de RUP destacan que el proceso de desarrollo de software define tres características fundamentales como: dirigido por casos de uso, centrado en la arquitectura, e iterativo e incremental. (13)

- **Dirigidos por Casos de Usos:** los casos de uso son una técnica de captura y descripción de los requisitos funcionales del sistema desde la perspectiva del usuario, se usan como una guía para el diseño, implementación y prueba, estos constituyen un elemento integrador.
- **Centrado en la arquitectura:** aquí se describen los componentes del modelo, que son de vital importancia para la construcción del sistema, además se establecen las bases del mismo que son necesarias para entenderlo y desarrollarlo, lo que hace posible la reutilización a gran escala y provee una base para la gestión del proyecto.
- **Iterativo e incremental:** RUP propone cuatro fases: inicio, elaboración, construcción y transición, y cada una de ellas se divide en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo en cada una de las fases. Con su culminación se obtiene un producto con un determinado nivel, que irá creciendo incrementalmente en cada iteración.

Como consecuencia de la poca comunicación que existe con el cliente, producto de que este se encuentra fuera del centro de desarrollo, es de suma importancia mantener un control del proceso de desarrollo y además generar una gran cantidad de documentación que esté al alcance del cliente en caso de que los desarrolladores de la aplicación no estén disponibles, producto de la sustitución del personal del equipo de trabajo, el

personal que le sigue debe tener la mayor cantidad de documentos, con el objetivo de mejorar, entender y dar mantenimiento a la aplicación.

RUP como metodología de desarrollo se integra con el Lenguaje Unificado de Modelado (UML) permitiendo realizar el análisis, documentación e implementación de sistemas orientados a objetos.

1.5 Lenguajes de modelado

1.5.1 Lenguaje Unificado de Modelado (Unified Modeling Language, UML)

UML es un lenguaje muy conocido y utilizado en el modelado de sistemas de software, permite visualizar, especificar, construir y documentar el sistema a desarrollar. Ofrece un estándar para detallar un plano del sistema (modelo), el cual incluye características conceptuales como procesos de negocio, funciones del sistema, aspectos concretos como expresiones de lenguaje de programación, esquemas de bases de datos, entre otros. (14)

Para la definición del modelado de negocio existen estándares como el IDEF0 (Integrated Definition Methods) y la Notación de Modelado de Proceso del Negocio (Business Process Modeling Notation, BPMN), esta última es la seleccionada para el modelado del negocio del sistema a desarrollar.

1.5.2 Notación de Modelado de Procesos del Negocio (Business Process Modeling Notation, BPMN)

Para realizar el modelado de negocio se pueden emplear técnicas y notaciones que ayudan a conocer los objetivos del negocio y modelarlos.

BPMN es el nuevo estándar para el modelado de procesos de negocio y servicios web, es además una notación a través de la cual se expresan los procesos de negocio en un Diagrama de Procesos de Negocio (BPD). Este estándar agrupa la planificación y gestión del flujo de trabajo, así como el modelado y la arquitectura. (15)

Entre sus principales características se encuentran: (15)

- Proporciona un lenguaje gráfico común, con el fin de facilitar su comprensión a los usuarios de negocios.
- Utiliza una Arquitectura Orientada por Servicios, con el objetivo de adaptarse rápidamente a los cambios, aumentando las oportunidades del negocio y optimizar los procesos que faciliten la innovación del mismo.

La notación BPMN está compuesta por eventos (entrada, intermedios, fin), actividades (tarea y subproceso), gateways (compuertas), objetos conectores, swimlanes (canales) y artefactos. (15)

1.6 Plataforma de desarrollo

"Una plataforma de desarrollo es el entorno de software en el cual se desenvuelve la programación de una aplicación". (14)

Para la realización de este sistema de gestión se propone la utilización de los siguientes elementos que conforman el entorno de desarrollo:

1.6.1 Plataforma Java

Una plataforma Java consiste en una máquina virtual (VM) y una interfaz de programación de aplicaciones (API). La máquina virtual de Java es un programa, para un hardware y plataforma de software en particular, que ejecute una aplicación Java. Una API es una colección de componentes de software que se puede utilizar para crear otros componentes de software o aplicaciones. Cada plataforma Java proporciona una máquina virtual y un conjunto de API, esto permite que las aplicaciones escritas para la plataforma puedan ejecutarse en cualquier sistema compatible con todas las ventajas del lenguaje de programación Java. (16)

La plataforma JavaEE (Java Enterprise Edition) está diseñada para reducir la complejidad en el desarrollo de aplicaciones, para proporcionar un modelo de desarrollo, un conjunto de API tales como: Java Server Faces Technology, Java Server Pages Technology, Java Server Pages Standard Tag Library, Java Database Connectivity (JDBC), entre otras y un entorno de ejecución que permite a los desarrolladores centrarse en sus funcionalidades. Además el servidor de aplicaciones puede manejar transacciones, seguridad,

escalabilidad y gestión de los componentes desplegados, permitiéndole a los desarrolladores enfocarse en la lógica del negocio en lugar de las tareas de mantenimiento de bajo nivel. (16)

1.6.2 Sistema Gestor de Base de Datos (SGBD)

PostgreSQL v9.1

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (ORDBMS) que ha sido desarrollado de varias formas desde 1977. Está ampliamente considerado como el sistema de bases de datos de código abierto más avanzado del mundo. Posee características que tradicionalmente sólo se podían ver en productos comerciales de alto calibre. (17)

Algunas de las características fundamentales de PostgreSQL son: (17)

- ***DBMS Objeto_Relacional:*** aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas, como las consultas SQL declarativas, control de concurrencia multi-versión, soporte multi-usuario, transacción, optimización de consultas, herencia, y arrays (arreglos).
- ***Altamente Extensible:*** soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.
- ***Soporte_SQL_Comprensivo:*** soporta la especificación de características avanzadas tales como las uniones (joins)
- ***Integridad Referencial:*** soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.
- ***Lenguajes Procedurales:*** tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL.

La principal importancia de PostgreSQL es su habilidad para usar diversos lenguajes de programación, ofrece además una potencia adicional sustancial al incorporar conceptos adicionales. (18)

1.6.3 Entorno de Desarrollo Integrado (IDE)

NetBeans v7.0

El Entorno de Desarrollo Integrado (Integrated Development Enviroment, IDE), NetBeans es un software multiplataforma, de código abierto (open-source), libre bajo la licencia GPL (General Public License) y gratuito sin restricciones de uso, hecho principalmente para el lenguaje de programación Java, aunque permite también el desarrollo en otros lenguajes de programación. Tiene un atractivo balance entre la interfaz con múltiples opciones y un aceptable completamiento de código. (19)

Posee una serie de módulos que el usuario puede descargar para necesidades específicas de desarrollo. Específicamente el módulo NetBeans IDE para Web & JavaEE provee una amplia gama de herramientas relacionadas con el desarrollo de aplicaciones Web entre las que se pueden mencionar el nuevo Java EE 6 Web Profile, y su integración con el Web Server Apache Tomcat. (20)

NetBeans fue seleccionado como IDE por ser libre y sin restricciones de uso, ya que facilita el trabajo de los programadores por medio de la detección de errores y de la depuración de código, además soporta gran cantidad de lenguajes de programación, entre ellos Groovy.

1.6.4 Framework de desarrollo

Grails v2.0.1

Grails es un framework libre para desarrollo web, de código abierto y que utiliza la JVM, está construido sobre cinco pilares fundamentales: (21)

1. Groovy (lenguaje de programación) para la creación de propiedades y métodos dinámicos en los objetos de la aplicación.
2. Spring (framework para el desarrollo de aplicaciones) para los flujos de trabajo e inyecciones de dependencias.
3. Hibernate (framework de persistencia) para el Mapeo objeto-relacional.
4. SiteMesh (framework de aplicación) que gestiona la creación de las vistas.
5. Ant como framework para gestionar la compilación de un proyecto.

La idea de Grails, es crear un marco de desarrollo que favorezca la productividad al crear aplicaciones web, integrando factores de configuración comunes a la mayoría de los escenarios siguiendo paradigmas tales como Convención sobre Configuración cuyo

objetivo fundamental es disminuir el número de decisiones que el desarrollador debe tomar, ganando en sencillez pero no pierde por ello la flexibilidad y No te Repitas, que su finalidad es promover la disminución de la duplicación. (22)

Algunas de las características fundamentales de Grails son:

- ***Productividad:*** posee tres características fundamentales que intentan incrementar su productividad en comparación con los marcos de trabajo Java tradicionales, es un marco de trabajo preparado para funcionar desde el primer momento, y con funcionalidades disponibles a través de métodos dinámicos.
- ***Persistencia:*** GORM (Grails Object Relational Mapping) es el medio a través el cual los datos del modelo de dominio se hacen persistentes utilizando métodos dinámicos cuyas funcionalidades son similares a los DAO (Data Access Object).

1.6.5 Lenguaje de programación

Groovy v2.0

Groovy es un lenguaje de programación con una sintaxis similar a Java que compila a código de bytes de Java y se ejecuta en la Máquina Virtual de Java (JVM), permitiendo su integración a la perfección con Java, haciendo el aprendizaje de los programadores más fácil. Como base para su confección se usaron lenguajes tales como Java, Python y otros. Sin embargo, su sintaxis es mucho más flexible y potente que Java. Uno de los puntos más fuertes sobre Groovy y Grails es que son nativos de la JVM. (23)

Dada la universalidad de Java hoy en día, sería demasiado pedir a los desarrolladores implementar todas sus infraestructuras basadas en API (Interfaz de Programación de aplicaciones), bibliotecas y marcos empezando desde el principio. Por esta razón, Groovy y Grails están destinados a ser un gran éxito en el mundo empresarial. (22)

Se seleccionó Groovy pues posee una gran flexibilidad, dinamismo e integración además de su sintaxis similar al lenguaje Java. Además este lenguaje está integrado al marco de trabajo seleccionado para el desarrollo de la aplicación.

1.6.6 Servidor Web Apache Tomcat

Tomcat v7.0.30

Tomcat es una implementación completamente funcional de los estándares de JSP (Java Server Pages) y Java Servlet, es el servidor Web más utilizado a la hora de trabajar con el lenguaje Java en aplicaciones web, puede especificarse también como el manejador de las peticiones de JSP recibidas por servidores Web populares, como el servidor Apache HTTP de la Fundación de software de Apache o el servidor Microsoft Internet Information Server (IIS). Tomcat puede utilizarse como un contenedor solitario (principalmente para desarrollo y depuración) o como plugin para un servidor web existente (en este caso está integrado como un plugin al framework Grails). (24)

1.6.7 Herramienta para gestionar reportes dinámicos

IReport v4.1.1

En el caso específico de NetBeans se puede utilizar la herramienta iReport, la misma es un constructor o diseñador de informes visual, poderoso, intuitivo y fácil de usar para JasperReport escrito en Java. (25)

Algunas de las principales características de iReport son: (25)

- Está escrito en JAVA al 100%, es código abierto (open-source) y gratuito.
- Maneja el 98% de las etiquetas de JasperReport.
- Permite diseñar con sus propias herramientas figuras geométricas como rectángulos, líneas, elipses, además de cartas y subreportes.
- Recopilador y exportador integrados.
- Tiene asistentes para generar los subreportes.
- Tiene asistentes para las plantillas.
- Facilidad de instalación.

Permite utilizar JasperReport para generar los reportes diseñados.

JasperReport

JasperReport es una herramienta informática para generar reportes, de código abierto y desarrollada en Java la cual puede crear archivos de diferentes formatos: PDF, Microsoft Excel, entre otros. Puede ser usado en aplicaciones que puedan ejecutar código Java, incluyendo JavaEE o aplicaciones Web. (26)

1.6.8 Herramienta CASE

Las Herramientas CASE (Computer Assisted Software Engineering) se definen como un "*conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un Software".* (27) La principal ventaja de la utilización de una herramienta CASE, es la mejora de la calidad de los sistemas realizados y, en segundo término, el aumento de la productividad. Para conseguir estos dos objetivos es conveniente contar con una organización y una metodología de trabajo, además de la propia herramienta.

Visual Paradigm Enterprise Edition v8.0

Visual Paradigm es una herramienta CASE profesional que de igual forma, soporta el ciclo de vida completo del desarrollo de software, así como la notación BPMN. Posee una serie de características entre las que incluye generación de código y, además, es capaz de generar diagramas de ingeniería directa e inversa desde el código y permite el control de versiones. Se han realizado varias ediciones de este producto entre las que se incluye la Enterprise Edition la cual soporta BPMN 2.0 para el modelado de los procesos del negocio. (28)

Presenta una interfaz de uso intuitiva y con muchas facilidades a la hora de modelar los diagramas, esta herramienta puede generar diagramas de UML entre los que se encuentran: diagrama de clases, de caso de uso, componentes, despliegue, entre otros.

1.7 Conclusiones parciales

En el presente capítulo se realizó un estudio del estado del arte de los procesos de gestión de información y las soluciones informáticas que actualmente se enfocan en esa área. El estudio arrojó como resultado que los sistemas estudiados no satisfacen las necesidades del sistema de salud cubano en el área de las ópticas en cuanto a la gestión de información, por lo cual se concluyó que la solución adecuada es el desarrollo de una aplicación web para la gestión de información de las ópticas en Cuba.

Se realizó una evaluación de la metodología de desarrollo y las herramientas a utilizar definiéndose para la realización de la aplicación propuesta como metodología de desarrollo de software a RUP y como lenguaje unificado de modelado UML, tomando en

consideración que la integración de ambas permite el análisis, documentación e implementación de sistemas orientado a objetos.

Una vez definida la plataforma seleccionada para el desarrollo, se procede a detallar las características del sistema.

Capítulo II: Características del Sistema

2.1 Introducción

En el presente capítulo se describe la propuesta de solución, representando los procesos del negocio, puntualizando en los que serán objeto de automatización. Además se enumeran los requisitos funcionales y no funcionales, así como una descripción de los casos de uso del sistema y los diagramas relacionados.

2.2 Procesos del negocio

El modelado de los procesos del negocio se realizará mediante la notación BPMN. Los procesos de negocio identificados se muestran en los siguientes diagramas:

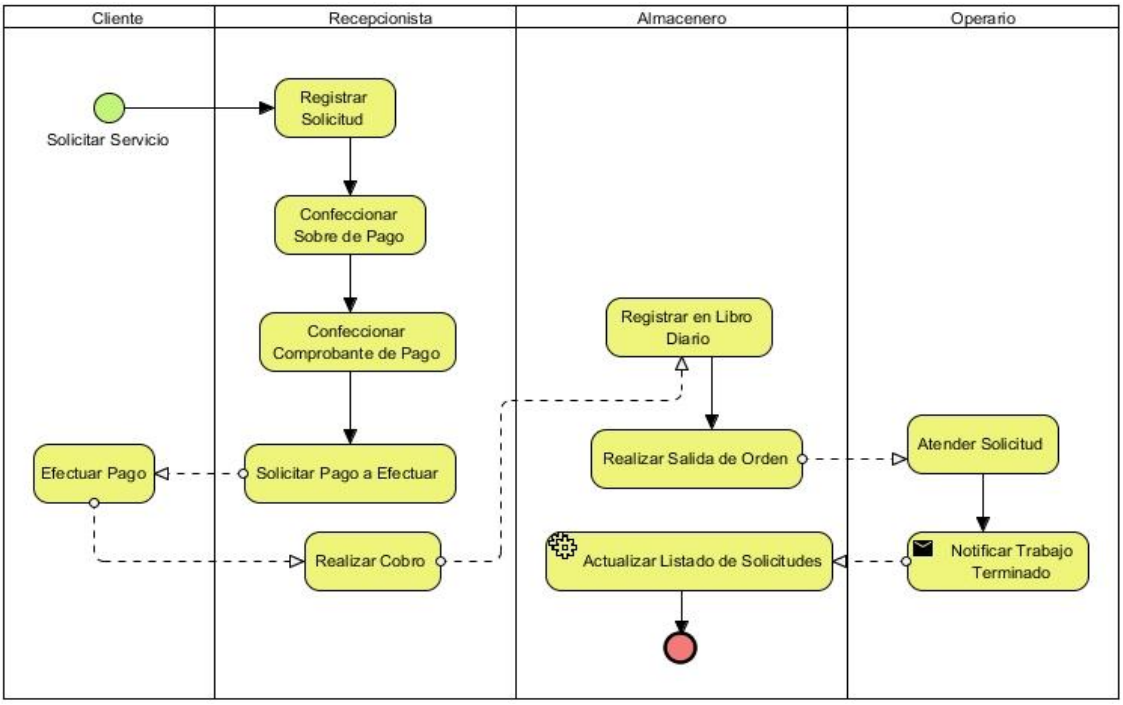

Figura: 2.1 Realizar Solicitud.

Realizar Solicitud: con la llegada de un cliente a la óptica es atendido por la recepcionista, a la cual debe entregarle una receta médica con los datos de la medición de la vista realizada en la consulta optometrista, luego solicita el servicio y/o producto, la recepcionista registra los datos de la solicitud y confecciona el sobre de pago donde es especificado el servicio o producto a consumir, en dependencia del tipo que sea, procede

a confeccionar un comprobante de pago para el cliente y otro de copia para la óptica, esta le solicita el pago a efectuar el cual se realiza por adelantado, el cliente paga el servicio o producto en dependencia de su tipo, la recepcionista realiza el cobro y el almacenero registra la solicitud en el libro diario y elabora la orden según los datos obtenidos anteriormente. El operario atiende la solicitud y una vez terminado el trabajo le notifica al almacenero que su trabajo está terminado y este a su vez actualiza el listado de solicitudes del libro diario.

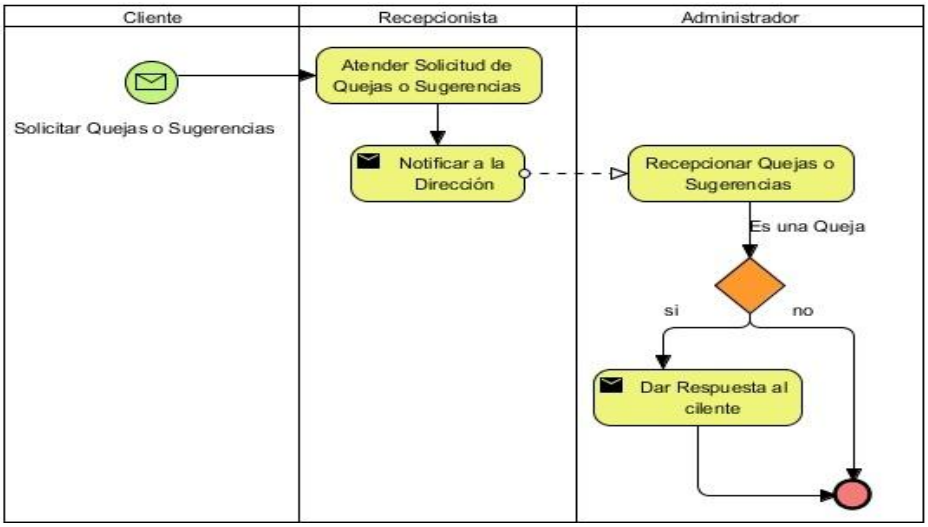

Figura: 2.2 Realizar Quejas o Sugerencias.

Realizar Quejas o Sugerencias: el cliente solicita presentar una queja o dejar una sugerencia con el objetivo de que la óptica mejore sus servicios y logre mayor calidad en sus productos, esta es planteada a la recepcionista la cual es la encargada de atenderla, luego se notifica a la Dirección de la óptica donde el administrador la almacena con los datos de contacto del cliente. Si lo que se almacena es una queja, entonces se le da una respuesta el cliente que sea apropiada y convincente, sino termina el proceso de realizar quejas o sugerencias.

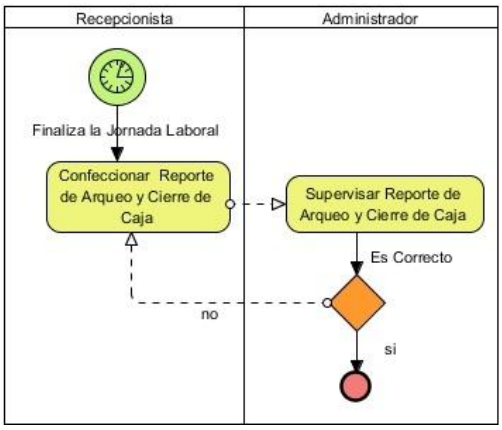

Figura: 2.3 Realizar Arqueo y Cierre de Caja.

Realizar Arqueo y Cierre de Caja: al finalizar la jornada laboral de la óptica se realiza el arqueo y cierre de caja, que no es más que hacer un balance entre el dinero que había en la caja registradora al iniciar el día y el dinero almacenado al culminar la sesión de trabajo y verificar si los ingresos coinciden con las reportadas por cada comprobante de pago de cada servicio. La recepcionista confecciona el reporte de arqueo y cierre de caja que contiene los datos de las ventas del día, el administrador es el encargado de supervisar la confección de este reporte. Si están correctos los datos, entonces finaliza este proceso, sino debe volverse a confeccionar el reporte.

Figura: 2.4 Elaborar Reportes.

Elaborar Reportes: el administrador es el encargado de confeccionar los reportes de las ventas y servicios prestados por la óptica, entre los que se encuentran los de materiales de trabajo, de medios básicos, de los servicios más solicitados por los clientes, entre otros. Estos reportes pueden ser diarios, semanales, mensuales o anuales a petición del Director o alguna entidad del Ministerio de Salud Pública (MINSAP).

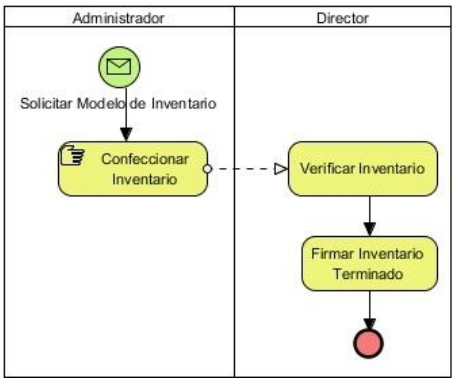

Figura: 2.5 Realizar Inventario.

Realizar Inventario: el administrador solicita los modelos de inventario y en dependencia de las necesidades de la óptica solicita los materiales y medios de trabajo a los almacenes del MINSAP, a la Empresa Provincial de Ópticas y de forma interna se le solicita al director del centro, el cual debe entregar los modelos y a su vez el administrador confecciona el inventario el cual contiene los medios básicos, los materiales de trabajo y la materia prima (armaduras, cristales, graduaciones, entre otros), luego el director firma el inventario terminado.

Nombre	Justificación
Cliente	Persona que interviene en los procesos referentes a solicitar un servicio y realizar quejas o sugerencias.
Recepcionista	Persona encargada de atender el proceso referente a solicitar servicio, gestionar las quejas o sugerencias y realizar el arqueo y cierre de caja.
Administrador	Persona que interviene en el proceso gestionar las quejas o sugerencias. Realiza la generación de reportes los cuales pueden ser diarios, semanales, mensuales o anuales, además supervisa la confección del arqueo y cierre de caja.
Operario	Persona que interviene en el proceso solicitar servicio, atendiendo la solicitud y una vez terminado el trabajo se lo notifica a la recepcionista para que actualice el listado de solicitudes en el libro diario.

Tabla: 2. 1 Personas que Intervienen en los procesos de negocio.

2.3 Reglas del negocio

"Las reglas de negocios (o las directivas empresariales) definen y controlan la estructura, el funcionamiento y la estrategia de una organización". (29)

Relación

- El cliente solicita uno o varios servicios de la óptica. A la asociación entre el cliente y el producto final se le denomina Prestaciones de Servicios.

Restricción de operaciones*:*

- Para registrar una solicitud, es necesario que el cliente presente la receta con la medición de la vista realizada en una consulta de optometría.
- Para la recepcionista registrar una queja, es necesario que el cliente presente su carnet de identidad y en el caso que así lo requiera debe proveer datos de contacto.
- El tiempo con que el administrador cuenta para dar respuesta a las quejas o sugerencias es de al menos tres días.
- Los datos de contacto provistos por el cliente son estrictamente confidenciales y no se les dará a conocer a nadie ajeno a la óptica.
- El cobro de los servicios prestados al cliente debe realizarse en moneda nacional y se le debe entregar un comprobante de pago.
- Para realizar una reclamación el cliente debe presentar su comprobante de pago, la recepcionista lo verifica y consulta al administrador para luego dar respuesta.
- Al confeccionar un reporte determinado, el administrador debe tener presente si los datos que este debe reflejar son del diario, semanal, mensual o anual.
- La recepcionista debe informar al cliente que debe recoger el producto en un periodo determinado por la institución en dependencia del tipo de solicitud.

2.4 Requisitos funcionales

Los requisitos funcionales son "*capacidades o condiciones que el sistema debe cumplir, especifican las acciones que el sistema debe ser capaz de realizar, sin tener ningún tipo de restricción física y cómo debe reaccionar ante situaciones particulares. Estos deben ser de fácil entendimiento para el usuario y para los desarrolladores".* (30)

La captura de requisitos es un acto para descubrir o averiguar lo que se quiere construir, está basada en la utilización de técnicas tales como cuestionarios, lluvia de ideas, entrevistas, entre otras. Las técnicas seleccionadas para el levantamiento de los requisitos del sistema fueron la lluvia de ideas y las entrevistas, además se realizó un estudio de sistemas similares para determinar las funcionalidades que fueran comunes y que no se hubieran detectado con las técnicas empleadas.

A continuación se mencionan los requisitos identificados:

RF1: Autenticar Usuario.

Permitir el acceso al sistema y a las funcionalidades del mismo según el rol asignado.

RF2: Cargar Configuración.

Permitir que el sistema cargue las funcionalidades según el rol del usuario autenticado.

RF3: Cerrar Sesión.

Permitir el cierre de la sesión cuando el usuario lo decida.

RF4: Asignar Rol a Usuario.

Permitir la asignación del rol a cada usuario una vez autenticado.

RF5: Gestionar Usuario.

Permitir la gestión de los usuarios que harán uso de la aplicación.

5.1 Registrar Usuario.
5.2 Mostrar Usuario.
5.3 Ver Detalles de Usuario.
5.4 Modificar Usuario.
5.5 Eliminar Usuario.
5.6 Modificar Contraseña.

RF6: Gestionar Tarjeta de Estiba.

6.1 Registrar Tarjeta de Estiba.
6.2 Mostrar Tarjeta de Estiba.
6.3 Ver Detalles de Tarjeta de Estiba.
6.4 Modificar Tarjeta de Estiba.
6.5 Eliminar Tarjeta de Estiba.

RF7: Gestionar Nomencladores.

Permitir la gestión de los siguientes nomencladores:

7.1 Gestionar Provincia.

7.2 Gestionar Municipios.

7.3 Gestionar Color.

7.4 Gestionar Tipos de Armadura.

7.5 Gestionar Tipos de Cristales.

7.6 Gestionar Tipos de Medios Básicos.

7.7 Gestionar Tipos de Materiales de Trabajo.

7.8 Gestionar Estados del Medio de Trabajo.

7.9 Gestionar Tipos de Graduaciones.

RF8: Gestionar Queja.

8.1 Permitir que se le adjunte una respuesta a una queja determinada.

8.2 Permitir la gestión de una Queja para la retroalimentación (Feedback) con el cliente y trabajar en la mejora de la calidad de los servicios y productos.

8.2.1 Registrar Queja.

8.2.2 Mostrar Queja.

8.2.3 Ver Detalles de Queja.

8.2.4 Registrar Respuesta a la Queja.

RF9: Gestionar Sugerencias.

Permitir la gestión de una sugerencia para la retroalimentación (Feedback) con el cliente y trabajar en la mejora de la calidad de los servicios y productos.

9.1 Registrar Sugerencias.

9.2 Mostrar Sugerencias.

RF10: Notificar Queja Registrada.

Permitir la notificación al cliente mediante correo electrónico que su queja ha sido registrada.

RF11: Notificar Respuesta de una Queja.

Permitir la notificación al cliente mediante correo electrónico la respuesta a su queja.

RF12: Generar Reporte de Venta.

Generar un reporte con las ventas, de un período de tiempo, donde se refleja todo el dinero recaudado por la óptica.

RF13: Generar Reportes de Quejas o Sugerencias.

Generar un reporte con los datos de todas las quejas y sugerencias según un rango de tiempo que se seleccione.

RF14: Generar Reporte de Medios Básicos.

Generar un reporte de inventario con los datos de todos los medios básicos de la óptica o los datos específicos según el rango de tiempo y el estado del medio básico que se seleccione.

RF15: Generar Reporte de Materiales de Trabajo.

Generar un reporte de inventario con los datos de todos materiales de trabajo de la óptica o los datos específicos que se seleccione.

RF16: Generar Reportes de Cristales de Espejuelos.

Generar un reporte de inventario con los datos de cristales de espejuelos o los datos específicos que se seleccione.

RF17: Generar Reporte de Solicitudes del Cliente.

Generar un reporte de inventario con los datos de las solicitudes del cliente a la óptica según un rango de tiempo y el estado de dicha solicitud que se seleccione.

RF18: Generar Reporte de Solicitudes del Almacén.

Generar un reporte de inventario con los datos de las solicitudes del almacén según un rango de tiempo y el estado de dicha solicitud que se seleccione.

RF19: Exportar Reportes a pdf.

Permitir la exportación de cada uno de los reportes generados al formato pdf.

RF20: Consultar existencia de graduación de lentes y/o espejuelos en la óptica.

Verificar la existencia de las graduaciones de los lentes en la óptica.

RF21: Gestionar Solicitud Cliente.

Permitir la gestión de la solicitud de un cliente.

21.1 Registrar Solicitud Cliente.

21.2 Mostrar Solicitud Cliente.

21.3 Ver Detalles de Solicitud Cliente.

21.4 Modificar Solicitud Cliente.

21.5 Eliminar Solicitud Cliente.

RF22: Gestionar Solicitud Almacén.

Permitir la gestión de la solicitud de un cliente.

22.1 Registrar Solicitud Almacén.

22.2 Mostrar Solicitud Almacén.

22.3 Ver Detalles de Solicitud Almacén.

22.4 Modificar Solicitud Almacén.

22.5 Eliminar Solicitud Almacén.

RF23: Generar Comprobante de Pago.

Permitir la generación del comprobante de pago que se le entregará al cliente.

RF24: Imprimir Comprobante de Pago.

Permitir la impresión del comprobante de pago de la venta realizada.

RF25: Modificar Estado de la Solicitud Cliente.

El sistema debe permitir modificar el estado de la solicitud.

RF26: Notificar Solicitud del Cliente Registrada.

Permitir la notificación al cliente mediante correo electrónico que su solicitud ha sido registrada.

RF27: Notificar Recogida de Producto.

Permitir la notificación al cliente mediante correo electrónico que debe recoger el producto en una fecha determinada.

RF28: Gestionar Información de Productos y Servicios.

Permitirla gestión de la información de los productos y servicios que brinda la óptica.

28.1 Registrar Información.

28.2 Mostrar Información.

28.3 Ver Detalles de Información.

28.4 Modificar Información.

28.5 Eliminar Información.

RF29: Consultar Información de Servicios y Productos.

Permitir al cliente visualizar la información, sin necesidad de autenticarse.

RF 30: Gestionar Medios Básicos.

Gestionar los Medios Básicos que entren a la óptica.

30.1 Registrar Medios Básicos.

30.2 Mostrar Medios Básicos.

30.3 Ver Detalles de Medios Básicos.

30.4 Modificar Medios Básicos.

30.5 Eliminar Medios Básicos.

RF 31: Gestionar Materiales de Trabajo.

Gestionar los Materiales de trabajo que entren a la óptica.

31.1 Registrar Materiales de trabajo.

31.2 Mostrar Materiales de trabajo.

31.3 Ver Detalles de Materiales de trabajo.

31.4 Modificar Materiales de trabajo.

31.5 Eliminar Materiales de trabajo.

RF 32: Realizar Arqueo y Cierre de Caja.

Proveer las funcionalidades que permitan realizar un balance del dinero en la caja registradora.

32.1 Verificar la cantidad de dinero al iniciar el día y la cantidad al final de la sesión de trabajo.

32.2 Realizar una comparación entre las dos cantidades.

32.3 Verificar que coincide con las ganancias del pago de los servicios brindados.

RF 33: Visualizar Gráficas y Estadísticas de Ventas.

Permitir la visualización de las gráficas que muestren el estado de las ventas especificando un rango de tiempo.

2.5 Requisitos no funcionales

Los requisitos no funcionales son *"propiedades, cualidades o restricciones que el producto debe cumplir, tales como restricciones de tiempo, estándares, de desarrollo, entre otros con el objetivo de lograr un producto confiable, atractivo y seguro"*. (30)

2.5.1 Software

La aplicación debe estar instalada en un servidor central con sistema operativo de Linux/Windows, debe tener además instalado el servidor web Apache Tomcat v7.0.30. Debe disponer del navegador web Mozilla Firefox v13.0 o superior para acceder a la aplicación y el visor de PDF Foxit Reader v6.0 o superior para visualizar los documentos en formato digital.

2.5.2 Hardware

Teniendo como base los requisitos mínimos de hardware de las herramientas y tecnologías seleccionadas se plantea:

Para explotación del cliente: PC Pentium III o superior, CPU 133 MHZ o superior, 512 MB de RAM o superior y 650 MB de espacio libre en disco.

Para explotación del servidor: PC Pentium III o superior, CPU 800M Hz Intel Pentium III o superior, 512 MB de RAM o superior y 750 MB de espacio libre en disco.

2.5.3 Apariencia o interfaz externa

El sistema deberá poseer una interfaz gráfica uniforme que incluirá un menú en forma de árbol a la izquierda con las funcionalidades que ofrecerá el sistema, a la derecha del mismo estará ubicada la información, la parte superior contará con un banner que incluirá el logotipo y nombre de la aplicación.

2.5.4 Seguridad

La seguridad del sistema está basada en niveles de acceso sobre las funcionalidades y la información. Los principios básicos que determinan la seguridad del sistema son los siguientes:

- La seguridad se establecerá por perfiles que se le asignarán a los usuarios que interactúen con el sistema, para garantizar que la información almacenada solo sea modificada y/o visualizada por los usuarios autorizados.
- Seguridad a nivel del gestor de datos, estableciendo privilegios de acceso sobre los datos almacenados y garantizando el acceso a los datos sólo a aquellos usuarios con los permisos necesarios para hacerlo, y sólo a los datos que le esté permitido acceder de acuerdo a sus privilegios en el sistema.

2.6 Descripción de la solución propuesta

Las ópticas forman parte del Sistema de Salud Cubano, su objetivo es brindar servicios y productos para la satisfacción de las necesidades de los clientes que visiten dichos establecimientos. Para dar respuesta a la situación problemática planteada, haciendo uso de la plataforma de desarrollo seleccionada, para la construcción de la aplicación, se propone como solución: la creación de un Sistema de Gestión de la Información para las Ópticas de Cuba el cual está dividido en los siguientes módulos:

- ***Módulo Administración:*** permitir la autenticación de los usuarios, cargar la configuración para cada uno de ellos según el perfil (rol) que tenga, además permitir el cierre de la sesión, también debe permitir la gestión de los usuarios, lo que integra la modificación, búsqueda y eliminación de los mismos y la gestión de los perfiles para cada usuario del sistema.
- ***Módulo Reportes:*** permitir la generación de reportes con la información referente a las ventas, quejas o sugerencias, materiales de trabajo, medios básicos, así como las solicitudes del cliente y del almacén, estos se realizan en un periodo de tiempo determinado por la Dirección de la Óptica, ya sean diarios, semanales, mensuales, entre otros.
- ***Módulo Nomencladores:*** permitir la gestión de los nomencladores como los tipos de armadura, de cristales, de graduaciones, de medios básicos, además la gestión de los municipios, provincias y el color para las armaduras de los espejuelos.
- ***Módulo Información:*** permitir al cliente consultar la información referente a los productos y servicios que brinda la óptica, permitir además al administrador de la óptica la realización del arqueo y cierre de caja mediante el cual se hace un balance entre el dinero inicial y el final para conocer las ganancias, este se realiza diariamente

y la visualización de las gráficas y estadísticas de venta. Permitir la gestión de la información de los productos y servicios, de los materiales de trabajo, de los medios básicos, lo cual incluye consultar, modificar y eliminar los mismos.

- ***Módulo Retroalimentación:*** permitir la gestión de las sugerencias, la gestión de las quejas, así como realizar notificaciones por correo electrónico del registro y respuestas de las quejas.
- ***Módulo Solicitudes:*** permitir la gestión de las solicitudes del cliente, las solicitudes del almacén, consultar la existencia de graduaciones de lentes o espejuelos en la óptica, así como generar e imprimir el comprobante de pago que se le entregará al cliente una vez que solicite el producto o servicio que requiere. Además permitir las notificaciones por correo electrónico de la recogida del producto y las solicitudes del cliente registradas.

2.7 Modelo de casos de uso del sistema

Los casos de usos son una técnica de definición de los requisitos funcionales, es decir, el comportamiento del sistema. El modelo de casos de uso es el encargado de describir lo que hace el sistema para cada usuario. (31)

En la aplicación sistema de gestión de la información para las ópticas en Cuba, intervienen los siguientes actores:

Nombre	Justificación
Cliente	Es un actor del sistema que no precisa de autenticación en el mismo para acceder o consultar la información publicada sobre los servicios y productos de la óptica.
Usuario	Es un actor genérico del sistema el cual debe autenticarse para obtener acceso mediante un perfil (rol), con un conjunto de funcionalidades asignadas para cada rol.
Administrador del Sistema	Es un actor del sistema especializado del actor Usuario. Encargado de administrar y controlar el comportamiento del sistema, dígase la gestión de la información, de usuarios, entre otros.
Administrador de la	Es un actor del sistema especializado del actor Usuario.

Óptica	Encargado de la generación de los reportes de medios básicos, materiales de trabajo, de venta, de solicitudes de cliente y almacén, entre otros así como la exportación de éstos al formato PDF, además debe dar respuesta a las quejas y realizar el arqueo y cierre de caja.
Recepcionista	Es un actor del sistema especializado del actor Usuario. Encargado de gestionar la solicitud del cliente, la solicitud del almacén, las quejas o sugerencias, además de generar e imprimir los comprobantes de pago.

Tabla: 2. 2 Actores que intervienen en el Sistema.

Para mejor comprensión y agrupación de los casos de uso de la aplicación se hizo necesario realizar el siguiente diagrama de paquetes.

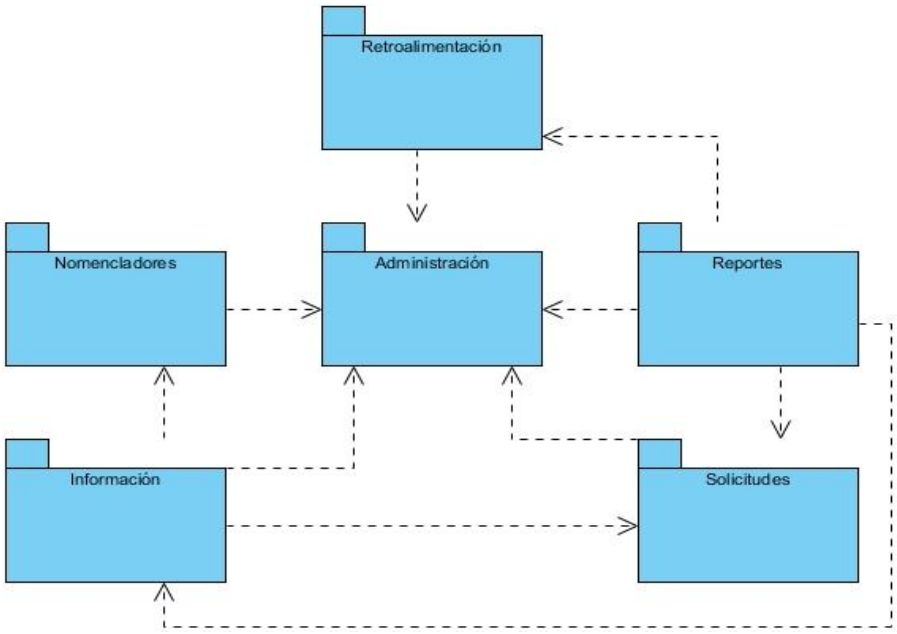

Figura: 2.6 Diagrama de Paquetes del Sistema.

Los diagramas para la relación entre actores y casos de uso se presentan a continuación:

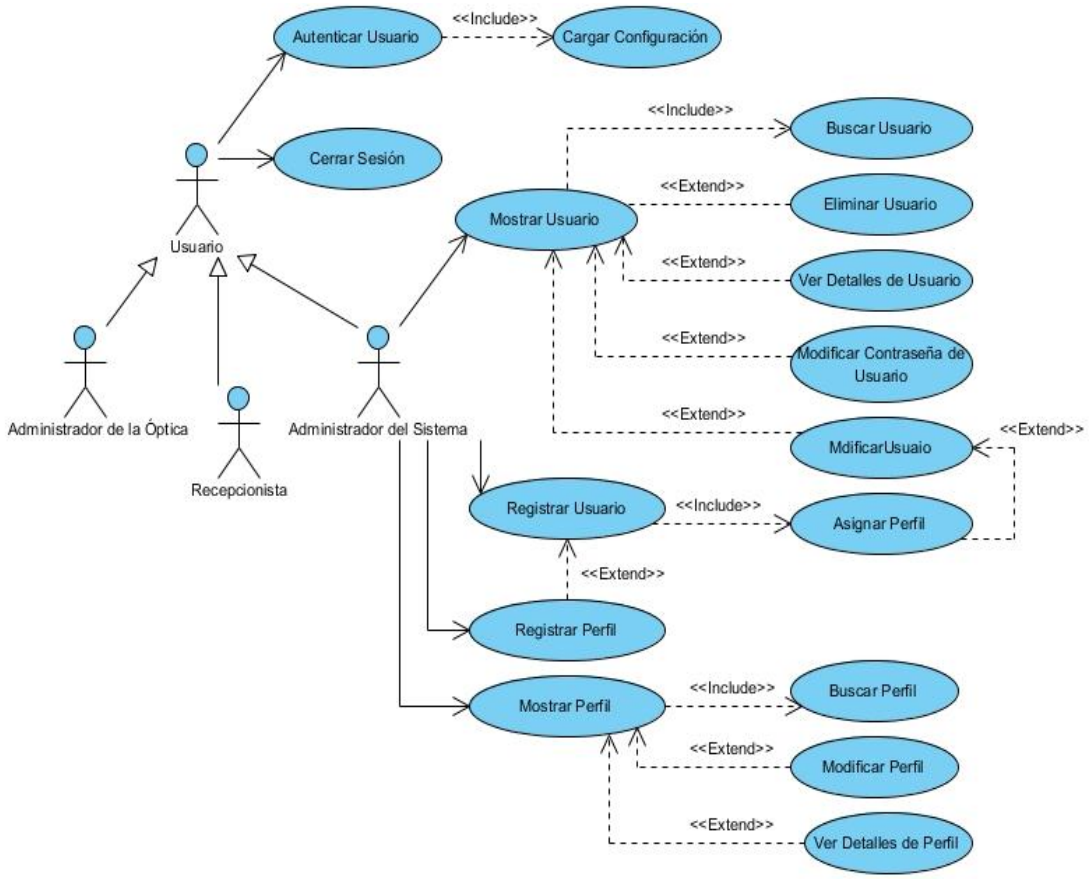

Figura: 2.7 Paquete Administración.

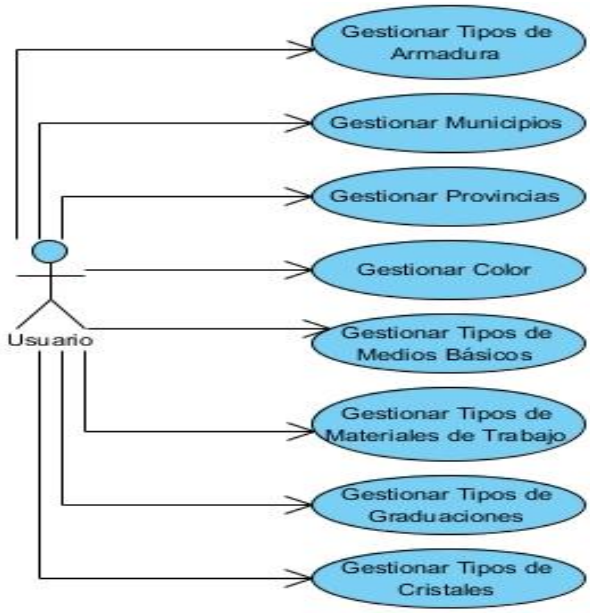

Figura: 2.8 Paquete de Nomencladores.

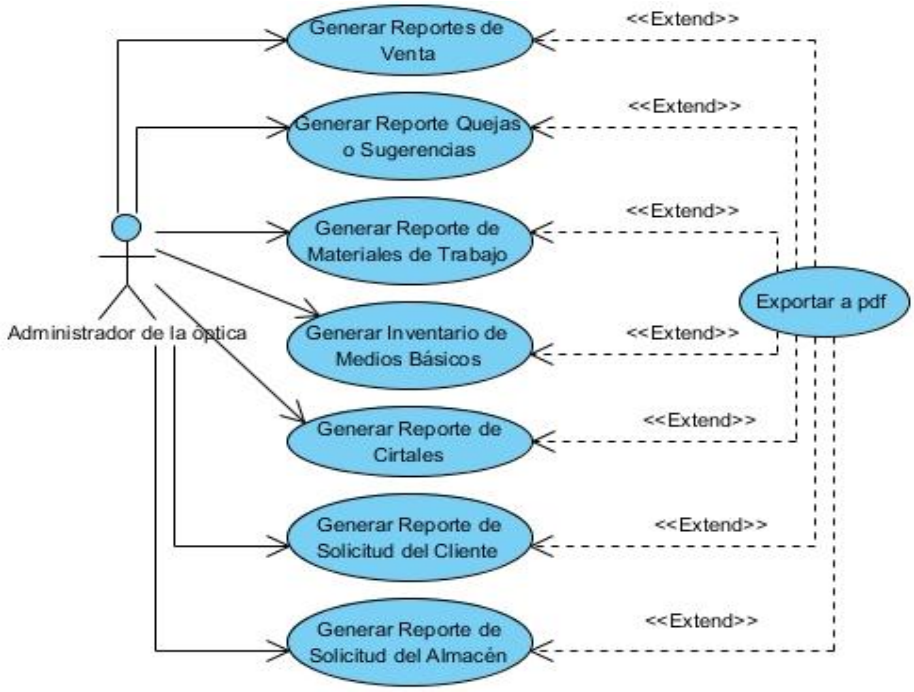

Figura: 2.9 Paquete de Reportes.

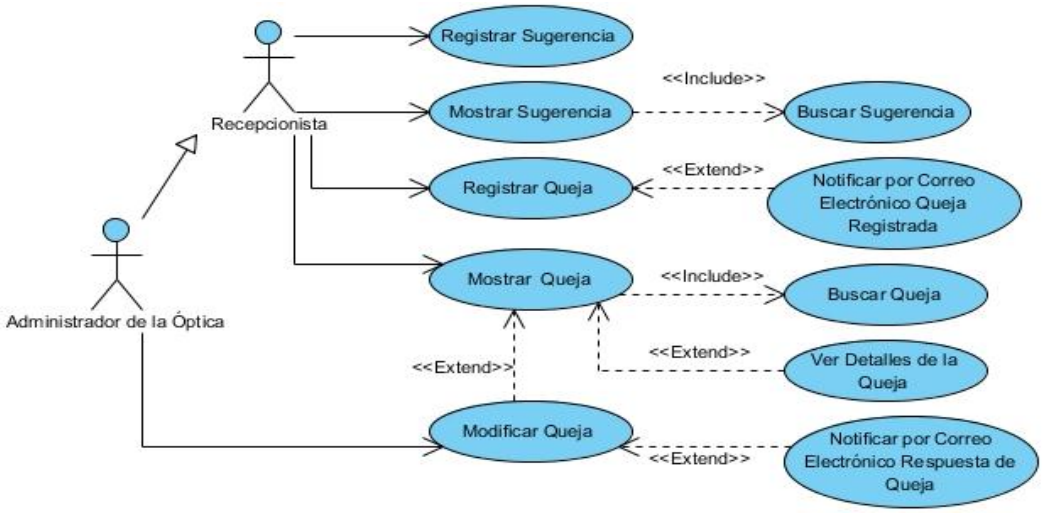

Figura: 2.10 Paquete de Retroalimentación.

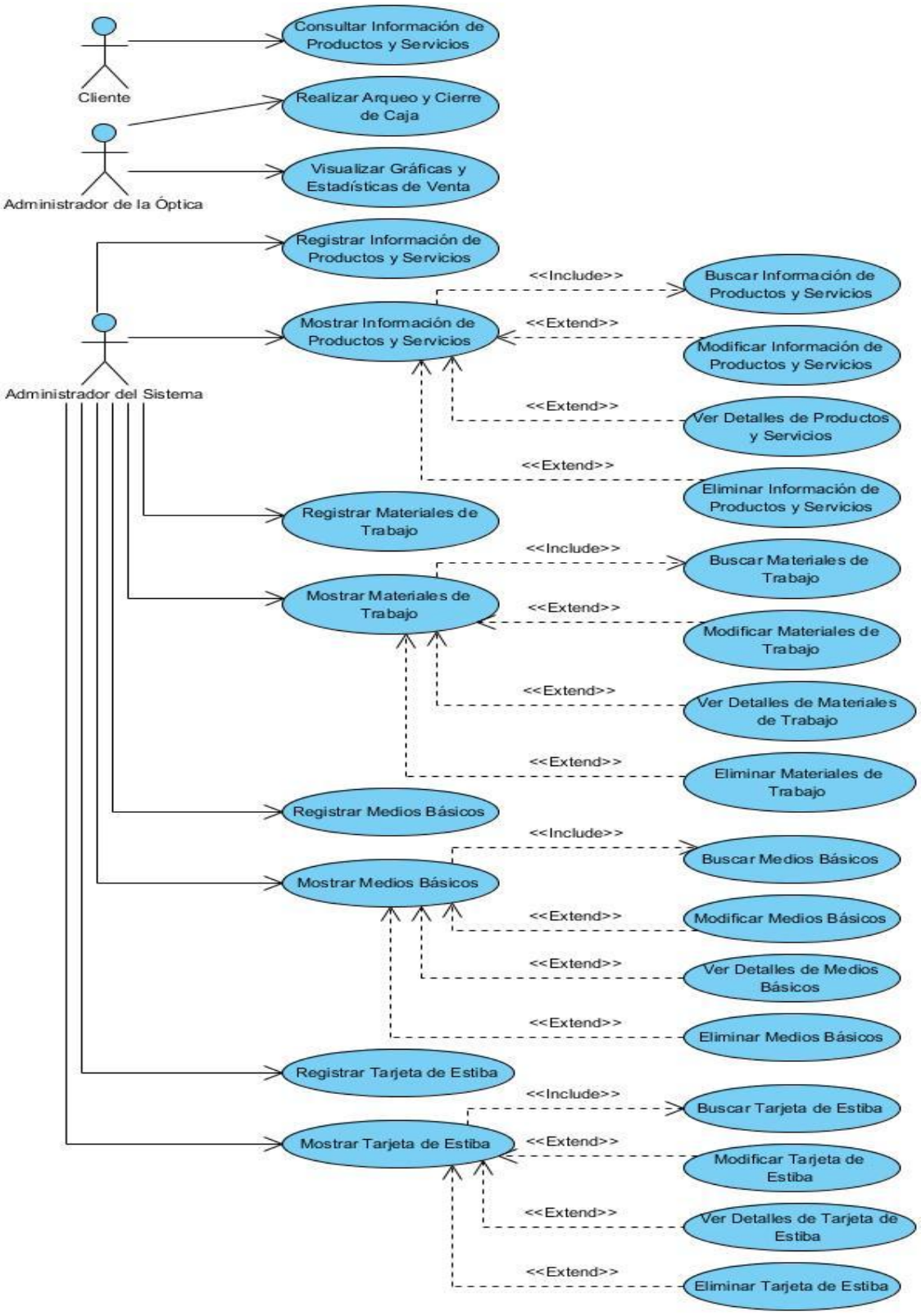

Figura: 2.11 Paquete de Información.

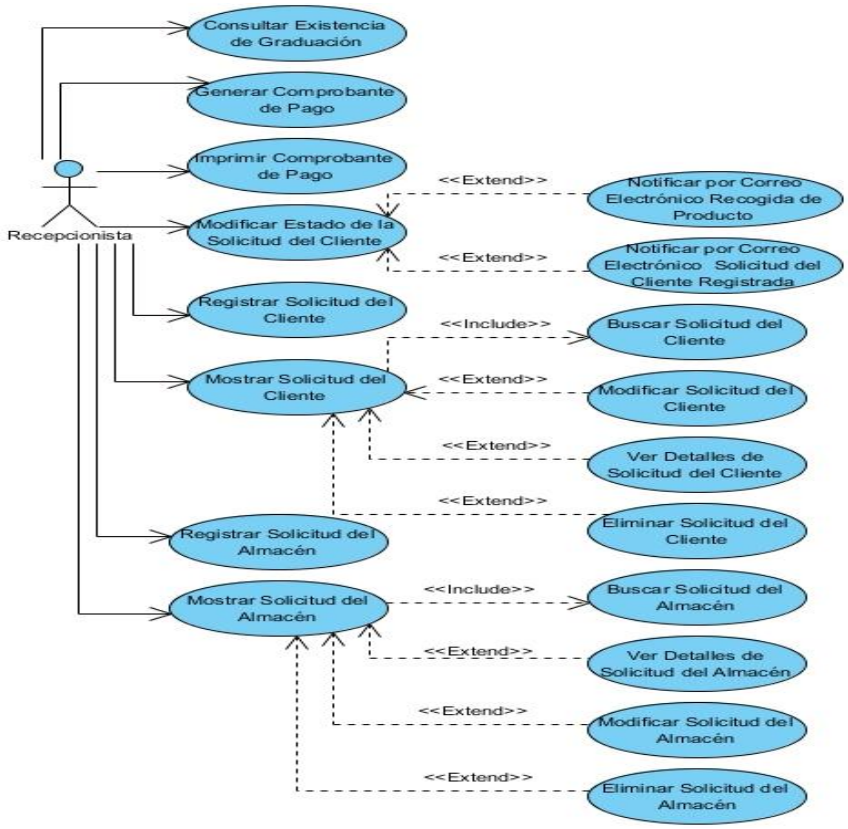

Figura: 2.12 Paquete de Solicitudes.

2.8 Expansión de los casos de uso

A través de la expansión de los casos de uso es posible describir paso a paso la secuencia de eventos que los actores siguen para completar un proceso mediante el sistema. Las descripciones de los casos de usos del sistema se muestran en el Anexo1.

2.9 Conclusiones parciales

En el presente capítulo se realizó el modelado de los procesos del negocio relacionados con la gestión de la información que se llevan a cabo dentro de una óptica, se realizó una descripción detallada de la solución propuesta de acuerdo con los elementos del negocio, que constituyó la base para la obtención de los requisitos funcionales y no funcionales del sistema, de acuerdo con las características del entorno donde se usará la aplicación, los cuales fueron agrupados y detallados a través de casos de uso. Una vez realizados estos pasos se procede a realizar el diseño del sistema.

Capítulo III: Diseño del Sistema

3.1 Introducción

En el presente capítulo se hace referencia a los aspectos correspondientes al diseño, teniendo presente la propuesta del sistema y las funcionalidades seleccionadas. Se confeccionan los diagramas de clases del diseño, el cual contribuye a la consolidación de una arquitectura estable. Se especifican además los patrones de diseño.

3.2 Arquitectura de Grails

La arquitectura de Grails está concebida por 3 capas lógicas principales: Web Layer, Service Layer y Data Layer, cada capa está separada de la siguiente, su interacción se lleva a cabo mediante interfaces que definen funcionalidades que la misma debe brindar, también llamadas fachadas cuya función fundamental es asegurar que el acoplamiento sea el más bajo posible y la abstracción del funcionamiento de la capa inferior. (21)

Grails implementa el patrón (Modelo-Vista-Controlador, MVC) en la que se separa la lógica empresarial de la presentación de la aplicación, lo cual le permite cambiar fácilmente el aspecto de su aplicación sin modificar su comportamiento.

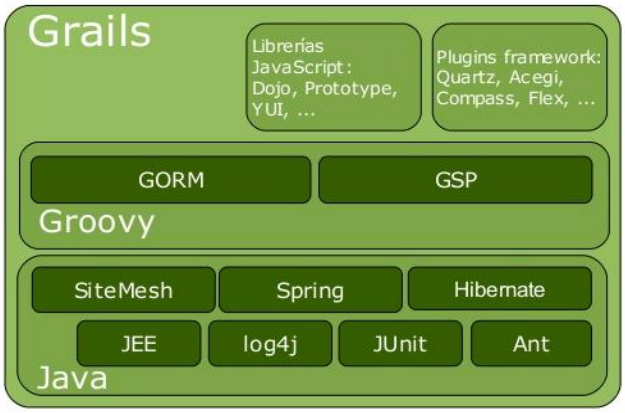

Figura: 3.1 Arquitectura de Grails. (23)

Web Layer: La capa web se compone de dos partes principales: vistas y controladores, teniendo en Controllers las clases controladoras y en View las Groovy Server Pages o más conocidas como GSP. En la Lógica de Presentación se manejará todo el flujo web utilizando la implementación del patrón MVC que brinda Grails mediante Spring MVC. Esto permite cambiar fácilmente el aspecto de la aplicación, sin modificar su

comportamiento. La capa de presentación se compone principalmente de: modelo, vistas, controladores. (23)

- **Controlador:** Un controlador de Grails es una clase responsable por el manejo de los pedidos provenientes de la aplicación, es el encargado de mantener el flujo de comunicación entre las vistas y el modelo, están definidos por los Groovy controllers o controladores Groovy.
- **Modelo:** Una de las actividades fundamentales llevadas a cabo por los controladores en Grails es obtener los datos que serán mostrados en la vista, está definido por clases de dominio de Groovy las cuales se mapean en la base de datos utilizada y permiten el manejo y almacenamiento de los datos.
- **Vista:** la tecnología JSP es utilizada por Grails para la interacción con el usuario, basada en una implementación mediante GSP, el mismo permite a los desarrolladores mezclar las etiquetas de lenguajes de marcas tradicionales como HTML con código Java para producir vistas dinámicas. Las Vistas son los recursos que junto al modelo generado por los controladores le permiten al cliente visualizar la información.

Service Layer: La capa de servicios es la encargada de encapsular toda la lógica de la aplicación en fachadas de negocio que son utilizadas por los controladores en la capa de presentación, dejándoles el manejo de flujo de solicitudes con redirecciones. Sus clases radicarán según la arquitectura propuesta por Grails en el paquete Services.

Data Layer: La capa de datos es la encargada de manejar los objetos de acceso a datos separándolos del mecanismo de persistencia utilizado, mediante las interfaces que exponen las operaciones de persistencia. Grails para evitar a los programadores tener que trabajar directamente con el sistema gestor de base de datos y tablas, permite trabajar con objetos en su lugar. Utiliza Hibernate, la biblioteca más popular para Java, como una herramienta de Mapeo Objeto-Relacional (ORM). Sin embargo dada la naturaleza dinámica de Grails y la adopción del convenio sobre la configuración, se crea sobre una versión superior de la implementación de Hibernate llamado Grails Mapeo Objeto-Relacional (GORM) que simplifica el trabajo con Hibernate y elimina cualquier configuración externa. (32)

3.3 Arquitectura del sistema

La arquitectura del sistema se refiere a *"una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema".* (33)

3.3.1 Capas lógicas del sistema

Capa Presentación

Los componentes de esta capa son los encargados de proveer una interfaz entre el sistema y el usuario, básicamente es responsable de brindarle información al usuario por parte del sistema y viceversa. A esta capa pertenecen los archivos javascript (.js) encargados de validar los campos y velar que sean insertados correctamente; las vistas (.gsp) le permite al usuario interactuar con el sistema de forma directa; los controladores que son los que reciben las solicitudes hechas por el usuario, realiza las operaciones de lógica de negocio sobre los modelos y decide la vista que será mostrada y las hojas de estilo (.css) las cuales por medio de reglas brindan un formato y colores a los componentes de la vista.

Capa de Negocio

Esta capa contiene los servicios que son los componentes encargados de implementar la lógica de negocio de la aplicación. Se comunica con la capa de presentación, para recibir las peticiones que el usuario ha realizado y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él.

Capa de Datos

Es la capa donde se almacenan los datos mediante la capa de negocio, encargada de ofrecer, modificar, almacenar, borrar y recuperar datos, mediante el gestor (o los gestores) de bases de datos que la aplicación requiera. El repositorio de datos descansa sobre XML para la correcta construcción de la capa de presentación y está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos y reciben solicitudes de almacenamiento o recuperación de información.

3.4 Patrones utilizados

Un patrón es conocido como "*una descripción de un problema y la solución, a la que se da un nombre, y que se puede aplicar a nuevos contextos; idealmente, proporciona consejos sobre el modo de aplicarlo en varias circunstancias, y considera los puntos fuertes y compromisos"*. (34) En otras palabras, brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares.

3.4.1 Patrones Arquitectónicos

Un patrón arquitectónico especifica un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes. Resuelve problemas arquitectónicos, de adaptabilidad a requisitos cambiantes, performance, modularidad, acoplamiento, entre otros. (34)

Modelo Vista Controlador (MVC)

Este patrón separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. Grails sigue un patrón de diseño muy popular en el mundo del desarrollo de aplicaciones web, el MVC, el cual propone que los componentes del sistema estén organizados en 3 capas diferentes según su misión dentro del mismo. (21)

- **Modelo o Capa de Datos:** esta capase encarga de encapsular los datos y las funcionalidades, contiene además los componentes que representan y gestionan los datos manejados por la aplicación, es decir, son los encargados de realizar las operaciones relacionadas con la base de datos.
- **Vistas o Capa de Presentación:** los componentes de esta capa son los encargados de interactuar con el usuario, mostrarle las diversas acciones disponibles y el estado actual de los datos del sistema.
- **Controlador o Capa de Control:** posee los componentes que luego de recibir las entradas u órdenes hechas por el usuario, los cuales son traducidos a solicitudes de servicio que gestionan la aplicación de la lógica del negocio sobre el modelo de datos y determina la vista que se debe mostrar a continuación.

Cuando se dice que la capa de control "gestiona la aplicación de la lógica de negocio" se hace referencia a que estos son los responsables de que esta se aplique, esto no quiere decir que se implemente la lógica de negocio en las clases controladoras, esta es implementada en una cuarta capa, la cual se explica a continuación:

- **Capa de Servicios:** Contiene los componentes encargados de implementar la lógica de negocio correspondiente a la aplicación a desarrollar.

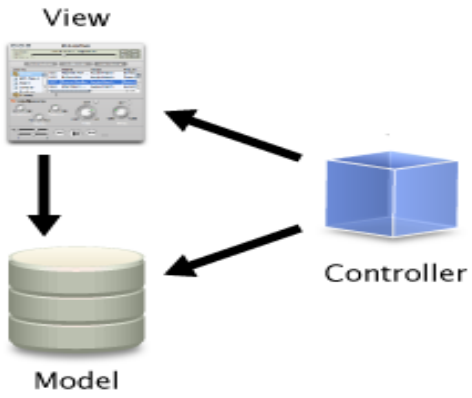

Figura: 3.2 Modelo-Vista-Controlador.

3.4.2 Patrones de diseño

Un patrón de diseño está relacionado con los aspectos del diseño de los subsistemas. Es una solución estándar para un problema común de programación y una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios. (34)

Inversión de control (IoC)

La inversión de control es otro patrón utilizado por Grails, a través del cual las dependencias de un componente no deben ser gestionadas desde el propio componente, con el objetivo que este sólo contenga la lógica necesaria para hacer su trabajo. Al crear un componente en la aplicación, Grails configura a Spring para controlar su ciclo de vida y sus dependencias (qué otros componentes necesita para desarrollar su trabajo y cómo seguirlos).

El objetivo principal de este patrón es mantener los componentes lo más sencillos posibles, incluyendo únicamente código que tenga relación con la lógica de negocio, logrando que la aplicación sea más fácil de comprender y mantener.

3.4.2.1 Patrones GRASP

GRASP (Generales de Software para Asignación de Responsabilidades*)* son patrones generales de software que describen los principios fundamentales de la asignación de responsabilidades a objetos. El nombre se eligió para indicar la importancia de captar estos principios, con el objetivo de diseñar el software de manera eficaz. (34)

- **Controlador:** Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación para aumentar la reutilización de código y a su vez tener un mayor control del flujo de eventos del sistema, se encarga de asignar responsabilidades a clases específicas facilitando la centralización de actividades como la validación, la seguridad, entre otras. En Grails los controladores son los encargados de posibilitar una capa intermedia entre las vistas y el modelo.
- **Alta cohesión:** Se aplica en la mayoría de las clases del diseño las cuales deben ser coherente, ya que en cada una solo se implementan las funcionalidades que le corresponden.
- **Bajo acoplamiento:** Se refiera a tener las clases lo menos relacionadas entre sí que se pueda, de forma tal que al producirse una modificación en alguna de ellas, tenga la menor repercusión posible en el resto de clases, incrementando la reutilización y disminuyendo la dependencia entre las clases.
- **Experto:** Se mantiene el encapsulamiento, los objetos utilizan su propia información para llevar a cabo sus tareas de las clases que tiene. De esta manera se garantiza una alta cohesión y un bajo acoplamiento.

3.4.2.2 Patrones GOF

Los patrones GoF (Gang of Four) son una serie de posibles soluciones a problemas que suelen ser comunes en el desarrollo del software, se divide en tres grandes categorías como creacionales, estructurales y de comportamiento. (35)

- **Singleton:** Garantiza que una clase sólo tenga una instancia, proporcionando un punto de acceso global a la misma, este patrón se utiliza en la instanciación de los

servicios en los controladores. En Grails por defecto todos los servicios que contiene son Singleton.

- **Decorador / Decorator:** El uso de este patrón se centra en la utilización de SiteMesh integrado a la arquitectura del framework utilizado. El cual permite partiendo de una plantilla inicial de la página, incorporar modificaciones a cada vista en particular.

3.5 Diagrama de clases del diseño

Un diagrama de clases proporciona una perspectiva estática que representa el diseño estructural del sistema mostrando un conjunto de clases, sus atributos y las relaciones entre ellos. (36)

Los diagramas de clases son utilizados durante el proceso de diseño de los sistemas, donde se establece el diseño conceptual de la información que contendrá el sistema. A continuación se muestran los diagramas gestionar solicitudes de cliente, de almacén y el gestionar usuarios, los restantes se encuentran en el Anexo 2.

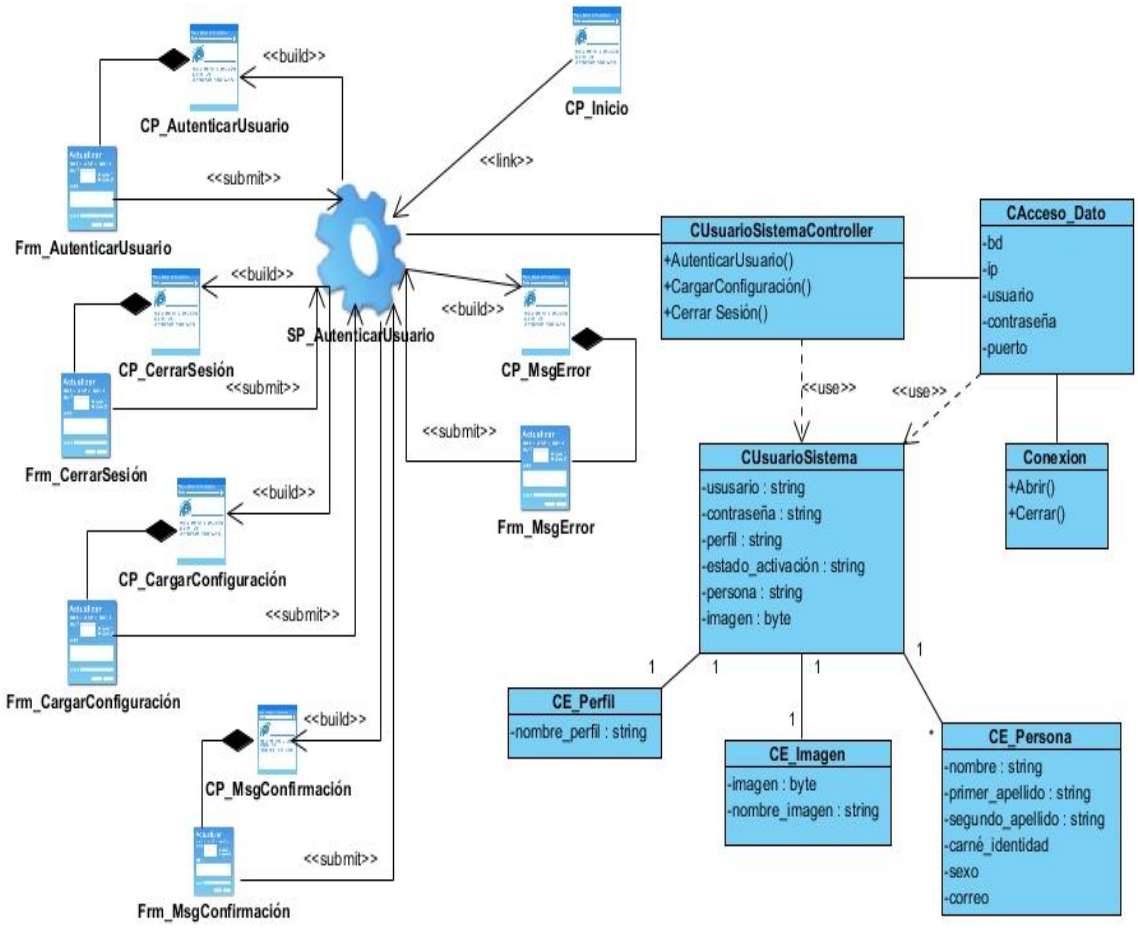

Figura: 3.3 Diagrama de clase del diseño Autenticar Usuario.

3.6 Diagrama entidad-relación

El Diagrama Entidad-Relación (DER) le permite al ingeniero del software identificar los objetos de datos y sus relaciones por medio de una notación gráfica, define en su análisis estructural los datos que son adicionados, almacenados, modificados y producidos dentro de una aplicación. (37)

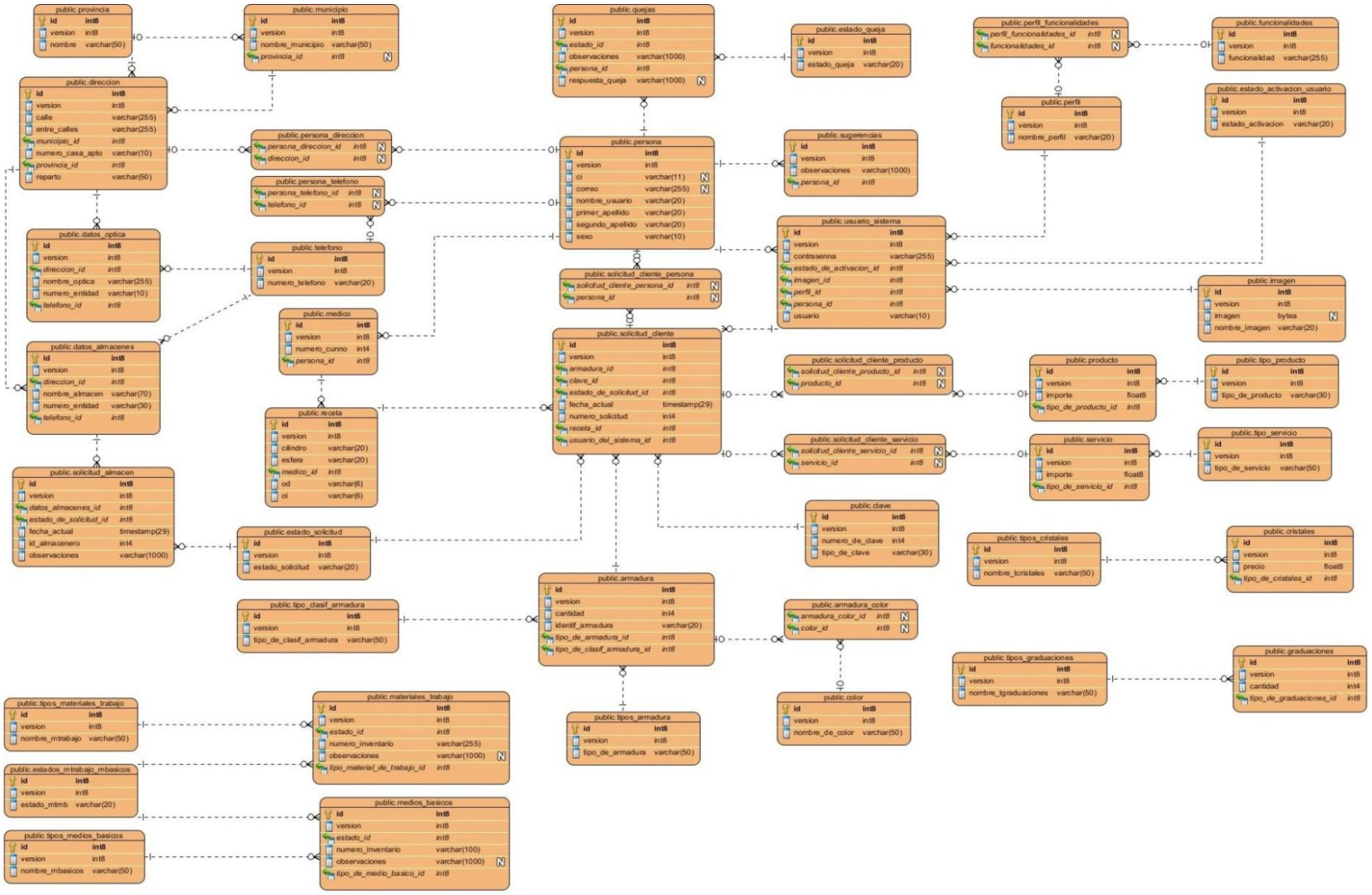

Figura: 3.4 Diagrama Entidad-Relación.

3.7 Conclusiones parciales

Al concluir el presente capítulo se realizó el análisis de la arquitectura del sistema propuesta por el marco de trabajo Grails, se han generado artefactos tales como los diagramas de clases del diseño elaborados para darle solución al sistema, así como el modelo de datos perteneciente al mismo. Se hizo referencia a los patrones seleccionados para el diseño e implementación del sistema, lo cual sirvió de base para próximos capítulos.

Capítulo IV: Implementación y Prueba

4.1 Introducción

En el presente capítulo se organizan los componentes mediante el diagrama de componentes para definir la estructura en capas de la aplicación y se realiza además el diagrama de despliegue del sistema. Se define además la estrategia de prueba a seguir y los resultados de las mismas una vez aplicadas al sistema, contribuyendo a mejorar la calidad, usabilidad e identificar fallos en la aplicación.

4.2 Diagrama de componentes

Los diagramas de componentes son utilizados para modelar los componentes del sistema incluye además los artefactos que son implementados por estos componentes y las relaciones entre ellos. (37)

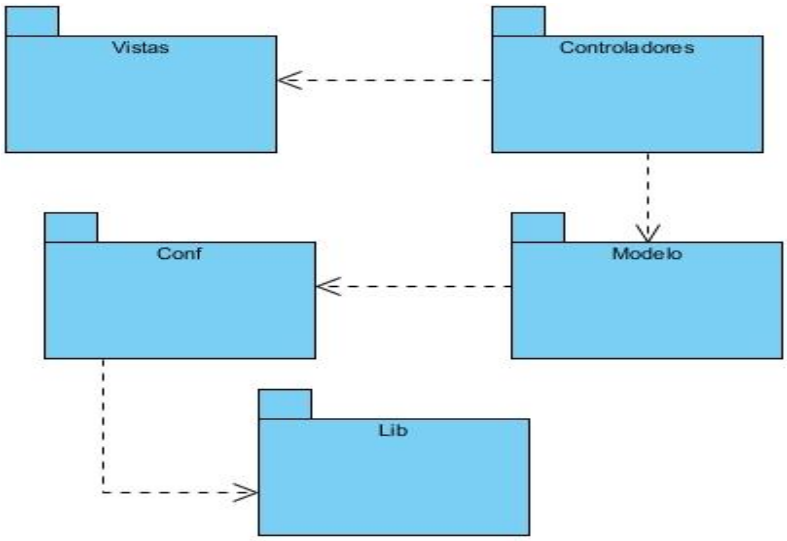

Figura: 4.1 Paquete de Componentes.

4.3 Descripción de los componentes

4.3.1 Paquete de componentes "Vistas"

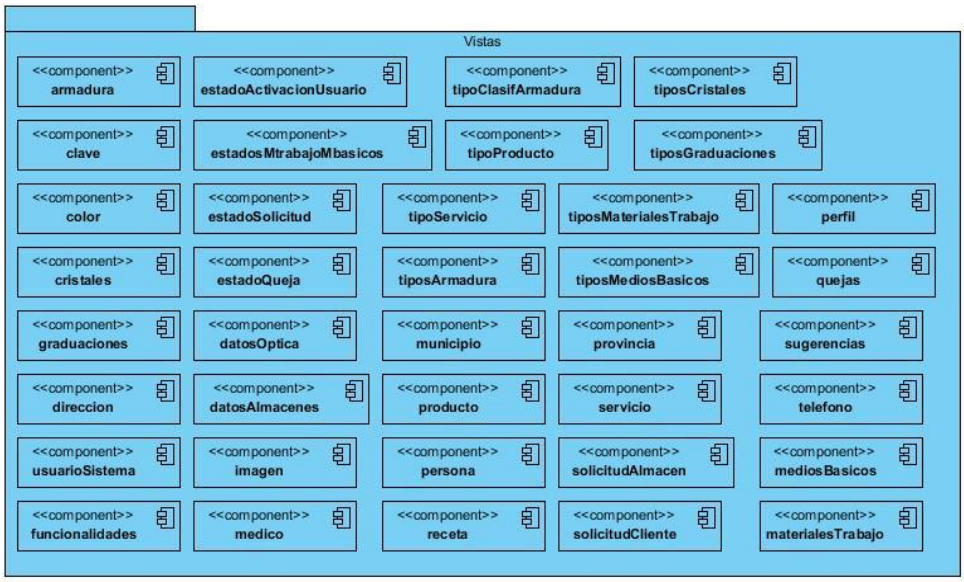

Figura: 4.2 Paquete Vistas.

El paquete de componente "Vistas" es una representación de las paginas gsp con las cuales el usuario interactúa, permitiendo la entrada de datos al sistema y realizar consulta de dichos datos.

4.3.2 Paquete de componentes "Lib"

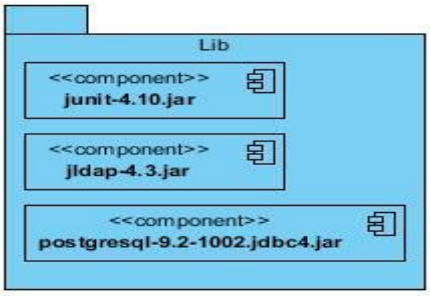

Figura: 4.3 Paquete Lib.

Este paquete es utilizado para lograr la conexión a la Base de datos con el uso del driver correspondiente al gestor de Bases de datos utilizado. Se utiliza además una librería que permite la conexión a Unit (junit-4.10) para las pruebas unitarias del sistema.

4.3.3 Paquete de componentes "Controladores"

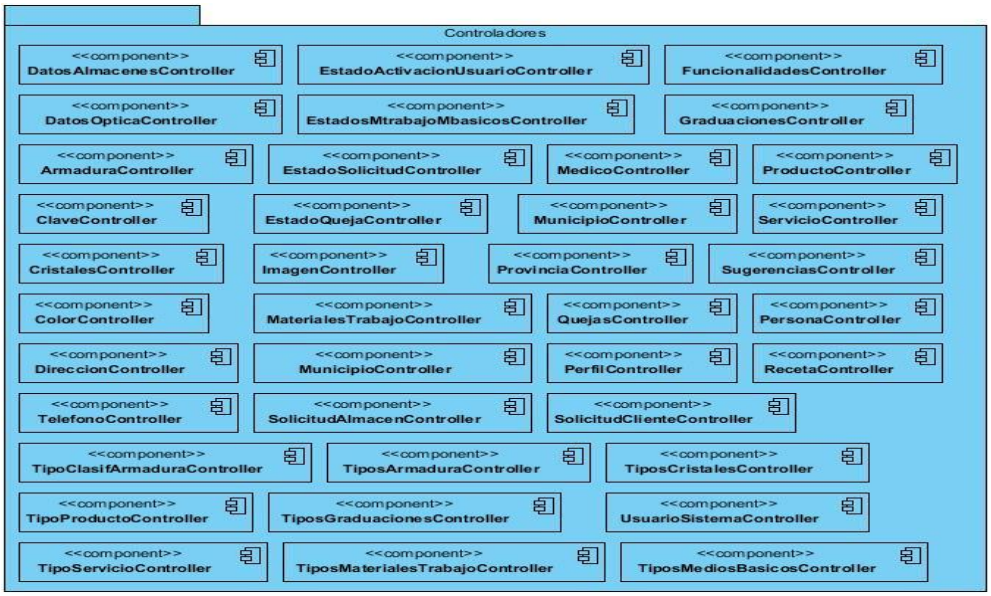

Figura: 4.4 Paquete Controladores.

El paquete de componentes "Controladores" es una representación de las operaciones que permiten el flujo y la comunicación entre las vistas y el modelo, son los encargados de recibir las órdenes por parte del usuario, gestionar la ejecución de la lógica de negocio y posteriormente actualizar la vista para que el usuario pueda ver como ha quedado el modelo de datos tras las actualizaciones.

4.3.4 Paquete de componentes "Conf"

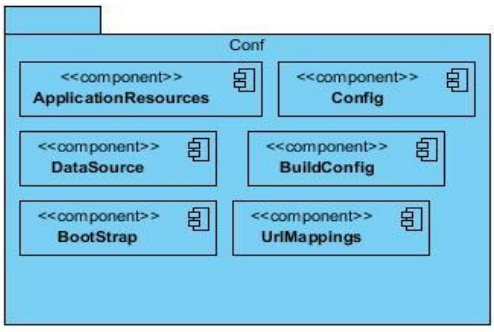

Figura: 4.5 Paquete Conf.

Este paquete es utilizado para la configuración y definición del acceso local o remoto a los datos. Constituye un componente del Framework donde se ubican ficheros groovy de configuración. Garantiza el control de las acciones en los controladores para lograr el acceso a las acciones realizadas por cada usuario del sistema.

4.3.5 Paquete de componentes "Modelo"

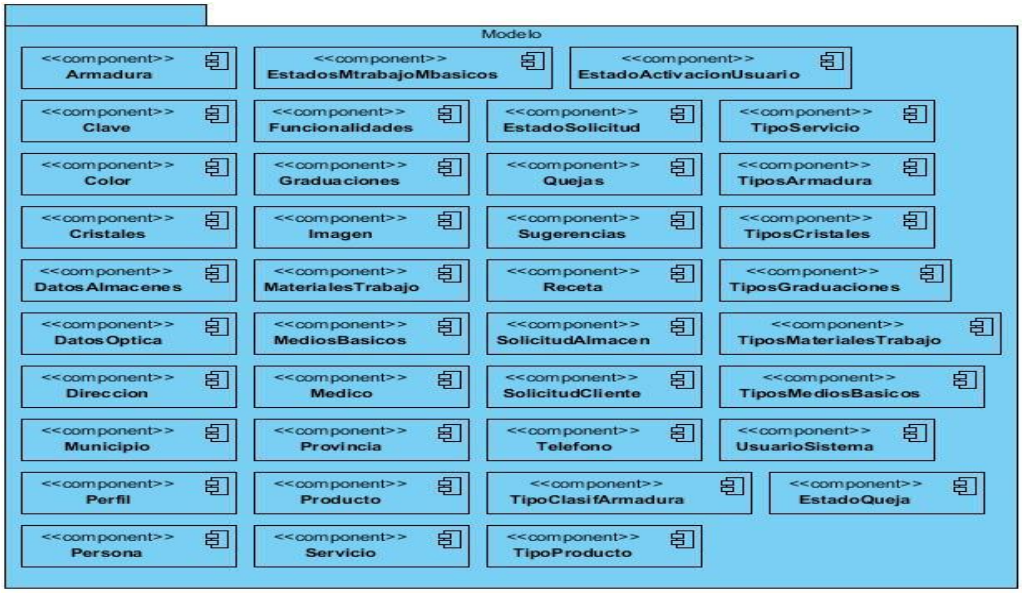

Figura: 4.6 Paquete Modelo.

Son las clases que representan todo el negocio del sistema. Estas clases del dominio constituyen entidades o tablas que son utilizadas en la base de datos mapeadas por el GORM, siendo este un gestor de persistencia para controlar el ciclo de vida de dichas entidades.

4.4. Diagrama de despliegue

"El diagrama de despliegue se utiliza para modelar la disposición física de los componentes de hardware utilizado en la implementación del sistema y la relación entre cada uno de ellos". (37)

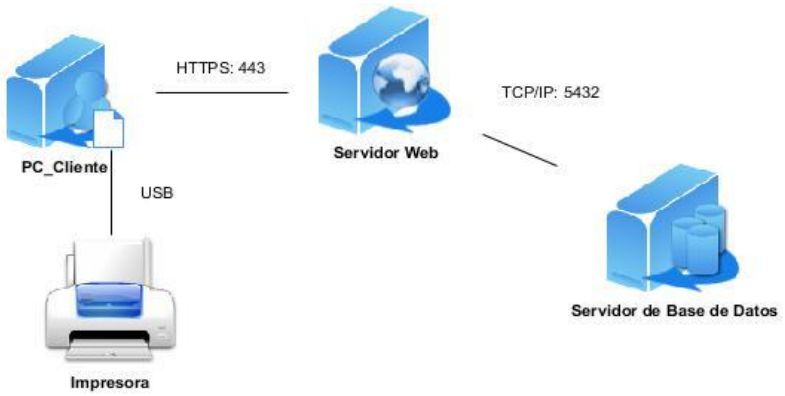

Figura: 4.7 Diagrama de Despliegue.

Para el despliegue del sistema es necesario:

- Una PC Cliente que puede ser de tipo portátil o de escritorio la cual debe tener instalado un navegador web preferentemente Mozilla Firefox v13.0 o superior y el Foxit Reader v6.0 o superior.
- Un Servidor Web que debe tener instalado el contenedor web Apache Tomcat v7.0.30 que permita las conexiones con el cliente, generando las respuestas a sus peticiones. Las conexiones del cliente con el servidor se realizan mediante el protocolo HTTPS.
- Un servidor de base de datos que debe tener instalado el sistema gestor de base de datos PostgreSQL v9.1.
- Una impresora que será utilizada para la impresión de los documentos, reportes, comprobantes de pago, entre otros.

4.5 Pruebas del sistema

El único instrumento adecuado para determinar el estado de la calidad de un producto software es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software en específico o al sistema de software en su totalidad, con el objetivo de medir el grado en que se cumplen los requisitos. En las pruebas se usan casos de prueba, especificados de forma estructurada mediante técnicas de pruebas. (38)

4.5.1 Tipos de pruebas

Funcionalidad

- Función: Pruebas fijando su atención en la validación de las funciones, métodos, servicios, caso de uso.
- Seguridad: Asegurar que los datos o el sistema solamente es accedido por los actores deseados.

Fiabilidad

- Integridad: realiza la valoración de la resistencia a fallos que tiene el sistema.
- Estructura: esta prueba es aplicada a aplicaciones web con el objetivo de asegurar que los enlaces están conectados y muestran su contenido deseado.

Rendimiento

- Performance profile: es una prueba enfocada en la monitorización del tiempo de ejecución, el acceso a los datos, identificar los cuellos de botellas, así como los procesos ineficientes.

Soportabilidad

- Instalación: se encarga de asegurar la instalación de las configuraciones de software y hardware bajo ciertas condiciones como poco espacio en disco, entre otras.

4.5.2 Métodos de pruebas

Los métodos de prueba definen la estrategia a seguir en función de la verificación y validación del sistema diseñados para descubrir fallos. Los métodos más significativos son las pruebas de caja blanca y las pruebas de caja negra.

Pruebas de Caja Blanca: es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba. (37) Se encarga de comprobar los caminos lógicos del software a través de los casos de prueba que se ejerciten con conjuntos específicos de condiciones y/o bucles y sus límites, así como las estructuras de datos.

Pruebas de Caja Negra: denominada también prueba de comportamiento, se centran en los requisitos funcionales del software, tiene como objetivo verificar que la entrada se acepta correctamente y que ejerciten en su totalidad todos los requisitos funcionales de un sistema. (37)

La prueba de caja negra intenta encontrar las siguientes categorías de errores (37):

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externos.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

4.5.3 Estrategia de prueba seguida

4.5.3.1 Pruebas unitarias

La estrategia seguida para la realización de las pruebas al Sistema de Gestión de la Información para las Ópticas de Cuba contiene dos niveles: el primero lo constituyen las pruebas de Unidad y el segundo nivel la prueba de Sistema. En el primer nivel fueron realizadas las pruebas de forma automática haciendo uso del sistema de prueba que posee el Framework utilizado, mientras que el segundo nivel fue usada la técnica manual denominada diseño de casos de prueba los cuales incluyen las pruebas de funcionalidad y usabilidad.

Los test unitarios son aquellos en los que se verifica que un método se comporta como debería, sin tener en cuenta su entorno. Esto significa que cuando se ejecutan las pruebas unitarias de un método Grails no inyectará ninguno de los métodos dinámicos con los que cuenta la aplicación cuando se está ejecutando, permitiendo que al trabajar con entidades o servicios el desarrollador es el responsable de crear y gestionar los objetos. (41)

Durante el proceso de desarrollo, es posible probar el estado de la aplicación mediante el comando: **grails test-app.** Este ejecuta las pruebas unitarias y genera un reporte en texto y HTML, al cual será posible recurrir en caso de existir algún fallo en la aplicación.

Para crear un test unitario de un servicio se ejecuta el script:

grails create-unit-test<nombre de la prueba>

Este script crea una nueva batería de pruebas (encargada de asegurar la calidad y guiar el proceso de desarrollo del sistema) en la carpeta Unit Test del proyecto. La clase que se genera hereda de GrailsUnitTestCase, lo cual pone a disposición una serie de métodos

para facilitar la tarea de probar los servicios definidos. Esta prueba es ejecutada sin levantar el contexto Grails, de tal manera que las entidades no disponen de los métodos dinámicos inyectados por GORM y no se realiza la inyección de dependencias. Para esto, GRAILS aprovecha el soporte nativo en Groovy para distintas formas tales como los métodos Mock, permitiendo que los objetos se comporten en pruebas como lo harían en ejecución.

4.5.3.2 Casos de prueba

Un caso de prueba es un conjunto de entradas de pruebas, condiciones de ejecución, resultados esperados desarrollados para cumplir un objetivo en particular o una función esperada. La entidad más simple que siempre es ejecutada como una unidad, desde el comienzo hasta el final.

Los casos de pruebas deben verificar:

- Si el producto satisface los requisitos del usuario, tal y como se describe en las especificaciones.
- Si el producto se comporta como se desea, tal y como se describe en las especificaciones funcionales del diseño.

Un caso de prueba se diseña según las funcionalidades descritas en los casos de uso, este es elaborado previo a la realización de las pruebas funcionales de la aplicación. Cada planilla de caso de prueba recoge la especificación de un caso de uso, dividido en secciones y escenarios, para hacer más fructífera la ejecución de las pruebas.

A continuación se muestra el diseño del caso de prueba, correspondiente al flujo básico del CU Autenticar Usuario y del CU Registrar Usuario. La versión expandida de algunos de los Diseños de Casos de Prueba de los CU se encuentra en los anexos a partir del Anexo 3.

Escenario	Descripción	Nombre de Usuario	Contraseña	Respuesta del sistema	Flujo central
EC 1.1: Autenticar Usuario.	Se autentica un usuario en el sistema satisfactoriamente.	V oferrer	V orqui123	Carga la configuración para el usuario autenticado.	1. Introducir el Nombre de Usuario y la Contraseña. 2. Seleccionar la opción "Aceptar".
EC 1.2: Cancelar Autenticación.	Se cancelan las operaciones que se estén realizando.	NA	NA	Cancela la operación que se esté realizando y cierra la aplicación.	1. Seleccionar la opción "Cancelar".
EC 1.3: No Existe Conexión con la BD.	Se muestra un mensaje indicando que no existe conexión con la BD.	V oferrer	V orqui123	Comprueba que no existe conexión con la BD y muestra un mensaje indicándolo. Cierra la aplicación.	1. Introducir el Nombre de Usuario y la Contraseña. 2. Seleccionar la opción "Aceptar". 3. Seleccionar la opción "Aceptar".
EC 1.4: Datos de Autenticación Incorrectos.	Se muestra un mensaje indicando que existen datos incorrectos.	V oferrer	I yen1	Muestra un mensaje indicando que el nombre de usuario y la contraseña proporcionados no son correctos. Cierra la interfaz del mensaje y muestra la interfaz de autenticación.	1. Introducir el Nombre de Usuario y la Contraseña. 2. Seleccionar la opción "Aceptar". 3. Seleccionar la opción "Aceptar".
		I .yyt0rn3s.	V orqui123		
EC 1.5: Usuario Autenticado en otra sesión.	Se muestra un mensaje indicando que la sesión del usuario está abierta.	V oferrer	V orqui123	Muestra un mensaje indicando que el usuario ya está autenticado en la aplicación.	1. Introducir el Nombre de Usuario y la Contraseña. 2. Seleccionar la opción "Aceptar". 3. Seleccionar la opción "Aceptar".

Tabla: 4. 1 Diseño de casos de prueba Autenticar Usuario.

4.6 Resultados de las pruebas

A continuación se muestra gráficamente un resumen de las no conformidades detectadas al sistema, utilizando los casos de prueba diseñados:

Figura: 4.8 Total de No Conformidades por Iteraciones.

Entre las principales no conformidades encontradas se pueden mencionar:

- ✓ Tamaño de fuente muy pequeño en algunas interfaces como principal.gsp.
- ✓ Faltas de ortografía (tildes) en las interfaces de registrar, mostrar medios básicos, entre otros.
- ✓ Problemas en la validación de los campos.
- ✓ Incorrecta validación de los formatos de datos de cada entrada hecha por el usuario.

Las no conformidades encontradas fueron resueltas satisfactoriamente. Esto no evidencia que la aplicación se encuentre libre de posibles errores pero si se resolvieron todas las no conformidades detectadas durante las tres iteraciones realizadas.

4.7 Conclusiones parciales

En este capítulo se muestran los artefactos generados durante la fase de implementación como el diagrama de despliegue y los diagramas de componentes, el código implementado estuvo enfocado en el cumplimiento de los requisitos funcionales y no funcionales antes definidos. Se realizaron pruebas para demostrar el correcto funcionamiento del sistema.

Capítulo V: Estudio de la Factibilidad.

5.1 Introducción

Una vez conceptualizado el sistema es importante evaluar la factibilidad para saber si se puede desarrollar de acuerdo a un determinado alcance y si es conveniente llevarlo a cabo. En el presente capítulo se hace un estudio de factibilidad, beneficios y costo del sistema propuesto, mediante el cual se obtendrá una estimación del tamaño del software, el esfuerzo y el tiempo necesario para el desarrollo.

5.2 Método de estimación Puntos por Caso de Uso

El método de estimación por Puntos de Casos de Uso fue desarrollado en 1993 por Gustav Karner, el cual se utiliza para la estimación del tiempo de desarrollo de un sistema mediante la asignación de "pesos" a un número de factores que lo afectan. (42)

El método resulta muy efectivo para estimar el esfuerzo requerido en el desarrollo de los primeros casos de uso de un sistema, si se sigue una aproximación iterativa como el Proceso Unificado de Software, consta de cuatro etapas, en las que se desarrollan los siguientes cálculos:

- Factor de peso de los actores sin ajustar (UAW), el cual es un análisis de la cantidad de actores presentes en el sistema y la complejidad de cada uno de ellos.
- Factor de peso de los casos de uso sin ajustar (UUCW), está dado por la complejidad de cada caso de uso.
- Puntos de caso de uso ajustados (UCP).
- Esfuerzo horas-hombre.

Para el cálculo se procede de forma similar a Puntos de Función: se calcula una cuenta no ajustada Puntos Casos de Uso (UAUCP), asignando una complejidad a los actores y a los casos de uso.

5.2.1 Cálculo de puntos de Casos de Uso sin ajustar

UUCP = UAW + UUCW

Dónde:

UUCP: Puntos de Casos de Uso sin ajustar

UAW: Factor de Peso de los Actores sin ajustar

UUCW: Factor de Peso de los Casos de Uso sin ajustar

Tipo	Descripción	Peso	Cant * Peso:
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API, Application Programming Interface).	1	0*1
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto.	2	0*2
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica.	3	2*3
Total:			6

Tabla: 5. 1 Cálculo del factor de peso de los actores sin ajustar (UAW).

No.	Nombre de Caso de Uso	Cantidad de Transacciones	Tipo
1	Mostrar Usuario.	5	Medio
2	Asignar Rol.	3	Simple
3	Exportar a PDF.	7	Medio
4	Mostrar Información de Productos y Servicios.	4	Medio
5	Mostrar Materiales de Trabajo.	4	Medio
6	Mostrar Medios Básicos.	4	Medio
7	Mostrar Queja.	3	Simple
8	Modificar Estado de Solicitud.	2	Simple
9	Mostrar Solicitud de Cliente.	4	Medio
10	Mostrar Solicitud de Almacén.	4	Medio

Tabla: 5. 2 Cantidad de transacciones por casos de uso.

Tipo	Descripción	Peso	Cant * Peso:
Simple	El caso de uso contiene de 1 a 3 transacciones.	5	3*5
Medio	El caso de uso contiene de 4 a 7 transacciones.	10	7*10
Complejo	El caso de uso contiene más de 8 transacciones.	15	0*15
Total:			85

Tabla: 5. 3 Cálculo del factor de peso de los CU sin Ajustar (UUCW)

La cantidad de transacciones está dada partiendo de la descripción textual del caso de uso, mientras más detallada esté la descripción textual, más transacciones se puede encontrar y la estimación será más exacta.

Luego: UUCP = UAW + UUCW
UUCP = 6 + 85
UUCP = 91

5.2.2 Cálculo de puntos de Casos de Uso ajustados

UCP = UUCP * TCF * EF

Dónde:

UCP: Puntos de casos de uso ajustados
UUCP: Puntos de casos de uso sin ajustar
TCF: Factor de complejidad técnica
EF: Factor de ambiente

El cálculo del Factor de Complejidad Técnica (TCF) está dado en la siguiente fórmula donde cada uno de los factores se cuantifica con un valor de 0 a 5.

Dónde: $TCF = 0.6 + 0.01 * \Sigma (Peso_i * Valor_j)$

Factor	Descripción	Peso	Valor	Comentario	Σ ($Peso_i$ * $Valor_i$)
T1	Sistema distribuido.	2	4	El sistema es distribuido.	8
T2	Objetivos de performance o tiempo de respuesta.	1	4	Se requiere un buen rendimiento del sistema.	4
T3	Eficiencia del usuario final.	1	2	No hay restricciones.	2
T4	Procesamiento interno complejo.	1	0	No hay cálculos complejos.	0
T5	El código debe ser reutilizable.	1	4	Es reutilizable.	4
T6	Facilidad de instalación.	0.5	5	El sistema debe ser fácil de instalar.	2.5
T7	Facilidad de uso.	0.5	5	Debe ser un sistema amigable y de fácil uso.	2.5
T8	Portabilidad.	2	4	Se requiere que el sistema sea portable.	8
T9	Facilidad de cambio.	1	5	El sistema debe ser flexible ante posibles cambios.	5
T10	Concurrencia.	1	5	Si hay concurrencia.	5
T11	Incluye objetivos especiales de seguridad.	1	5	El sistema gestiona información cuya confidencialidad es muy importante.	5
T12	Provee acceso directo a terceras partes.	1	3	Provee acceso a terceras partes.	3
T13	Se requieren facilidades especiales de entrenamiento a usuarios.	1	2	No se requieren facilidades especiales para el entrenamiento de los usuarios.	2
Total:					51

Tabla: 5. 4 Cálculo de factor de complejidad técnica (TCF).

TCF = 0.6 + 0.01 * Σ ($Peso_i$ * $Valor_j$)

TCF = 0.6 + 0.01 * 51

TCF = 1.11

El cálculo del Factor de Ambiente (EF) está dado en la siguiente fórmula donde cada uno de los factores se cuantifica con un valor de 0 a 5.

Dónde: EF = 1.4 - 0.03 * Σ ($Peso_i$ * $Valor_j$)

Factor	Descripción	Peso	Valor	Comentario	Σ ($Peso_i$ * $Valor_j$)
E1	Familiaridad con el modelo de proyecto utilizado.	1.5	2	El grupo no está familiarizado con ningún modelo de proyecto.	3
E2	Experiencia en la aplicación.	0.5	3	No hay mucha experiencia.	1.5
E3	Experiencia en orientación a objetos.	1	4	La mayoría del grupo ha programado Orientado a Objetos.	4
E4	Capacidad del Analista líder.	0.5	0	No hay analista principal.	0
E5	Motivación.	1	3	El grupo está motivado.	3
E6	Estabilidad de los requisitos.	2	3	Se esperan cambios en futuras versiones.	6
E7	Personal part-time	-1	0	Todo el equipo es full-time.	0
E8	Dificultad del lenguaje de programación	-1	3	Se usará el lenguaje Java orientado a objetos, el cual no es difícil de aprender.	-3
Total:					14.5

Tabla: 5. 5 Cálculo del factor ambiente (EF).

EF = 1.4 - 0.03 * 14.5

EF = 0.96

Luego UCP = UUCP * TCF * EF

UCP = 51 * 1.11* 0.96

UCP = 54,3456

5.2.3 Cálculo del Esfuerzo

E = UCP * CF

Dónde:

E: esfuerzo estimado en horas-hombre.

UCP: puntos de Casos de Uso ajustados.

CF: factor de conversión.

Esta estimación del esfuerzo en horas-hombre se realiza teniendo presente sólo el desarrollo de la funcionalidad especificada en los casos de uso.

Para calcular Factor de Conversión (CF):

CF = 20 horas-hombre (si Total EF ≤ 2)

CF = 28 horas-hombre (si Total EF = 3 ó Total EF = 4)

CF = abandonar o cambiar sistema (si Total EF ≥ 5)

Total EF = Cant EF < 3 (entre E1 – E6) + Cant EF > 3 (entre E7 – E8)

Como Total EF = 2 + 0

Total EF = 2

Entonces CF = 20 horas -hombre ya que Total EF = 2.

Luego E = UCP * CF

E = 54,3456 * 20 horas-hombre

E = 1086,91 horas-hombre

5.2.4 Distribución del esfuerzo entre las actividades

Actividad	% Esfuerzo	Valor Esfuerzo
Inicio	10	108.691
Diseño	20	217.382

Implementación	40	434.764
Prueba	15	163.036
Sobrecarga(otras actividades)	15	163.036
Total	100	1086.909

Tabla: 5. 6 Distribución de Esfuerzo estimado en los flujos de trabajo de RUP.

5.3 Beneficios tangibles e intangibles

El desarrollo del Sistema de Gestión de la Información para las Ópticas de Cuba aporta un considerable beneficio económico al país, donde el mayor aporte está enfocado en la disponibilidad de una herramienta informática, que permitirá la reducción de costos en accesorios y material de oficina de uso diario necesarios para realizar los procesos dentro de las ópticas tales como papel, bolígrafos, cintas de impresoras, papel para impresoras, entre otros.

A demás de los beneficios tangibles que proporciona la implementación de este software se generan los siguientes beneficios intangibles:

1. Optimizar los procesos agilizando el control y preservación de la información gestionada.
2. Disminución del tiempo y el esfuerzo que se invierte en la gestión de la información dentro de las Ópticas de Cuba, al gestionar la información de cada una de ellas.
3. Brindar información de forma eficiente y confiable, que sirva de apoyo a la toma de decisiones por parte del cliente.
4. Mejor capacidad de búsqueda y actualización de información, reduciendo la fuerza de trabajo en el proceso y aumentando el control de los recursos que dispone las ópticas.

5.4 Conclusiones parciales

En el presente capítulo se efectuó el estudio de factibilidad a través del método de estimación por puntos de casos de uso, el cual permitió arribar a la conclusión de que resulta factible la implementación del Sistema de Gestión de la Información para las Ópticas de Cuba, ya que el costo en su desarrollo es mucho menor que el costo actual de

un sistema de este tipo en el mercado internacional, según la estimación realizada. Además se analizaron los beneficios tangibles e intangibles que proporcionará este sistema.

Conclusiones Generales

El presente trabajo describe el proceso de desarrollo del Sistema de Gestión de la Información para las Ópticas de Cuba, siendo la respuesta al problema a resolver planteado. Para lo cual fue realizado un estudio del marco teórico en el que se sustenta la investigación, centrados en las características que debe poseer un sistema de gestión de información de una óptica.

Con el uso de la metodología de desarrollo RUP se generó abundante documentación del sistema que sirve de base para la implementación de versiones posteriores.

La utilización de Grails como framework que facilita el desarrollo de aplicaciones web, permitió la reducción de tiempo y esfuerzo por parte de los programadores, logrando una mayor productividad. Además se dio cumplimiento a los objetivos propuestos inicialmente obteniendo los siguientes resultados:

- Se realizó un análisis de soluciones informáticas existentes para los Sistemas de Gestión de la Información para ópticas en el ámbito internacional.
- Se describieron las herramientas y tecnologías a utilizar para la implementación del sistema.
- Se definieron requisitos indispensables a ser resueltos con el sistema a desarrollar.
- Se generaron los artefactos relacionados con el diseño y la implementación del sistema.
- Se realizaron pruebas al sistema desarrollado de acuerdo a las estrategias definidas.

Recomendaciones

Con los resultados obtenidos en la investigación y la experiencia adquirida en el desarrollo de la aplicación, se recomienda:

- Ampliar las funcionalidades del sistema según se vayan incrementando las necesidades del cliente.
- Utilizar la información y la documentación generada para proyectos futuros similares, siempre que se cumplan las normas de confidencialidad establecidas en la Universidad.
- Desplegar el sistema en todas las ópticas de Cuba.

Glosario de Términos

ACL (Lista de Control de Acceso): es una lista de los permisos asociados a un objeto.

API (Interfaz de Programación de Aplicaciones): conjunto de funciones y procedimientos (métodos) que ofrece una biblioteca para ser utilizado por otro software como una capa de abstracción.

CASE: Computer Assisted Software Engineering (Ingeniería de Software Asistida por Computadora).

HTML: HyperText Markup Language (Lenguaje de Marcado de Hipertexto).

HTTP: HyperText Transfer Protocol (Protocolo de transferencia de hipertexto).

JavaEE: Java Enterprise Edition.

JDBC: Java Data Base Connectivity (Conectividad de Bases de Datos de Java).

JSP: Java Server Pages (Páginas Servidoras de Java).

JVM: Máquina Virtual de Java.

Open ID: Proveedor de Identidad.

Open-source: Representa el software de dominio público, esto significa sin licencia, cuyo código fuente está disponible y se le permite usar y modificar.

ORDBMS: Sistema de Gestión de Bases de Datos Objeto-Relacionales.

POO: es un paradigma de programación que usa los objetos en sus interacciones, para diseñar aplicaciones y programas informáticos.

XML: Extensible Markup Language (Lenguaje de Marcas Extensible).

Referencias Bibliográficas

1. **Academia, Real.** Real Academia Española. [En línea] 2013. [Citado el: 7 de febrero de 2013.] http://lema.rae.es/drae/?val=%C3%B3ptica.

2. —. Real Academia Española. [En línea] 2013. [Citado el: 7 de enero de 2013.] http://lema.rae.es/drae/?val=informaci%C3%B3n.

3. **Enrique Dans.** Information Management. *Reflexiones sobre las tecnologías de la información.* [En línea] 28 de noviembre de 2006. [Citado el: 7 de enero de 2013.] http://informationmanagement.wordpress.com/category/gestion/gestion-de-la-informacion/.

4. **The British Standards Institution.** bsi. [En línea] © The British Standards Institution, 2013. [Citado el: 10 de febrero de 2013.] http://www.bsigroup.com.mx/es-mx/Auditoria-y-Certificacion/Sistemas-de-Gestion/De-un-vistazo/Que-son-los-sistemas-de-gestion/.

5. **Díaz Pérez, Maidelyn, Contreras, Yimian de Liz y Rivero Amador, Soleidys.** SciELO. *Características de los sistemas de información que permiten la gestión oportuna de la información y el conocimiento institucional.* [En línea] ACIMED v.20 n.5 Ciudad de La Habana nov. 2009, Nov de 2009. [Citado el: 11 de febrero de 2013.] http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352009001100006. ISSN 1561-2880.

6. **Arjonilla Domínguez, Sixto Jesús y Medina Garrido, José Aurelio .** *La gestión de los sistemas de información en la empresa.* España : s.n., 2002. ISBN: 84-368-1632-3.

7. **Ocuco.** OCUCO. *Software Whit Vision.* [En línea] © Ocuco, 2012. [Citado el: 13 de febrero de 2013.] http://www.ocuco.es/index.html.

8. **Netsys Computación.** OPTIVEN. *Sistema de Gestión Óptica y Optométrica/ Oftalmológica.* [En línea] 2012. [Citado el: 12 de febrero de 2013.] http://www.netsys.com.ve/OPTIVEN/index.html.

9. **Solinsur.** software selección. *OPTISOF.* [En línea] © www.softwareseleccion.com, 2009. [Citado el: 19 de febrero de 2013.] http://www.softwareseleccion.com/optisof-p-2606.

10. **Braña, Francisco.** Portal Programas. *Optisof.* [En línea] RedAccenir, S.L., 2003 - 2013. http://gratis.portalprogramas.com/Optisof.html.

11. **Softonic.** Softonic enjoy software! *Pyme Manager Opticas.* [En línea] Copyright INTERSHARE, S.L. , 1997-2013. [Citado el: 6 de marzo de 2013.] http://pyme_manager_opticas.softonic.com/.

12. **Academia, Real.** Real Academia Española. [En línea] 2013. [Citado el: 11 de febrero de 2013.] http://lema.rae.es/drae/?val=metodolog%C3%ADa.

13. **Kroll, Per y Kruchten, Philippe.** *The rational unified process made easy.* Boston : Addison Wesley, 2003, 2003. ISBN: 0-321-16609-4.

14. **Rumbaugh, James, Booch, Grady y Jacobson, Ivar.** *El lenguaje unificado de modelado.* Madrid : Addison Wesley, 2000, 2000. ISBN: 84-7829-037-0.

15. **Miers, Derek y White, Stephen A.** *BPMN Modeling and Reference Guide: Understand and Using BPMN.* Florida, USA : Publisher´s Cataloging-in- Publication Data, 2008. ISBN 978-0-9777527-2-0.

16. **J. Evans, Ian y Ball, Jennifer.** Oracle. *Your First Cup: An Introduction to the Java EE Platform.* [En línea] Copyright © Oracle and/or its affiliates., august de 2010. [Citado el: 18 de febrero de 2013.] http://download.oracle.com/javaee/5/firstcup/doc/..

17. **Worsley, John y Drake, Joshua.** *Practical PostgreSQL.* s.l. : Copyright © 2002 Command Prompt, Inc, 2002. ISBN: 1-56592-846-6.

18. **Geschwinde, Ewald.** *PostgreSQL. Developer´s Handbook.* California, EEUU : Copyright © 2002 by Sams Publishing, 1996. ISBN:0-672-32260-9.

19. **Keegan, Patrick.** *NetBeans IDE field guide.* New Jersey : Upper Saddle River : Prentice Hall, 2006, 2006. ISBN: 0-13-239552-5.

20. **Oracle Corporation.** Web y JavaEE. *NetBeans IDE Features.* [En línea] © Oracle Corporation and/or its affiliates, 2012. [Citado el: 12 de febrero de 2013.] http://www.netbeans.org/features/web/index.html.

21. **Brito, Nacho.** *Manual de desarrollo Web con GRAILS.* España : s.n., 2009. ISBN:978-84-613-2651.

22. **M. Judd, Christopher y Nusairat, Joseph Faisal.** *Beginning Groovy and Grails : from novice to professional.* California : Berkeley : APress, cop.2008, 2008. ISBN: 978-1-4302-1045-0.

23. **GLEN, SMITH y LEDBROOK, PETER.** *Groovy in Action.* New York : Manning Publications. Co, 2009. ISBN- 978-1-933988-93-1.

24. **Downey, Tim.** *Web development with Java.* London : London : Springer, cop., 2007. ISBN: 978-1-84628-862-3.

25. **Torres, Arcos y Mauricio, Christian.** Escuela Politécnica Nacional. *Generación dinámica de reportes basado en IREPORT & JASPERREPORT bajo plataforma J2EE.* [En línea] marzo de 2007. [Citado el: 13 de marzo de 2013.] http://bibdigital.epn.edu.ec/handle/15000/329.

26. **Swenson, Erik.** "Reports made easy with JasperReports". *JavaWorld.com.* [En línea] Copyright © Infoworld, Inc., 2006-2013. [Citado el: 12 de febrero de 2013.] http://www.javaworld.com/javaworld/jw-09-2002/jw-0920-opensourceprofile.html.

27. **S. Davis, William y Mata, Antonio.** *Herramientas CASE.* La Habana : Thomson-Paraninfo, 1992. ISBN: 84-283-1927-8.

28. **Visual Paradigm International Ltd.** *Visual Paradigm International Ltd.* [En línea] 2007. [Citado el: 12 de febrero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/..

29. **García Molina, Jesús, y otros, y otros.** msdn. *Crear y usar reglas de negocios.* [En línea] © 2013 Microsoft, 23 de abril de 2013. [Citado el: 12 de mayo de 2013.] http://msdn.microsoft.com/es-es/library/aa577691%28v=bts.10%29.aspx.

30. **Michael Dobson, Joe Luttrell .** LESSONS FROM HISTORY. *Functional versus Non-Functional Requirements and Testing.* [En línea] Copyright ©2001-2013 Mark Kozak-

Holland, 1 de febrero de 2013. [Citado el: 10 de marzo de 2013.] http://www.lessons-from-history.com/node/83.

31. **García Molina, Jesús, Ortín, M. José y Toval Alvarez, José Ambrosio.** *De los Procesos del negocio a los Casos de Uso.* Sevilla,España : Ciencia y Técnica Administrativa, 2007. ISSN-e 1666-1680.

32. **Judd, Christopher M. y Faisal Nusairat, Joseph.** *Beginning Groovy and Grails: from novice to professional.* California : Berkeley : APress, cop.8, 2008. ISBN: 978-1-4302-1045-0.

33. *Architectural patterns regarding web application domain usability.* **Arciniegas Herrera, José Luís, y otros, y otros.** págs. 52-55, Universidad Nacional de Colombia : s.n., 1980, Vols. Vol. 30, Nº. 1, 2010. ISSN: 0129-5608.

34. **Larman, Craig.** *UML y Patrones. Una introducción al análisis y diseño orientado a objetos y al proceso unificado.* España : México : Prentice-Hall, cop. 1999, 1999. ISBN: 970-17-0261-1.

35. **Aedo Cuevas, Ignacio.** *Ingenieria De La Web Y Patrones De Diseño.* Madrid : Madrid, (etc.) : Prentice Hall, cop. 2005, 2005. ISBN: 84-205-4609-7.

36. **LARMAN, CRAIG .** *UML Y PATRONES. UNA INTRODUCCIÓN AL ANÁLISIS Y DISEÑO ORIENTADO A OBJETOS Y AL PROCESO UNIFICADO. SEGUNDA EDICIÓN.* Madrid : PEARSON EDUCACIÓN, S.A., 2003. ISBN: 84-205-3438-2.

37. **Pressman, Roger S.** *Ingeniería del Software. Un enfoque Práctico.* Sexta Edición. Madrid : McGraw-Hill, 2006. ISBN: 970-10-5473-3.

38. **Bolaños Alonso, Daniel, Sierra Alonso, Almudena y Alarcón Rodríguez, Idoia.** *Pruebas de software y JUnit.* Madrid : Pearson Educación, 2008. ISBN: 978-84-8322-354-3.

39. **SMITH, GLEN y LEDBROOK, PETER.** *Grails in Action.* New York : Manning Publications Co., 2009. ISBN 978-1-933988-93-1.

40. **Julián Gómez.** Laboratorio de las TI. *Método de Estimación Puntos Casos de Uso (Use Case Points).* [En línea] Copyright © , 14 de febrero de 2013 . [Citado el: 15 de mayo de 2013.] http://www.laboratorioti.com/2013/02/14/metodo-de-estimacion-puntos-casos-de-uso-use-case-points/.

41. **Trinchet Soler, Rafael y Pedrianes Vigo, Margarita.** *Origen, estado actual y perspectivas de la Red Nacional de Cirugía Pediátrica.* La Habana : s.n., 2004. ISSN 1024-9435, ISSN-e 1530-2880.

42. *Pruebas unitarias en Java.* **Aranda Crespo, Oscar.** págs. 20-26, Madrid : Red de Bibliotecas Universitarias (REBIUN), 2006, Vol. Nº 134. ISSN: 1134-4792.

43. *Ingeniería del Software y Pruebas.* **Aguiar Fernández, María del Mar.** págs. 90-96, España : Ediciones Técnicas REDE, 2010, Vol. Nº 550. ISSN 0482-6396.

Bibliografía

44. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000.** *El Proceso Unificado de Desarrollo del Software.* [trad.] Salvador Sánchez, y otros. Español. Madrid : Addison Wesley, 2000. pág. 464. ISBN: 84-7829-036-2.

45. **Nacho, Brito. 2009.** Manual de desarrollo web con Grails. [En línea] Junio de 2009. http://www.manual-de-grails.es. v1.0.4.

46. **Pressman, Roger S. 2001.** *Ingeniería del Software. Un enfoque Práctico.* [trad.] Darrel Ince. 5ta Edición. s.l. : Mc Graw Hill, 2001. pág. 614.

47. **M. Judd, Christopher, Faisal Nusairat, Joseph y Shingler, James. 2008.** *Beginning Groovy and Grails: From Novice to Professional.* United States of America : s.n., 2008. ISBN-13 (pbk): 978-1-4302-1045-0.