Universidad de las Ciencias Informáticas

Facultad 7

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

Título: Cálculo de Lentes Intraoculares del módulo Consulta Externa en el sistema alas HIS

Autora: Yenisley Montelier Ortiz

Tutor: Ing. Alexander Rodríguez Rabelo

Cotutor: Ing. Ruber González Pedraza

La Habana, junio 2013

"Año 55 de la Revolución"

SÍNTESIS DEL TUTOR:

Ing. Alexander Rodríguez Rabelo: Graduado de ingeniero en Ciencias informáticas en el año 2007, profesor Asistente. Se ha desempeñado como jefe de proyecto, jefe de departamento y vicedecano de investigación y postgrado y actualmente se desempeña como director del Centro de Informática Médica (CESIM). Ha impartido las asignaturas de Matemática, Práctica Profesional, Debate Histórico Contemporáneo y Metodología de la Investigación Científica. Ha participado en los proyectos de desarrollo de software alas BQO y alas HIS donde se ha desempeñado como implementador, analista, administrador de la configuración y jefe de proyecto.

Correo electrónico: arodriguezra@uci.cu

Ing. Ruber González Pedraza: Instructor recién graduado en el año 2011 de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Profesor vinculado a la Facultad 7 y miembro del Departamento de Sistemas de Gestión Hospitalaria.

Correo electrónico: rpedraza@uci.cu

AGRADECIMIENTOS

Gran parte de lo mejor que hay en nosotros está ligado a nuestro amor a la familia, esta nos proporciona valores que quedan para toda la vida. Hoy, he llegado a alcanzar mis metas gracias a mi familia, es por eso que comenzaré agradeciéndole a mi familia gigantesca.

Agradecer a mis padres por el apoyo incondicional, por los buenos valores inculcados y la buena educación. A mi madre por hacer el papel de madre y padre a la vez en ocasiones adversas, por darme un consejo, una caricia, un beso y sobre todo estar siempre para mí cuando lo necesito y ayudarme en el camino de la vida, por levantarme cuando me caí y por aprobar todas mis decisiones por muy erróneas que fuesen, gracias a ella soy lo que soy. A mi padre por su silencio, por su apoyo, por sus consejos, por su carácter y por enseñarme a ser mejor cada día, por darme luz y por ser mi ángel de la guarda en todo momento.

A mis hermanas: A Yeney por enseñarme a crecer y darme seguridad, por protegerme y no querer que sufra, a Yasendry (TATA) por demostrarme que hay que sobreponerse ante todo, por demostrarme valentía ante la vida.

A mi segundo padre, mi padrastro Juani por malcriarme, quererme, ayudarme, apoyarme, consentirme, por cuidar de mí y de mi familia, por ser simplemente como es.

A todas mis tías, tíos y abuelos: a Mileidis mi segunda madre por estar ahí siempre que lo necesito, por quererme como una hija más, a mis tías de trinidad por creer en mí y apoyarme desde lejos, a mi tía Damaris por impulsarme, a mi tío Osmaldo por quererme como una hija y creer en mí, a todos por confiar en mí.

A todos mis primas: a Yusleidys por ser mi hermana cuando no las tenía, por ser amiga y consejera, a Yailer por ser mi hermano varón el que no tengo y defenderme de todo desde pequeñitos, a Yariolis por ser prima-hermana, doctora, psicóloga y amiga, a todos por quererme siempre.

A mis sobrinos por darme la oportunidad de participar en su infancia, y en momentos importantes en su vida, por vivir con ellos momentos maravillosos y a mi cuñado Pastor por darme apoyo y confiar en mí.

A mi familia de amigos, a los de lactante, esos amigos de siempre: Maray por ser como mi hermana, a Oni, Oricel, Yanetsy, Lidiela, Armando, Alán, Andy, Yoan, y a los de lejos que hoy no están pero significaron mucho en mi formación Yanela,Diego, Heikel. A los amigos de hoy a Galia, Einara, Anilys, Eliza, hoy todos somos lo que quisimos, ser profesionales y útiles para la vida.

A amigos especiales como Dunia, Janeth, Diana, Maidelin, Aliet, Garkin, Eleidis que han sabido darme apoyo, consejos, han estado ahí conmigo en estos seis años de carrera universitaria, han compartido conmigo alegrías y tristezas y sobre todo se han comportado como mi propia familia.

A mis compañeros de grupo, el viejo y el nuevo, especialmente a Hiram, Teddy, Baby por los buenos consejos, a todos por los buenos momentos, a Lily por su paciencia, su ayuda, los buenos consejos y momentos, a Yeni, Betty, Jorge.

A mi novio Luis Ángel por su paciencia, su comprensión, por creer en mí, por darme fuerzas, por cuidarme, consentirme, mimarme y sobre todo por impulsarme cada día a seguir adelante cuando pensé que no podía.

A los profesores que de una forma u otra influyeron en mi formación: A Rosa Elena, William, Rita María, Maude, a la Chiqui, al Plata, a Marichu, a Yanersy, Pastor, Luis Mariano, Alexander, a todos gracias por su ardua labor y comprensión.

A las personas que día a día me impulsan a seguir adelante, personas que llegan a tu vida para ayudarte a crecer, a amigos que siempre estaré eternamente agradecida pues me han ayudado en esta larga carrera y han estado ahí siempre para mí con un buen consejo, un abrazo o simplemente una linda sonrisa, a ellos gracias por creer en mí.

A mis vecinos por tenerme como una más en su familia, especialmente a Juliana, Miguel, Ana gloria, Eliza, Migdilio, Alito, Miriela, Alexey, Ailin, Geidy, Tito, Borys, María, Lázaro.

Un agradecimiento especial al Comandante Fidel por tener la maravillosa idea de crear esta Universidad de excelencia y haberme dado la oportunidad de crecerme como profesional y ser útil para la vida.

DEDICATORIA

Dedico este trabajo a mi familia.

A mis padres por la dedicación, el apoyo y la buena educación que me han dado.

A mis sobrinos por ser esa luz que me inspira y me ayuda a seguir adelante.

A las personas que no están y han caminado junto a mí en todo.

RESUMEN

El Sistema de Información Hospitalaria alas HIS está compuesto por diferentes módulos, integrados entre sí, que comprenden a la mayoría de las áreas de una institución hospitalaria. En el módulo consulta externa se gestiona la información que se genera en la atención al paciente de forma especializada, sin embargo, el servicio de Oftalmología no cuenta con la implementación de los procesos vinculados al cálculo de Lentes Intraoculares (LIO) necesario en las intervenciones quirúrgicas del cristalino. El objetivo del presente trabajo de diploma es implementar las funcionalidades necesarias para gestionar el cálculo de los Lentes Intraoculares en el módulo Consulta Externa del Sistema de Información Hospitalaria alas HIS.

Para el desarrollo de las funcionalidades se asimiló la arquitectura definida por el departamento Sistemas de Gestión Hospitalaria, empleándose como patrón arquitectónico el Modelo Vista Controlador. Se utilizó Java como lenguaje de programación orientado a objetos, Eclipse como Entorno Integrado de Desarrollo, PostgreSQL como gestor de base de datos, Hibernate como herramienta de mapeo objeto relacional (ORM) para la persistencia de los datos y el JBoss Seam como Framework de integración.

Con la inclusión de las funcionalidades al Sistema de Información Hospitalaria se realizará el cálculo de los Lentes Intraoculares a partir de las fórmulas y criterios existentes hasta el momento. Además permitirá que los especialistas puedan introducir nuevas fórmulas y criterios de trabajo así como personalizar las constantes y los parámetros utilizados en los mismos, elevando los resultados en las operaciones oftalmológicas de implante de Lentes Intraoculares.

Palabras clave: constantes, consulta externa, criterios de trabajo, fórmulas, lentes intraoculares, Oftalmología, Sistema de Información Hospitalaria.

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El conocimiento ocupa sin lugar a dudas un lugar especial en la sociedad contemporánea, pues resulta de vital importancia su aprovechamiento y aplicación. A través de las distintas ciencias aplicadas han surgido un sin número de avances científicos y tecnológicos que permiten ampliar la innovación y el desarrollo en varios sectores de la sociedad.

El campo de la salud, es en la actualidad, uno de los más beneficiados con estos avances alcanzados, entre los que se encuentran; la aparición de nuevas vacunas, descubrimientos y confección de modernas técnicas y equipos médicos para la emisión de diagnósticos muy eficaces para los pacientes. Por otra parte es necesario destacar la unión indisoluble que en los últimos años tienen la medicina y las ciencias de la computación, dando lugar a la Informática Médica que se define como: “La disciplina que estudia los métodos y medios que permiten estructurar, recepcionar, representar y emitir la información médica-científica de forma automatizada.” (1)

Con la Informática Médica, aparecen un conjunto de componentes informáticos que contribuyen a mejorar los servicios que se brindan en las entidades médicas, entre los más destacados se encuentran: el envío de imágenes de Tomografía Axial Computarizada (TAC) y Ultrasonidos (US), la realización de consultas remotas en tiempo real o diferido, así como las consultas de segunda opinión por parte de especialistas para futuras investigaciones. El surgimiento de los Sistemas de Información Hospitalaria (HIS, por sus siglas en inglés), constituye el punto de partida para la gestión de la información de esos servicios, dando lugar a una excelente interrelación de todos los datos existentes de un modo eficiente y sencillo para facilitar la gestión de un centro hospitalario, con independencia de su tamaño y de los servicios que se brinden. (2)

Los HIS, se ajustan a las necesidades de la institución hospitalaria en que se va a implementar, permitiéndoles a los profesionales de la salud la informatización de sus operaciones básicas, mejorando considerablemente los procesos de asistencia médica. Es así como en los últimos años se acrecienta la producción de software y la creación de centros de desarrollo para la salud; como es el caso del Centro de Informática Médica (CESIM), perteneciente a la Universidad de las Ciencias Informáticas (UCI), que desarrolla el Sistema de Información Hospitalaria alas HIS, para gestionar la información referente, tanto

en la atención al paciente como en la labor administrativa, teniendo en cuenta las áreas con que cuente la institución.

El sistema alas HIS está compuesto por diferentes módulos relacionados entre si, en el módulo Consulta Externa se maneja la información que se genera en la atención al paciente de forma especializada. Dentro de los servicios que ofrece se encuentra el de Oftalmología, pero en la consulta de microcirugía del Cristalino es necesario gestionar el cálculo del Lente Intraocular (LIO), siendo este, el lente permanente que es implantado de forma quirúrgica en lugar del cristalino del ojo del paciente. Las medidas obtenidas del globo ocular son de ayuda para determinar el poder dióptrico apropiado del Lente Intraocular y obtener la refracción deseada. Estas medidas incluyen el poder central refractivo de la córnea, longitud axial, el diámetro horizontal de la córnea, profundidad de la cámara anterior y grosor del cristalino. La exactitud en predecir el poder necesario del lente está directamente relacionada a la exactitud de estas medidas. (3)

Desde hace varias décadas, los medios de cómputo han auxiliado a los profesionales de la salud en la realización de estos cálculos, por lo que el trabajo se ha automatizado en diversas ocasiones, obteniéndose productos que realizan los cálculos del poder dióptrico del Lente Intraocular automáticamente. Sin embargo, durante los últimos años la forma para realizar el cálculo del poder dióptrico de los Lentes Intraoculares ha variado considerablemente, pues existen diferentes fórmulas de cálculo y criterios para utilizar las mismas según la variación de los parámetros de medida del globo ocular, las cuales evolucionan vertiginosamente y a su vez se transforman.

Antes de realizar el cálculo del lente, el especialista realiza los estudios oftalmológicos al paciente para obtener las mediciones de queratometría y longitud axial. Evidentemente este proceso requiere de un gran esfuerzo y precisión, sin embargo estas actividades, se realizan por el personal médico de forma manual y no están soportadas por un sistema informático, lo que conlleva a dificultades en el trabajo y gran probabilidad de error en la toma de las medidas preoperatorias. En estos casos, pueden ocurrir errores refractivos postoperatorios, lo que no asegura que el paciente obtenga la refracción deseada luego de la cirugía y con ella cambios refractivos importantes. Además, la calidad de la intervención quirúrgica se ve afectada por la reiteración de exámenes que el paciente debe hacerse nuevamente, inevitables y necesarios para obtener los parámetros exactos y que no existan equivocaciones en el momento de

realizar el cálculo del Lente Intraocular y una vez implantado lograr una rehabilitación visual satisfactoria en el paciente.

Por la problemática anteriormente expuesta se determina como **problema a resolver** de la presente investigación: ¿Cómo calcular los Lentes Intraoculares relacionados con la cirugía del cristalino?

El problema definido está enmarcado en el **objeto de estudio** dirigido a la gestión del cálculo de Lentes Intraoculares en las consultas oftalmológicas. El **campo de acción** está centrado en la gestión del cálculo de Lentes Intraoculares en las consultas oftalmológicas.

Para resolver el problema identificado se propone el siguiente **objetivo general:** Desarrollar las funcionalidades identificadas por el analista para el cálculo de Lentes Intraoculares en el módulo Consulta Externa del Sistema de Información Hospitalaria alas HIS.

Para darle cumplimiento al objetivo propuesto, así como controlar y evaluar el proceso investigativo, se proponen como **tareas de la investigación**:

1. Valorar los sistemas informáticos existentes en el ámbito nacional e internacional, asociados al cálculo de Lentes intraoculares.
2. Asimilar la arquitectura definida por el Departamento Sistemas de Gestión Hospitalaria para el desarrollo de sus aplicaciones.
3. Valorar la propuesta del diseño hecha por el analista para la implementación de los procesos del cálculo de Lentes Intraoculares.
4. Implementar aplicando las pautas de diseño y guiándose por el modelo de caso de uso del módulo, los subprocesos: realizar cálculo de Lentes Intraoculares, personalizar fórmulas para el cálculo de Lentes Intraoculares, personalizar criterio para el cálculo del Lentes Intraoculares.

Para realizar la presente investigación se utilizan los **métodos científicos de investigación**, siendo estos:

Métodos teóricos

Estos métodos permiten estudiar las características del objeto de investigación que no son observables directamente, facilitan la creación de modelos y posibilitan el conocimiento del estado del arte del fenómeno, su evolución en una etapa determinada y su relación con otros fenómenos.

- ✓ **Analítico – sintético**

El empleo del método analítico-sintético permite analizar las teorías, los documentos sobre la investigación, para obtener los elementos más importantes que se relacionan con el cálculo de Lentes Intraoculares en las consultas oftalmológicas de las instituciones hospitalarias.

- ✓ **Histórico-lógico**

El método histórico-lógico permite comprobar teóricamente el desarrollo evolutivo en el estudio de los sistemas informáticos, las herramientas, etc, para el desarrollo de las funcionalidades para gestionar el cálculo de Lente Intraocular.

- ✓ **Modelación**

El modelo permite la creación de modelos (propuestas, alternativas, estrategias, etc.). Se utiliza en la representación de los modelos que se utilizan en la investigación, como son: el modelo de despliegue, diseño y datos.

Métodos empíricos

Estos métodos describen y explican las características del objeto, permite aproximarse al conocimiento del objeto mediante conocimientos directos y el uso de la experiencia.

- ✓ **Observación**

Este método puede utilizarse en cualquier momento de la investigación, permitiendo investigar el objeto en su manifestación externa. Además permite la recogida de información de cada uno de los conceptos definidos en la investigación.

Con la inclusión de las funcionalidades al sistema se espera obtener los siguientes **beneficios**:

1. Garantizar que la realización del cálculo de Lentes Intraoculares mediante el sistema alas HIS no caduque si se obtuviera nuevas fórmulas científicas para realizar el mismo.
2. Mejorar el proceso del cálculo del Lente Intraocular que se le implantará al paciente en la consulta de Microcirugía del Cristalino.
3. Permitir la evolución de las fórmulas utilizadas en el cálculo del Lente Intraocular desde módulo Consulta Externa del sistema alas HIS.
4. El desarrollo de los procesos: personalizar fórmulas y criterios así como realizar cálculo del Lente Intraocular contribuirá a la obtención de resultados más exactos en la implantación del Lente, aumentando la eficiencia de las acciones que tienen lugar en el área de Oftalmología de las instituciones hospitalarias.

Este documento se encuentra estructurado en cuatro capítulos, de la siguiente manera:

Capítulo 1: Fundamentación teórica de los procesos relacionados con el cálculo del Lente Intraocular; se realiza un estudio sobre los sistemas de información hospitalaria que gestionan los procesos asociados al cálculo del Lente Intraocular en cuba y el mundo. A partir de la arquitectura definida por el Departamento Sistemas de Gestión Hospitalaria se fundamentan las herramientas y tecnologías de desarrollo a utilizar en la investigación.

Capítulo 2: Descripción de la arquitectura de software del Sistema de Información Hospitalaria alas HIS; se detallan los requerimientos no funcionales y la propuesta de arquitectura del sistema; además se hace un análisis de posibles implementaciones, componentes o módulos ya existentes que pueden ser rehusados, así como las estrategias de integración, estándares y estilos de codificación a utilizar.

Capítulo 3: Descripción y análisis de la solución propuesta; se hace una valoración crítica del diseño propuesto por el analista. Se refinan los diagramas de clases del diseño y diagramas de interacción y se describen las nuevas clases, u operaciones necesarias.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA DE LOS PROCESOS RELACIONADOS CON EL CÁLCULO DE LOS LENTES INTRAOCULARES

En el presente capítulo se realiza una descripción de los conceptos más utilizados en la investigación para un mayor entendimiento de la misma. Se describen varios de los sistemas informáticos que tienen relación con el cálculo de Lentes Intraoculares, dando a conocer características técnicas de los mismos, además se explican las tecnologías, metodologías y herramientas que se van a utilizar en el desarrollo de la investigación, a partir de la arquitectura definida para el Sistema de Información Hospitalaria alas HIS.

1.1 Conceptos básicos relacionados con el dominio del problema

La oftalmología es la especialidad médica que ha desarrollado conocimientos, técnicas y habilidades especializadas en la atención de los ojos. Tratando temas como la fisiología normal de los ojos, los acontecimientos y las enfermedades que pueden afectar el normal funcionamiento de la vista, además de ciertos tratamientos y procedimientos útiles para el cuidado de los ojos y actividades de prevención de futuros problemas. (4)

El **especialista oftalmológico** es el médico especializado en Oftalmología.

El **poder dióptrico** es el poder de refracción de una lente óptica, medido en dioptrías.

Los **lentes intraoculares** (LIO) son pequeñas lentes elaboradas con un material sintético, que se implantan en el ojo para reemplazar al cristalino cuando las cataratas provocan visión nublada o si el cristalino perdió flexibilidad por la edad y no puede hacer foco para las tareas de visión. El tipo de lente y el poder dióptrico del lente son determinados por el especialista a partir de un conjunto de reglas y fórmulas preestablecidas con este fin. (5)

El **globo ocular** es el órgano esférico que posibilita la visión, y que permite captar los colores y formas del entorno, por medio de los estímulos luminosos. (6)

El **cristalino** es un componente del ojo con forma de lente biconvexa que está situado tras el iris y delante del humor vítreo. Su propósito principal consiste en permitir enfocar objetos situados a diferentes distancias. (7)

La **refracción ocular** es la refracción de la luz cuando atraviesa las distintas estructuras oculares como la córnea, el cristalino, el humor acuoso y el humor vítreo.

1.2 Antecedentes

En la actualidad existen diversas soluciones informáticas destinadas a mejorar la calidad en la prestación de servicios de asistencia médica. Como parte de la investigación se identificaron algunos sistemas informáticos desarrollados a nivel mundial relacionados al área de Oftalmología, o que contengan soluciones para la gestión del cálculo de Lentes Intraoculares. A continuación se muestran las principales características técnicas de los sistemas valorados.

1.2.1 Sistemas Nacionales

Sistema para el perfeccionamiento de la microcirugía ocular (SIPMO) es un sistema basado en el cálculo del astigmatismo inducido por la cirugía de catarata. La aplicación permite investigar las distintas variables que influyen en la microcirugía de la catarata en pacientes con y sin implantación de Lentes Intraoculares. Está implementado en Lenguaje C y Clipper 87 que utiliza las fórmulas de Binkhorst y Fyodorov, conectado a una base de datos DBSE III que almacena los datos de los pacientes con posibilidades de realizar un análisis rápido y exacto de todos los pacientes operados de catarata de cualquier etiología en el Hospital Docente Clínico-quirúrgico Hermanos Ameijeiras. (8)

Sistema Automatizado de Microcirugía (SAMC) es una solución diseñada para el Hospital Oftalmológico “Ramón Pando Ferrer” de La Habana. Este sistema tiene entre sus principales funcionalidades; la gestión de la información relacionada con el pre y post operatorio del servicio de Microcirugía del Cristalino, así como el control y cálculo de los Lentes Intraoculares que se lleva a cabo en el mismo. Está implementado en "Clipper'85 Winter" y herramienta líder de desarrollo bajo MS-DOS de aplicaciones relacionadas con bases de datos, lo que dificulta su mantenimiento y actualización. (9)

Bloque Quirúrgico Oftalmológico (BQO) es un sistema desarrollado por la Universidad de las Ciencias Informáticas en conjunto con el Instituto de Oftalmología “Ramón Pando Ferrer” para instituciones especializadas en Oftalmología. Entre sus funcionalidades se encuentra la gestión del cálculo y control de Lentes Intraoculares en la consulta de Microcirugía del Cristalino. Está desarrollado sobre una arquitectura

en tres capas, como gestor de base de datos PostgreSQL liberado bajo la licencia BSD, la metodología de desarrollo utilizada fue el Rational Unified Process (RUP) y el lenguaje unificado de modelado (UML), utilizando como servidor de aplicaciones Internet Information Service y el framework .net 2.0. (10)

1.2.2 Sistemas Internacionales

Emetropía es un software integral de Oftalmología que permite la gestión completa de una clínica de oftalmología. La aplicación está desarrollada en Visual Basic, la cual presenta un sistema de datos cliente/servidor, con un entorno amigable de Ventanas, compatible con versiones desde Windows 98 hasta Windows Vista. Utiliza Base de Datos estándar y robusta Microsoft SQL Server (Versión Profesional). Entre sus funcionalidades se encuentra la gestión de Lentes Intraoculares. (11)

VisionDat es un software de tecnología avanzada que ofrece diversas opciones en el manejo de consultorios oftalmológicos, el mismo no cubre la totalidad de las funcionalidades necesarias ya que recoge solamente los datos generales, estudios oftalmológicos, la historia clínica y estudios refractivos donde se recoge la gestión de Lentes Intraoculares. Este software se encuentra bajo licencia propietaria, no posee los servicios que necesitamos y solo podrá ser implantado en máquinas que utilicen sistema operativo Windows. Todos los ordenadores donde se instale deben tener obligatoriamente de memoria RAM como mínimo 1GB y en el mismo su rendimiento no es el más óptimo. (12)

Oftalclinic es un sistema de gestión integral de clínicas oftalmológicas que abarca todas las áreas de planificación, administración, gestión médica y control destinadas a optimizar los recursos humanos, permitiendo la conexión de equipos de diagnóstico computarizados. Emplea tecnología cliente-servidor sobre base de datos Microsoft SQL-Server de alto rendimiento, lo que gracias a esto, se pueden interconectar varias clínicas centralizando así toda la información. El sistema está diseñado como una aplicación de escritorio.Contiene módulos de cirugía oftalmológica general donde se gestionan los siguientes procesos: cirugía de catarata con facoemulsificación, cirugía de pterigium con injertos, cirugía láser para retinopatía diabética, trabeculectomía para glaucoma y cirugía refractiva. (13)

Al realizar un estudio sobre los diferentes productos existentes en Cuba y el mundo se arriba a la siguiente conclusión:

El análisis de los sistemas estudiados permitió, a partir de sus características y funcionalidades, asimilar aquellos que ofrecieran una solución al problema a resolver. Estos sistemas están desarrollados para gestionar la información de instituciones oftalmológicas. Entre sus funcionalidades permiten gestionar el cálculo del Lente Intraocular pero la gran mayoría de estos sistemas no le brinda al especialista la posibilidad de introducir nuevas fórmulas y criterios así como personalizar las constantes utilizadas en los mismos, además no cuentan con posibilidades de expansión hacia otras especialidades o áreas específicas de la institución hospitalaria. Sin embargo, el sistema alas BQO cumple con las funcionalidades requeridas para gestionar los procesos mencionados anteriormente pero no cuenta con funcionalidades requerida para gestionar los datos del paciente, por lo que necesita integrarse a través de servicios web con el sistema Galen, desarrollado por la empresa Softel, haciendo la implantación del sistema alas BQO dependiente de este, y por consecuencia más costosa. Además, no gestiona la información referida al resto de las áreas hospitalarias, como almacén, y farmacia, necesarias para el consumo de productos y servicios durante las cirugía del Cristalino; y tiene la desventaja que algunas de las tecnologías de desarrollo utilizadas están bajo licencias privativas.

1.3 Técnicas, tecnologías y metodologías actuales en las que se apoya la solución del problema.

A partir del requisito principal de integrar las funcionalidades relacionadas al cálculo de Lentes Intraoculares del módulo Consulta Externa al Sistema de Información Hospitalaria alas HIS, se acometió el estudio de la arquitectura de este sistema, así como las tendencias tecnológicas que permiten cumplir el objetivo propuesto. En consecuencia con lo anterior, se describen los elementos principales de esta arquitectura y las tecnologías empleadas.

1.3.1 Arquitectura

Arquitectura Cliente-Servidor

El sistema alas HIS se basa en la arquitectura cliente-servidor lo que permite que sea de fácil acceso desde diferentes ubicaciones. Esta es un modelo en el que las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos.

Básicamente este esquema consiste en un programa cliente que hace peticiones a otro programa (el servidor) que le brinda una respuesta. Esta idea es aplicable a programas que se ejecutan en una misma computadora, aunque resulta más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras. (14)

La separación entre el cliente y el servidor es de tipo lógico. El servidor puede ejecutarse en más de una máquina y puede estar constituido por más de un programa. Los servidores pueden ser de diferentes tipos, por ejemplo, pueden ser servidores de archivos, servidores de correo o servidores web; cada uno con propósitos diferentes, pero con la misma arquitectura básica. (15) Si el servidor se descompone en diferentes programas que pueden ser ejecutados por diferentes computadoras. (16)

Arquitectura en capas

La arquitectura en capas es un estilo de diseño cuyo objetivo principal es la separación y agrupamiento de los componentes del software atendiendo a la función que cumplen en el mismo. Para realizar el asociamiento se tienen en cuenta las funcionalidades relacionadas con el usuario del sistema, así como la información que éste gestiona y las operaciones que realiza sobre la misma, en dependencia de la complejidad que se necesita que tenga el sistema. La división se realiza en tres capas: la capa de presentación, la capa de negocio y la capa de datos. (17)

Patrón Modelo-Vista-Controlador (MVC)

Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. (18) Básicamente se utiliza para separar el código de la página HTML (siglas de HyperText Markup Language en inglés) lo máximo posible y poder reutilizar componentes fácilmente.

Modelo: Es el sistema de gestión de base de datos y la lógica de negocio. Las clases del modelo de clases contendrán funciones para consultar, insertar y actualizar información de la base de datos.

Vista: Es la página HTML y el código que provee de datos dinámicos a la página.

Controlador: Es el responsable de recibir los eventos de entrada desde la vista. (19)

1.3.2 Tecnologías utilizadas en el proceso de desarrollo

La arquitectura en capas y el patrón MVC, pueden relacionarse mediante cada uno de sus elementos, donde la capa de presentación podría corresponderse con la Vista, la capa de negocio con el Controlador y la capa de datos con el Modelo. A continuación se explican las tecnologías a utilizar en el desarrollo de las funcionalidades previstas, agrupadas por cada uno de los elementos del patrón descrito. (20)

Java

Java es un lenguaje de programación diseñado desde un principio orientado a objetos. Permite la ejecución de un mismo programa en múltiples sistemas operativos y ordenadores. Es un lenguaje compilado, pues genera ficheros de clases compiladas, las cuales son interpretadas por la máquina virtual de java que mantiene el control sobre las que se estén ejecutando. Permite programar aplicaciones web dinámicas, con acceso a bases de datos, utilizando el Lenguaje de Marcado Extensible (XML, en inglés), con cualquier tipo de conexión de red entre cualquier sistema. (21)

Capa de Presentación

La capa de presentación se encarga de proveer una interfaz entre el sistema y el usuario. Fundamentalmente, se responsabiliza de que se le comunique información al usuario por parte del sistema y viceversa, manteniendo una comunicación con la capa de negocio. (20)

Java Server Faces (JSF)

Java Server Faces (JSF) es un framework Java, que permite crear Interfaces de Usuario (UI) para aplicaciones web, mediante componentes reutilizables. Permite el manejo de estados y eventos, así como la asociación entre los datos de la interfaz y los datos de la aplicación web. (22)

RichFaces

RichFaces es una librería de componentes web enriquecidos, de código abierto y basada en el estándar JSF. Provee facilidades de validación y conversión de los datos proporcionados por el usuario y administración avanzada de recursos como imágenes, código Java Script, entre otros. Permite crear UI modernas de manera eficiente y rápida, altamente configurables en cuanto a temas y esquemas de

colores predefinidos por el propio framework o desarrollados a conveniencia por el usuario, lo que mejora la experiencia de este. (23)

Ajax4JSF

Ajax4jsf es una librería de código abierto que se integra totalmente en la arquitectura de JSF, y extiende la funcionalidad de sus etiquetas dotándolas con tecnología Ajax de forma limpia y sin añadir código JavaScript. Con este se puede variar el ciclo de vida de una petición JSF, recargar determinados componentes de la página sin necesidad de recargarla por completo y realizar peticiones automáticas al servidor. Ajax4jsf permite dotar a cada aplicación JSF de contenido mucho más profesional con muy poco esfuerzo. (24)

Facelets

Facelets es un framework para plantillas, en donde es posible diseñar de forma libre una página web y luego asociarle los componentes JSF específicos. Aporta mayor libertad al diseñador y mejora los informes de errores que tiene JSF. (25)

Capa de Negocio

La capa de negocio contiene los procesos a realizar con la información recibida desde la capa de presentación, las peticiones que el usuario ha realizado, y se responsabiliza de que se le envíen las respuestas adecuadas a la capa de presentación. (20)

JBoss Seam

JBoss Seam es un framework de código abierto que permite unir diferentes tecnologías y estándares de Java, Java Server Faces (JSF), Java Persistence API o (JPA), Enterprise Java Beans (EJB) y Business Process Management (BPM) en un sistema unificado con sofisticadas herramientas. Permite a los desarrolladores usar anotaciones POJO (Plain Object Java) para todos los componentes de la aplicación. (26)

Capa de Datos

La capa de datos contiene las clases que interactúan con la base de datos. Estas clases utilizan los procedimientos almacenados, para realizar todas las operaciones con la base de datos de forma transparente para la capa de negocio. (20)

Hibernate

Hibernate es un framework que provee herramientas de mapeo objeto/relacional y permite reducir el tiempo de desarrollo. Posee un lenguaje de consultas llamado HQL (parecido al lenguaje de consultas SQL). Sus herramientas soportan distintos tipos de base de datos, lo que confiere cierto nivel de portabilidad a las aplicaciones que lo utilizan. A través de la implementación del estándar JPA que provee Hibernate se puede realizar el acceso a datos. (27)

Enterprise JavaBeans (EJB)

Enterprise JavaBeans (EJB) son componentes del contexto de servidor que cubren la necesidad de intermediar entre la capa web y diversos sistemas empresariales. Encapsula la lógica del negocio de una aplicación de una forma integrada y permite construir aplicaciones de negocio portables y reutilizables usando el lenguaje de programación Java.

De esta manera no queda dispersa su representación en un grupo de sistemas empresariales. Están especialmente pensados para integrar la lógica de la empresa que se encuentra en sistemas distribuidos, de tal forma que el desarrollador no tenga que preocuparse por la programación a nivel de sistema, sino que se centre en la representación de entidades y reglas de negocio. (28)

Java Persistence API (JPA)

Java Persistence API (JPA) proporciona un modelo de persistencia basado en POJO's para mapear bases de datos relacionales en Java. Puede utilizarse directamente en aplicaciones web y aplicaciones clientes. El mapeo objeto/relacional, es decir, la relación entre entidades Java y tablas de la base de datos, se realiza mediante anotaciones en las propias clases de entidad, por lo que no se requieren ficheros descriptores XML. También pueden definirse transacciones como anotaciones JPA. (29)

1.3.3 Tecnologías Horizontales

Existen un conjunto de tecnologías que se extienden horizontalmente por todas las capas antes mencionadas y sirven de soporte a las tecnologías que se utilizan en cada una de ellas, estas son:

Java Platform Enterprise Edition (JavaEE 5)

Java Platform Enterprise Edition es una plataforma de programación (parte de la Plataforma Java) para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura distribuida o multicapa. Se basa ampliamente en componentes de software modulares y se ejecuta sobre un servidor de aplicaciones. (30)

Java Runtime Environment (JRE 6)

Java Runtime Environment (JRE) se corresponde con un conjunto de utilidades que permite la ejecución de programas Java sobre todas las plataformas soportadas. La Máquina Virtual de Java (JVM) es una instancia de JRE en tiempo de ejecución. Este interpreta el código Java y está compuesto además por las librerías de clases estándar que implementan el API de Java. Ambas JVM y API deben ser consistentes entre sí, de ahí que sean distribuidas de modo conjunto. (31)

JBoss Aplication Server

JBoss Server es un servidor de aplicaciones basado en J2EE (Java Platform, Enterprise Edition) e implementado en java. Por estar basado en dicho lenguaje, JBoss puede ser utilizado en cualquier sistema operativo que lo soporte y además implementa todo el paquete de servicios de J2EE. Es ideal como servidor de aplicaciones java para aplicaciones web. También soporta Enterprise Java Beans (EJB) 3.0 y cuenta con un conjunto de componentes claves como: JBoss AS 4.2, Hibernate 3.2.4, Seam 2.0. (32)

1.3.4 Metodologías de desarrollo de software

Una metodología es una colección de procedimientos, técnicas, herramientas y documentos auxiliares que ayudan a los desarrolladores de software en sus esfuerzos por implementar nuevas aplicaciones de software. En un proyecto de desarrollo, define quién debe hacer qué, cuándo y cómo hacerlo. Es un

proceso que puede seguir uno o varios modelos de ciclo de vida, indica cómo hay que obtener los distintos productos parciales y finales en el desarrollo de un software. (33)

En el desarrollo del presente trabajo se utiliza el Proceso Unificado de Modelado (RUP, por sus siglas en inglés), metodología que está predefinida por el Departamento de Gestión Hospitalaria.

Proceso Unificado Racional (RUP)

RUP es una metodología de desarrollo de un software que define claramente quién, cómo, cuándo y qué debe hacerse en el proyecto. Este proceso posee tres características, la primera es que está centrado en la arquitectura, ya que relaciona la toma de decisiones de cómo y en qué orden debe ser construido el sistema. La segunda características que tiene RUP es que está dirigido por Casos de Usos porque expresa lo que el usuario desea orientarle al proyecto o sea la importancia que tiene el mismo. Su última característica es, iterativo e incremental, es decir, divide en mini proyectos al proyecto, donde la arquitectura y los casos de usos cumplan sus objetivos.

RUP divide el proceso en 4 fases, la de Inicio, Elaboración, Construcción y Transición donde se realizan varias iteraciones en cada una de ellas. En la fase de Inicio se define el alcance del proyecto y se identifican los riesgos y los principales casos de uso, es de principal importancia en esta fase que los desarrolladores, clientes y usuarios finales logren ponerse de acuerdo en lo que quieren, además de entender el problema, la estructura y la dinámica de la organización. Ya en la fase de elaboración se seleccionan los casos de uso que permiten definir la arquitectura base del sistema, se realiza la especificación de los casos de uso seleccionados y se diseña la solución preliminar. En la fase de construcción se completan las funcionalidades del sistema, para ello se deben clarificar los requerimientos pendientes, administrar los cambios de acuerdo a las evaluaciones realizados por los usuarios y se realizan las mejoras para el proyecto. En la última fase, la de Transición, se asegura que el software esté disponible para los usuarios finales, se ajustan los errores y defectos encontrados en las pruebas de aceptación, se capacitan a los usuarios y se provee el soporte técnico necesario. (34)

Lenguaje Unificado de Modelado (UML 2.1)

UML es un lenguaje de representación visual que permite combinar diversos elementos gráficos y crear diagramas. Se usa para modelar sistemas y utiliza tecnología orientada a objetos. El Lenguaje Unificado de Modelado describe lo que hará un sistema pero no dice cómo implementarlo. Su objetivo es visualizar, especificar, construir y documentar los artefactos que se crean durante el proceso de desarrollo. Involucra todo el ciclo de vida del proyecto y está pensado para varios lenguajes y plataformas. (35)

1.3.5 Herramientas

Eclipse

Eclipse es un Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés) de código abierto y multiplataforma. Proporciona herramientas para gestionar espacios de trabajo, construir, lanzar y depurar aplicaciones y compartir objetos con el equipo de desarrollo. Está construido sobre un mecanismo para descubrir, integrar y ejecutar módulos (en inglés plug-ins). Este mecanismo de módulos es una plataforma ligera para componentes de software, que adicionalmente permite al Eclipse extenderse, usando otros lenguajes de programación como C/C++, Python y Java. (36)

JBoss Tools

JBoss Tools es un conjunto de herramientas para Eclipse. Entre estas herramientas, dispone de plug-ins que proporcionan soporte en Eclipse para Hibernate, JBoss Aplication Server (JBoss AS), JSF, XHTML, Seam, JBoss ESB (JBoss Enterprise Service Bus) o JBoss Portal, entre otros. JBoss Tools consta de varios módulos:

- ✓ Editor visual JSF y Facelets.
- ✓ Herramientas de generación, refactorización y completado de código Seam.
- ✓ Herramientas para Hibernate como ficheros de mapeo, anotaciones, ingeniería reversa con JPA.
- ✓ Herramienta para inicializar y parar el servidor de aplicaciones JBoss.
- ✓ Editor de workflows jBPM. (37)

PostgreSQL Server 8.3

PostgreSQL es un sistema de base de datos relacional de código abierto, que se destaca por su robustez, escalabilidad y cumplimiento de los estándares SQL. Este cuenta con diversas versiones para sistemas operativos tales como: Linux, Windows, Unix, Mac OS X, Solaris, BSD, Tru64 y otros. Permite crear, editar, copiar, extraer y bajar todo objeto de las bases de datos tales como: esquemas, tablas, vistas, funciones, dominios, reglas, secuencias, idiomas, operadores, etc. También construye consultas visualmente, ejecuta consultas y scripts SQL, visualiza y edita datos, representa datos como diagramas, exporta e importa datos desde y hacia los formatos de archivos de uso más popular. Además administra roles, usuarios, grupos y sus privilegios, y usa una serie de herramientas adicionales diseñadas para una fácil y eficiente operación con el Servidor PostgreSQL.

Dentro de las características que se destacan de PostgreSQL y que lo convierten en una herramienta eficiente para el trabajo en las base de datos están la consistencia, el aislamiento y la durabilidad, juntas aseguran que solo empieza aquello que se puede acabar, garantizan que una operación no puede afectar a otras y una vez realizada la operación, ésta persistirá y no se podrá deshacer aunque falle el sistema. (38)

PgAdmin III 1.6.3

Es una aplicación gráfica para trabajar con el gestor de bases de datos PostgreSQL, siendo la más completa y popular de código abierto. Está diseñado para responder a las necesidades de todos los usuarios, desde las simples escrituras de consultas SQL hasta el desarrollo de bases de datos complejas. La interfaz gráfica soporta todas las características de PostgreSQL y facilita su administración. La aplicación también incluye un resaltado de sintaxis SQL editor, un editor de código del lado del servidor, un agente de la programación y mucho más. (39)

Visual Paradigm 6.4

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a

un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. (40)

CAPÍTULO 2: DESCRIPCIÓN DE LA ARQUITECTURA DE SOFTWARE A UTILIZAR PARA DESARROLLAR LOS PROCESOS CON EL CÁLCULO DE LENTES INTRAOCULARES DEL MÓDULO CONSULTA EXTERNA.

Este capítulo tiene como objetivos, definir y describir los requerimientos no funcionales del sistema; analizar y fundamentar la arquitectura de software del Sistema de Información Hospitalaria alas HIS; detallar las estrategias de integración y especificar los estándares de diseño y codificación a utilizar en la realización de la implementación de las funcionalidades.

2.1 Requerimientos no funcionales

Los requerimientos no funcionales de un software son cualidades o propiedades que el producto debe tener. Estos muestran características que hacen al producto atractivo, usable, rápido o confiable y son fundamentales en el éxito del mismo. A diferencia de los requerimientos funcionales, estos no modifican la funcionalidad del producto, normalmente están vinculados a los requerimientos no funcionales, es decir, una vez se conozca lo que el sistema debe hacer se puede determinar cómo debe comportarse, qué cualidades debe tener o cuán rápido o grande debe ser. Los requerimientos no funcionales deben establecer restricciones en el producto que está siendo desarrollado, en el proceso de desarrollo y en restricciones específicas que el producto pueda tener. (41)

Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se tiene conocimiento que cumple con todas las funcionalidades requeridas, estas propiedades pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación. (42)

- ✓ **Requerimiento de usabilidad**

Las funcionalidades relacionadas con el cálculo de los Lentes Intraoculares del módulo Consulta Externa del sistema alas HIS, permitirán que los usuarios adquieran las habilidades necesarias para el trabajo con las mismas, brindándoles comodidad a la hora de acceder a las opciones correspondientes, por lo que se ofrece una navegabilidad sencilla. Además, la utilización de botones que permiten configurar las fórmulas,

posibilita al médico una vía más rápida de crear la expresión adecuada para la fórmula. El significado de los íconos y los textos debe ser claro para la persona que interactúe con la aplicación

- ✓ **Seguridad**

Las funcionalidades asociadas al cálculo de LIO se encontrarán dentro del sistema alas HIS. Este sistema de manera general en todos sus servicios, debe implementar mecanismos de seguridad y control a nivel de usuario, que garanticen el acceso de los mismos sólo a los niveles establecidos de acuerdo a la función que realizan. Las contraseñas podrán cambiarse sólo por el propio usuario o por el administrador del sistema. Estableciéndose para el servicio de Microcirugía del Cristalino un usuario, que en este caso será el oftalmólogo encargado de gestionar los datos correspondientes en la consulta.

Se registrarán todas las acciones que se realizan por cada usuario en todo momento. El sistema proporcionará un registro de actividades de cada usuario en el sistema. Ninguna información que se haya ingresado en el sistema será eliminada físicamente de la base de datos.

- ✓ **Requisitos de Hardware**

Los requerimientos de hardware estarán dados por la plataforma específica que se utilice para la inclusión de las funcionalidades asociadas a la investigación, en cuanto a sistema operativo, servidor de aplicaciones y gestor de bases de datos.

Estaciones de trabajo

Se necesitan estaciones de trabajo de 1GB de memoria RAM y un microprocesador Intel® Core-2 Duo o Intel® Dual-Core, con sistema operativo Windows o Linux.

Servidores

Para los servidores la solución estará conformada, fundamentalmente, por alta capacidad de procesamiento y redundancia, que permitan garantizar movilidad y residencia de la información y las aplicaciones bajo esquemas seguros y confiables.

Servidores de Base de datos: PowerEdge R910, Procesador Intel® Xeon® CPU E7- 8837 @ 2.67GHz, 16GB de memoria RAM, 1199GB de disco duro y sistema operativo Linux.

Servidores de Aplicaciones: PowerEdge R910, Procesador Intel® Xeon® CPU E7- 8837 @ 2.67GHz, 16GB de memoria RAM, 1199GB de disco duro y sistema operativo Linux.

- ✓ **Requisitos de Software**

Las funcionalidades relacionadas con el cálculo de los Lentes Intraoculares del módulo Consulta Externa estará integrado al sistema alas HIS, dicha aplicación debe poder ser desplegada en sistemas operativos Windows y Linux, utilizando la plataforma Java (Máquina Virtual de Java - Java Enterprise Edition), el servidor de aplicaciones JBoss AS y PostgreSQL como sistema para la gestión de la base de datos. Los usuarios deberán disponer de un navegador web, este puede ser Firefox 3.6, Google Chrome 14 o versiones superiores de ellos.

- ✓ **Restricciones en el diseño y en la implementación**

La capa de presentación contendrá todas las vistas y la lógica de la presentación. El flujo web se debe manejar de forma declarativa y basándose en definiciones de procesos del negocio. La capa del negocio mantendrá el estado de las conversaciones y procesos del negocio que concurrentemente pueden estar siendo ejecutados por cada usuario. La capa de acceso a datos contendrá las entidades y los objetos de acceso a datos correspondientes a las mismas. El acceso a datos está basado en el estándar JPA y particularmente en la implementación de este.

- ✓ **Requerimientos de interfaz de usuario**

Las ventanas del sistema que utilizará el usuario deberán contener bien estructurados los datos, además de permitir la interpretación correcta de la información. La interfaz deberá contar con accesos directos y menús desplegables que faciliten y aceleren su utilización. La entrada de datos incorrecta será detectada e informada al usuario.

2.2 Descripción de la arquitectura. Fundamentación

La arquitectura de software es un conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un software, permitiendo a los programadores, analistas y todo el conjunto de desarrolladores del software compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de la aplicación. Es considerada como el diseño de más alto nivel de la estructura de un sistema puesto que establecen la estructura, funcionamiento e interacción entre las partes del software. (41). Para la implementación de las funcionalidades que permitan gestionar el cálculo de Lentes

Intraoculares en el módulo Consulta Externa del Sistema de Información Hospitalaria alas HIS se utiliza el patrón de arquitectura Modelo Vista Controlador (MVC), sobre el cual se abordó en el capítulo anterior.

Con el objetivo de lograr una mejor organización de los componentes que forman parte del sistema, y siguiendo las características del patrón MVC, se agruparon dichos componentes en tres capas fundamentales, definidas por la arquitectura en capas: presentación, negocio y datos. Esta estructura garantiza que el cambio o sustitución de un elemento correspondiente a una de las capas, no altere el funcionamiento de otro que lo utilice y sea parte de alguna de las capas restantes.

La vista o capa de presentación está desarrollada básicamente con Java Server Faces (JSF), utilizándose la librería de componentes RichFaces, que se complementa fácilmente con la plataforma de trabajo de integración JBoss Seam y permite generar vistas no necesariamente basadas en HTML. Incluye conversión y validación de campos, establecimiento de reglas de navegación declarativas, la internacionalización y accesibilidad de la interfaz de usuario, un modelo orientado a eventos y combinado con Facelets, que brinda la capacidad añadida de la tecnología de plantillas.

La capa de negocio está compuesta por clases Java controladoras, que definen la lógica del negocio conjuntamente con los datos que se validan en la capa de presentación. Todo esto se desarrolla mediante la plataforma de trabajo JBoss Seam, creada para unificar todas las tecnologías estándares JSF, EJB3 y JPA. A estas clases, mediante anotaciones que provee JBoss Seam como marco de trabajo, se les puede especificar el contexto en que se encuentran, ya sea conversacional, evento, página, entre otros, los que definen el estado de los datos y las entidades que manejan.

En el modelo o capa de datos se usa la implementación de JPA de Hibernate, minimizando por un lado las configuraciones en XML sin chequeo de tipos. Se usa los servicios del contenedor de EJB3 y/o los contextos de persistencias administrados por JBoss Seam, para la transmisión del contexto de persistencia.

2.3 Estrategias de integración.

La práctica de reutilización de código fuente trae ventajas para toda aplicación que se encuentre en desarrollo y permite principalmente reducir tiempo, minimizar las redundancias y aprovechar el trabajo anterior. La forma más eficiente de la reutilización de código es la creación de componentes reutilizables para evitar la duplicidad del mismo.

Para el desarrollo del sistema alas HIS se utilizan varios componentes de forma común en todos los módulos del sistema con el objetivo de lograr uniformidad en el desarrollo y mejorar la calidad del trabajo. Entre los componentes que se reutilizan están: la clase ActiveModule para conocer en qué módulo y entidad se encuentra el usuario que está utilizando el sistema, la clase Bitácora para el control de las trazas de todas las acciones que se realizan con la aplicación; la clase User para saber qué usuario está trabajando con la aplicación en tiempo real.

2.4 Vista de despliegue

El Diagrama de Despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo. Un nodo es un elemento físico que existe en tiempo de ejecución y representa un recurso computacional, que generalmente tiene memoria y a menudo, capacidad de procesamiento. Los nodos se utilizan para modelar la topología del hardware sobre el que se ejecuta el sistema. Representan un procesador o un dispositivo sobre el que se pueden desplegar los componentes. La relación entre un nodo y el componente que despliega puede mostrarse con una relación de dependencia. (43)

La aplicación está distribuida en tres nodos**:** una computadora cliente y dos servidores (servidor de aplicación y servidor de base de datos), además se dispuso de la conexión con un dispositivo, en este caso la impresora. La comunicación entre los nodos se realiza por los protocolos HTTP para asociar la computadora cliente y el servidor de aplicaciones y por TCP/IP para establecer la conexión entre los dos servidores. La computadora cliente se conecta a la impresora a través de los puertos USB o LPT.

Diagrama de Despliegue

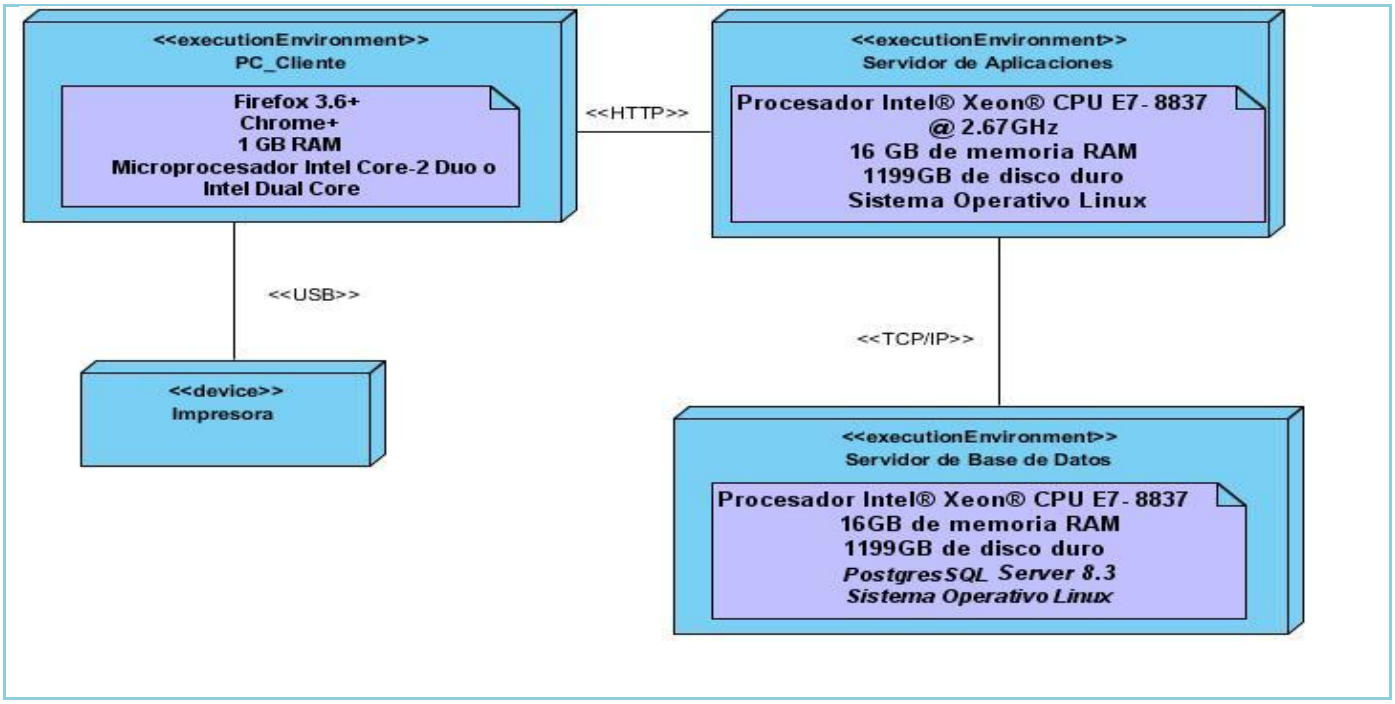

Figura 2.1 Diagrama de Despliegue

2.5 Estrategias de codificación. Estándares y estilos a utilizar

La mejor forma de lograr la legibilidad del código implementado por más de una persona es la aplicación de un estándar de codificación. Un estándar de codificación es un conjunto de directrices, normas y reglamentos sobre la forma de escribir código. Por lo general, incluye pautas sobre cómo nombrar variables, la forma de guión, el código, cómo poner entre paréntesis y palabras clave, etc. El propósito fundamental de un estándar de codificación es que el sistema tenga una arquitectura y un estilo consistente con lo cual resulte fácil de entender el formato y el estilo utilizado en el código escrito por otros desarrolladores. (44)

Para el desarrollo de las funcionalidades presentadas para el sistema alas HIS, se utiliza un estándar de codificación, para garantizar la homogeneidad del estilo de programación durante la implementación en el sistema. Para el nombre de las clases se utilizará la notación o estilo Pascal Casing, con la que los nombres quedan compuestos por múltiples palabras juntas iniciando cada palabra con letra mayúscula y terminando todo el nombre con "_" y el nombre del módulo al que pertenece. Los identificadores de

atributos y métodos de la clase se escribirán en notación Camel Casing: nombres compuestos por múltiples palabras juntas iniciando cada palabra con letra mayúscula excepto la primera palabra que debe iniciar con minúscula. Las clases controladoras terminarán con el nombre del módulo al que pertenecen y la palabra “controlador” al final.

A continuación se presentan las restricciones de nomenclatura y estándares de codificación que fueron utilizados basándose en los estándares antes mencionados:

Idioma

Todas las palabras deben ser en idioma español y sin acentuarse.

Identación

Inicio y fin de bloque: Se debe dejar dos espacios en blanco desde la instrucción anterior para el inicio y fin de bloque {}. Lo mismo sucede para el caso de las instrucciones if, else, for, while, do while, switch, foreach.

Aspectos generales: El indentado debe ser de dos espacios por bloque de código. No se debe usar el tabulador; ya que este puede variar según la computadora o la configuración de dicha tecla. Los inicios ({) y cierre (}) de ámbito deber estar alineados debajo de la declaración a la que pertenecen y deben evitarse si hay sólo una instrucción. Nunca colocar ({) en la línea de un código cualquiera, esto requiere una línea propia.

Comentarios, separadores, líneas, espacios en blanco y márgenes

Con un estándar en comentarios, separadores, líneas, espacios en blanco y márgenes se logra establecer un modo común para comentar el código de forma tal que se comprenda con sólo leerlo una vez.

Comentarios: No comentar cada línea de código. Establecer un modo común para realizar el comentario del código de forma tal que sea comprensible con sólo leerlo una vez. Se recomienda hacerlo al inicio de la clase o función, especificando el objetivo de la misma así como los parámetros que usa, detallando tipo de dato y objetivo del parámetro.

Ubicación de comentarios: Estos se ubican al inicio de cada clase o función y al final de cada bloque de código, especificando el objetivo de la misma así como el tipo de dato y el objetivo de cada parámetro.

Líneas en blanco: Se recomienda dejar una línea en blanco antes y después de la declaración de una clase o de una estructura y de la implementación de un método. Cuando el comentario se aplica a un grupo de instrucciones debe estar seguido de una línea en blanco.

Espacios en blanco: Se recomienda entre operadores lógicos y aritméticos para lograr una mayor legibilidad en el código.

Aspectos generales: Cuando se comente una sola instrucción se suprime la línea en blanco o se escribe a continuación de la instrucción. No se usa un espacio en blanco después del corchete abierto y antes del cerrado de un arreglo, después del paréntesis abierto y antes del cerrado y antes de un punto y coma.

Clases y objetos

El objetivo fundamental es nombrar las clases e instancias de forma estándar para todas las aplicaciones.

Apariencia de clases y objetos: Los nombres de las clases deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing, la cual sugiere iniciar cada palabra con letra mayúscula. Para el caso de las instancias se comenzará con un prefijo que identificará el tipo de dato, este se escribirá en minúscula.

Apariencia de atributos: Se empleará notación CamellCasing.

Apariencia de las funciones: El nombre de las funciones con verbos que denoten la acción que realiza y se empleará la notación CamellCasing. Las funcionen que obtienen algún dato emplean el prefijo get y si fijan algún valor se emplea el prefijo set.

Declaración de parámetros en funciones: Se declaran agrupados por tipos, definiendo primero los string y luego los numéricos, además se agrupan tenido en cuenta los valores por defecto.

Aspectos generales: El nombre que se emplea para las clases, objetos, atributos y funciones debe permitir que con sólo leerlo se conozca el propósito de los mismos.

Bases de datos, tablas, esquemas y campos

Apariencia de la base de datos: Los nombres serán cortos y descriptivos.

Apariencia de las tablas: Todas las letras en minúscula, en caso de ser un nombre compuesto se utiliza underscore (_) para separarlo.

Apariencia de los campos: El nombre de los campos se escribe con todas las letras en minúscula. Los campos identificadores comienzan con el identificador id seguido de underscore y posteriormente el nombre del campo.

Aspectos generales: El nombre empleado para las bases de datos, las vistas, las tablas, los campos y los procedimientos almacenados, deben permitir que con sólo leerlos se conozca el propósito de los mismos.

Controles

Apariencia de los controles: el nombre que se le asigna a los controles comienza con las primeras letras en minúscula, las cuales identifican el tipo de datos al que se refiere. Para los nombres compuestos se emplea la notación CamellCasing.

CAPÍTULO 3: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA PARA DESARROLLAR LOS PROCESOS DE CÁLCULO DE LENTE INTRAOCULAR DEL MÓDULO CONSULTA EXTERNA.

Con la elaboración del capítulo actual, se persiguen los siguientes objetivos: describir las nuevas clases u operaciones necesarias para implementar las funcionalidades, refinar el modelo de datos, conformar la vista de implementación con la intención de mostrar las organizaciones y dependencias lógicas entre componentes software y elaborar el diagrama de componentes que será utilizado para modelar la vista estática del sistema.

3.1 Valoración crítica del diseño propuesto por el analista

Antes de desarrollar las funcionalidades relacionadas con el cálculo de Lentes Intraoculares del módulo Consulta Externa, correspondiente al flujo de trabajo “Implementación”; se hace necesario realizar un análisis y valoración de las actividades llevadas a cabo por el analista. Al realizar el análisis de los artefactos generados durante el flujo de trabajo “Análisis y Diseño”, se llegó a la conclusión de que existían varios de ellos, que no cumplían con las nuevas especificaciones del sistema. Por lo antes planteado, se hizo necesario realizar un refinamiento del diseño, lo cual incluía la modificación de los Diagramas de Clases del Diseño y los Diagramas de Secuencia; con el objetivo de añadir los nuevos métodos y atributos que cumplieran con las nuevas necesidades.

Se hizo necesario además, la modificación del diagrama de clases persistentes de la base de datos, añadiendo atributos, clases y relaciones generando nuevamente el modelo de datos y el script para la base de datos. Por otra parte, la propuesta de diseño carecía de componentes necesarios para poder desarrollar los casos de uso del sistema, además de no cumplir con las pautas de diseño establecidas por el Departamento de Gestión Hospitalaria. Debido a esto, se realizó un proceso de reestructuración de la propuesta de diseño, para luego desarrollar las funcionalidades relacionadas con el cálculo del Lente Intraocular.

En el diseño se utilizan un conjunto de patrones de diseños que brindan una solución ya probada y documentada a problemas comunes en el desarrollo de software. Para la definicion de la solución propuesta se tuvieron en cuenta los Patrones de Software para la Asignación General de Responsabilidad (GRASP, por sus siglas en inglés) y GOF (Gang of Four en inglés) con el fin de asignar responsabilidades a las diferentes clases que se definen en el diseño.

Los patrones GRASP se encuentran dentro de los más utilizados, de todos ellos se destacan principalmente por su utilización en el diseño, los patrones Experto y Creador. El Experto permite que cada clase haga lo que tiene que hacer de acuerdo con la información que contienen las mismas, mientras que el Creador permite que se creen instancias en correspondencia con las responsabilidades de las mismas. Todo esto conserva el encapsulamiento de la información, pues los objetos utilizan su propia información para llevar a cabo las tareas y soporta un bajo acoplamiento que da lugar a sistemas fáciles de mantener. (45)

Dentro de esta misma clasificación de patrones que se usaron también se encuentran el de Bajo acoplamiento, que permite que las clases estén lo menos ligadas posibles, de forma tal que si se realiza un cambio en una de ellas, no afecte las demás del diseño, y el patrón Alta cohesión que permite que en cada clase se realicen los métodos de la forma más coherente posible, acorde a la operación que tienen que realizar. (46)

Se utilizó además el patrón Abstract Factory de GOF que permite el acceso a la base de datos a través de EntityManager ya que resulta ser la interfaz principal de JPA, con el objetivo de lograr la persistencia de las aplicaciones y realizar operaciones como inserción, lectura, modificación y eliminación sobre un conjunto de objetos persistentes.so

El Diseño también se encarga de modelar el sistema para que soporte todos los requisitos, funcionales y no funcionales que se le suponen. Este modelo, es usado como una entrada inicial en las actividades de implementación.

Para un mejor entendimiento del Modelo de Diseño es preciso acotar algunos términos que serán empleados en la confección del mismo, estos son:

- Diagrama de clases: Son diagramas de estructura estática que muestran las clases del sistema y sus interrelaciones. Son utilizados para mostrar lo que el sistema puede hacer (análisis) y como puede ser construido (diseño).

- Diagramas de Interacción: Son considerados artefactos que gráficamente muestran las relaciones entre los objetos, incluyendo los mensajes que se pueden enviar entre ellos. Este tipo de diagrama se encuentra dividido por los Diagramas de Comunicación y de Secuencia, el primero destaca la organización estructural de los objetos que envían y reciben mensajes. Los Diagramas de Secuencia se diferencian en dos características de los Diagramas de Comunicación, en primer lugar, la línea de vida, la cual es la línea discontinua vertical que representa la existencia de un objeto a lo largo de un período de tiempo y en segundo lugar está el foco de control; rectángulo delgado y estrecho que representa el período de tiempo durante el cual un objeto ejecuta una acción. (47)

A continuación se presentan las realizaciones de los Diagramas de Clases del Diseño (DCD) y los Diagramas de Secuencia (DS), de los casos de usos: “Configurar nueva fórmula”, “Configurar criterio” y “Realizar cálculo del Lente Intraocular”

DCD_Configurar nueva fórmula

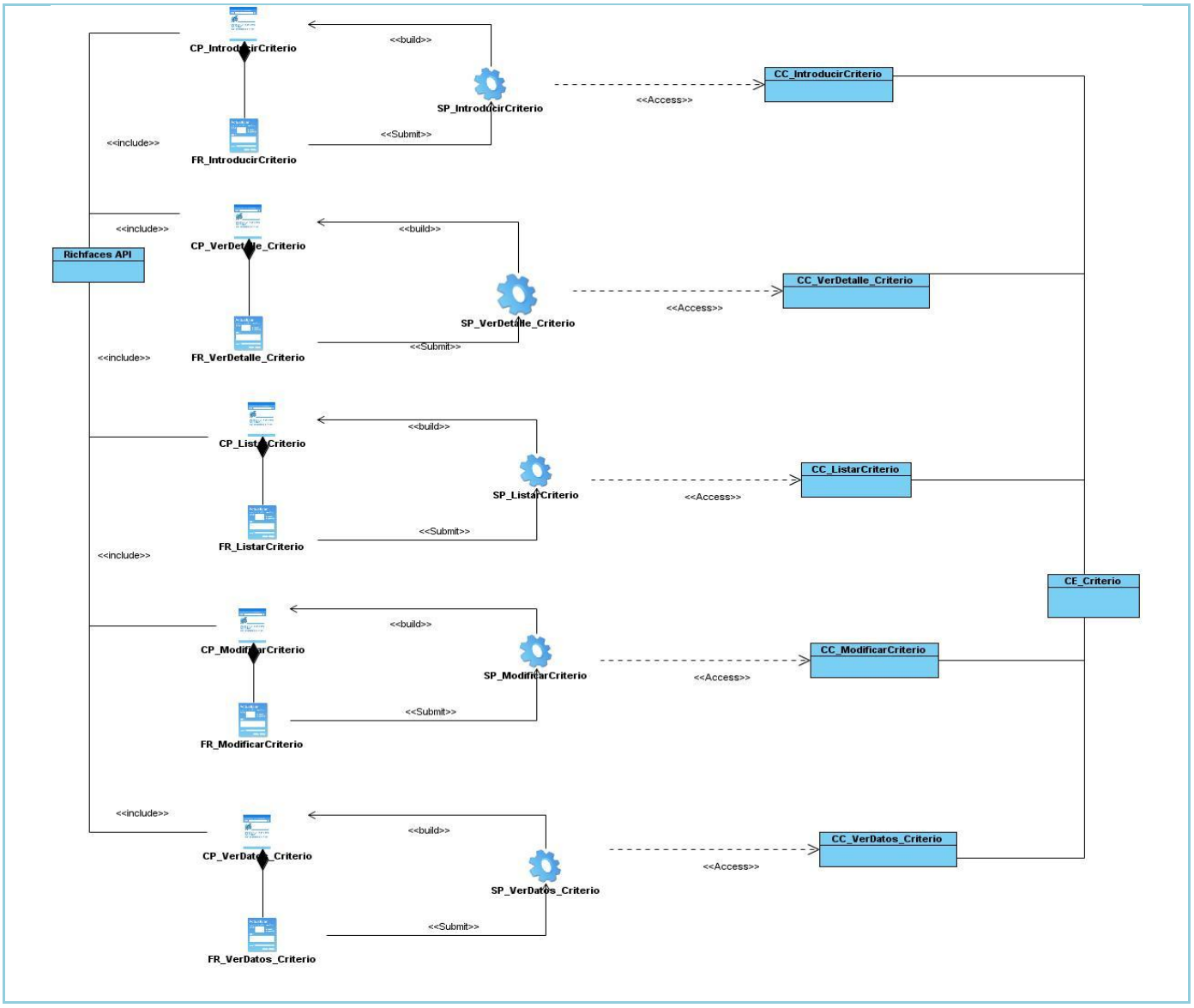

Figura 3.1 Diagrama de Clases de Diseño: Configurar Fórmula

DS_Configurar nueva fórmula

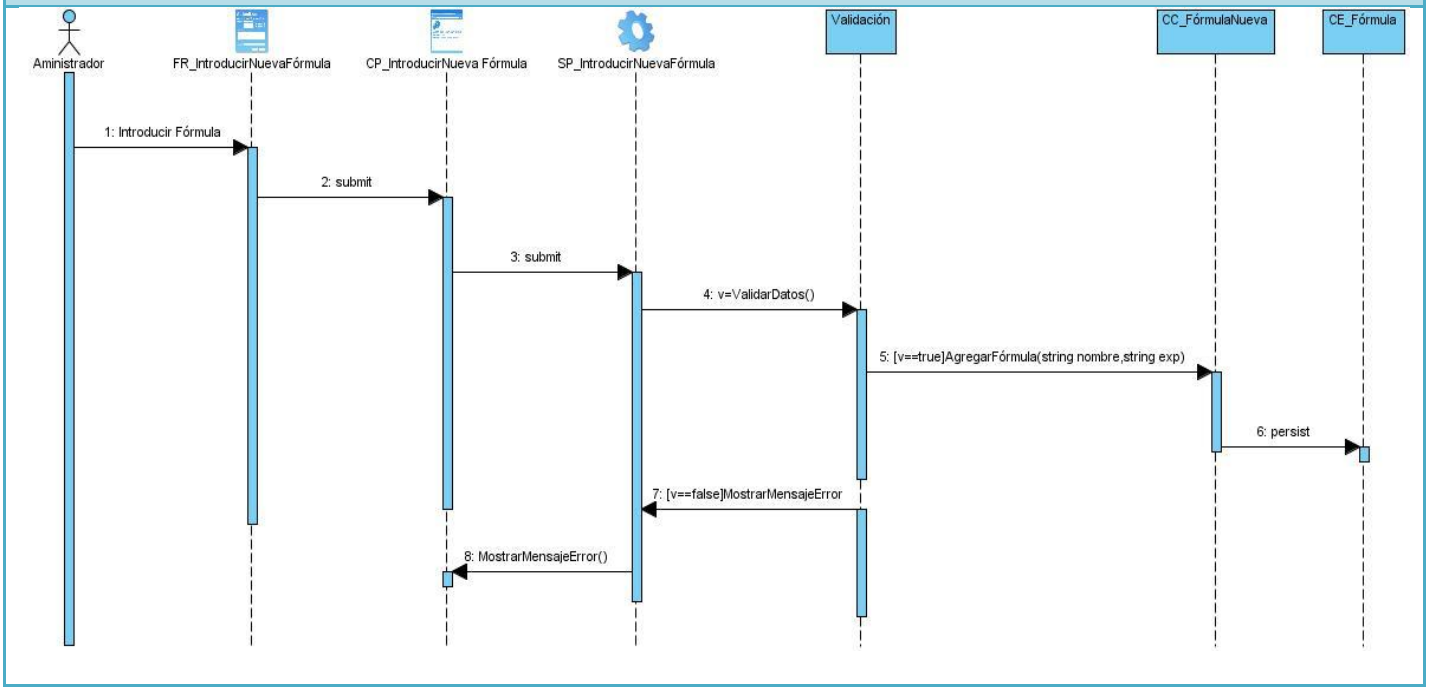

Figura 3.2 Diagrama de Secuencia: Configurar fórmula

DCD_Configurar nuevo criterio

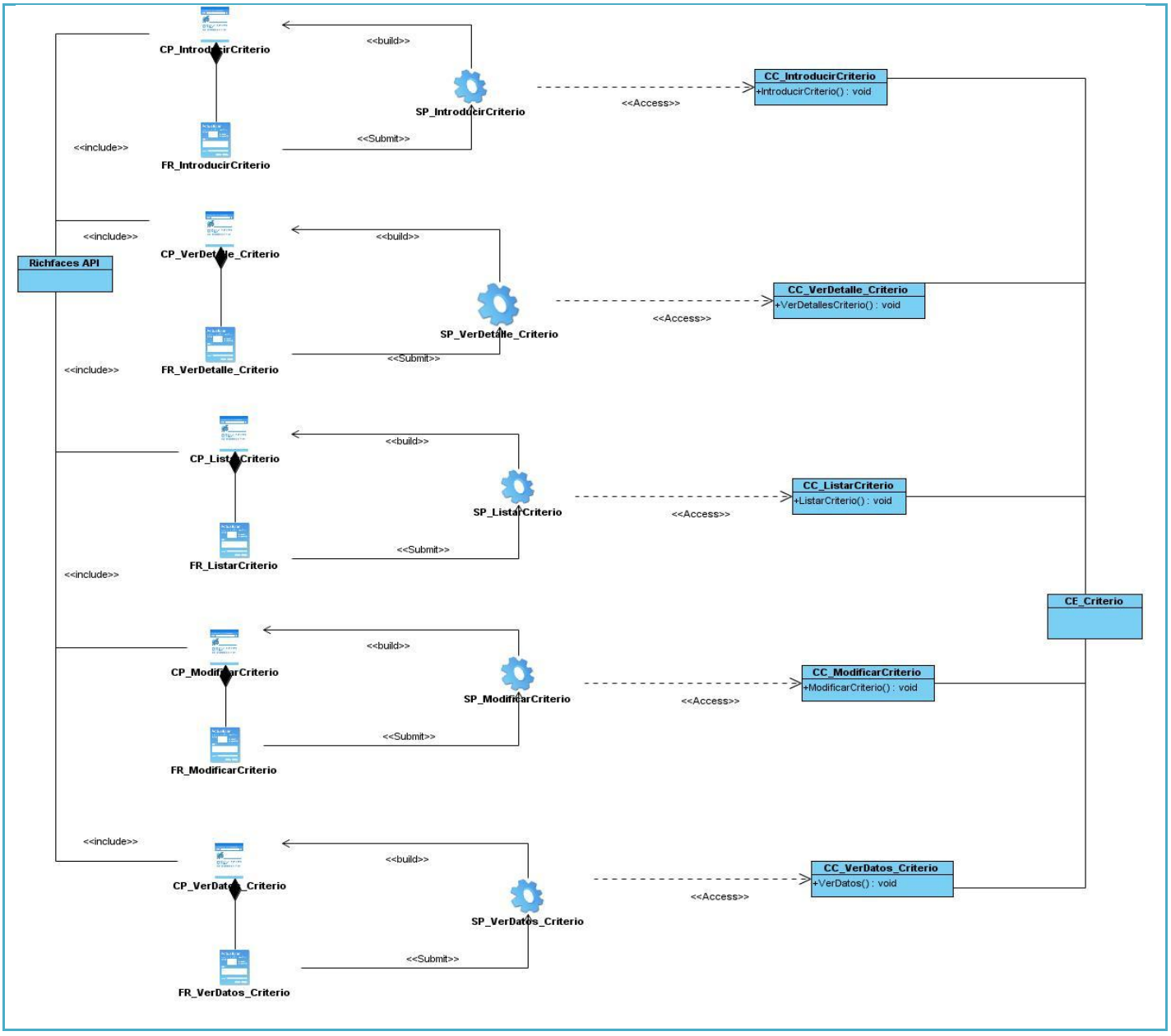

Figura 3.3 Diagrama de clases del Diseño: Configurar criterio

DS_Configurar nuevo criterio

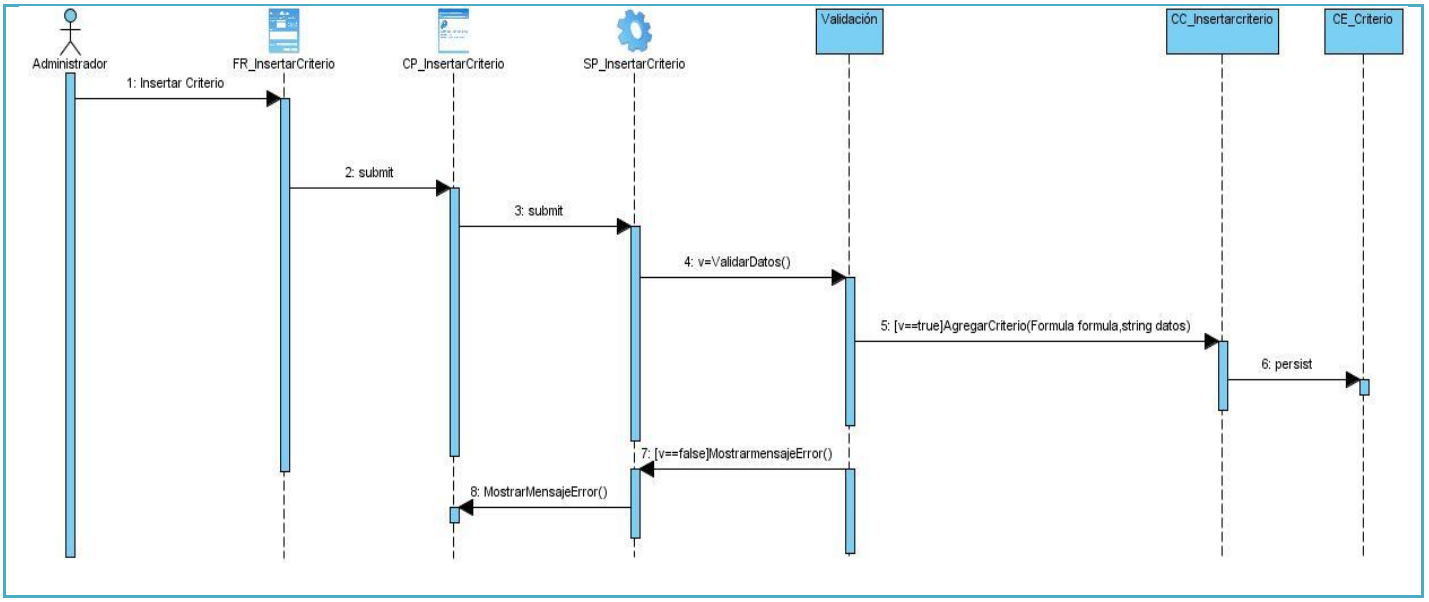

Figura 3.4 Diagrama de secuencia: Configurar Criterio

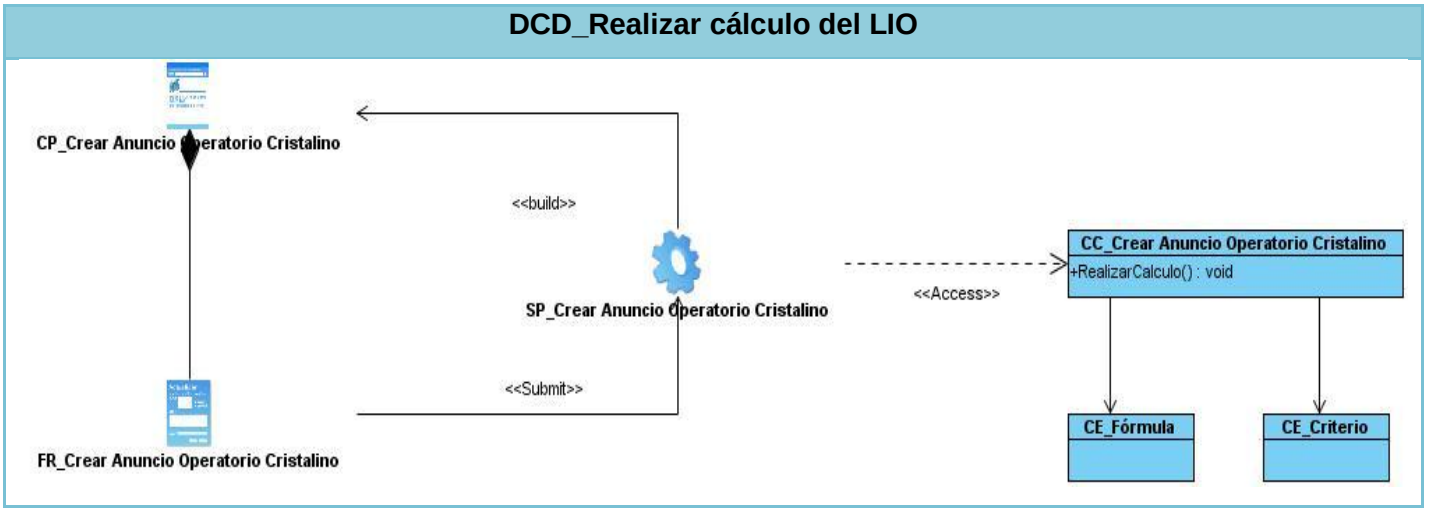

Figura 3.5 Diagrama de clases del diseño: Realizar cálculo

DS_Realizar cálculo del LIO

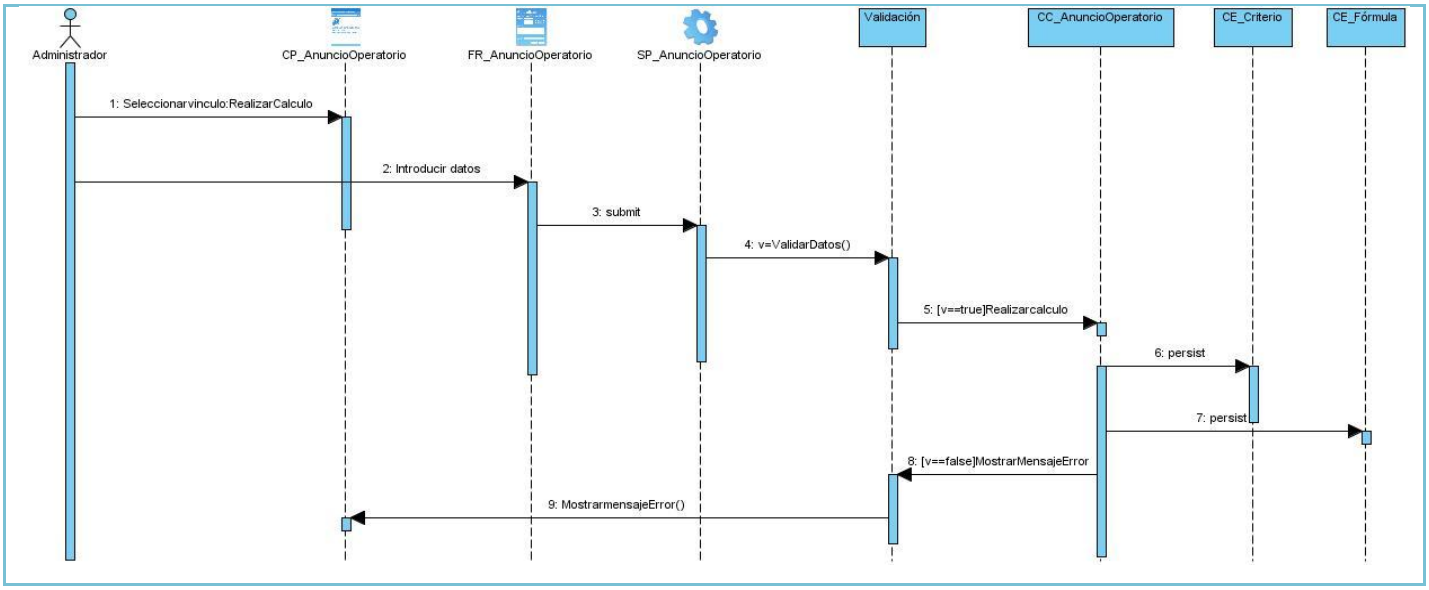

Figura 3.6 Diagrama de secuencia: Realizar cálculo

Para elaborar el diseño se definió una estructura de paquetes que permite dividir el sistema en fragmentos manejables para su implementación. Se emplea además el criterio de empaquetamiento por procesos y por clases, siguiendo la estructura de procesos definidos en el sistema.

Se crea un paquete Repositorio de clases que contiene tres subpaquetes: el de las Vistas, el de las Sesiones y el paquete de las Entidades. El paquete de las Vistas contiene los contenidos Web referentes a las páginas clientes y los formularios que la componen. En el de Sesiones se agrupan las clases controladoras autogeneradas por el entorno de desarrollo y las clases controladoras personalizadas. El subpaquete de las Entidades contiene las Entidades Autogeneradas y las Personalizadas. Las Entidades Autogeneradas son obtenidas desde la base de datos mediante el mapeo objeto-relacional (permite generar entidades desde la base de datos). Las Entidades Personalizadas son las modificadas o entidades que heredan de las autogeneradas y cambiadas para lograr un mejor funcionamiento.

A continuación se muestra el diagrama de paquetes del diseño del sistema:

Diagrama de Paquetes

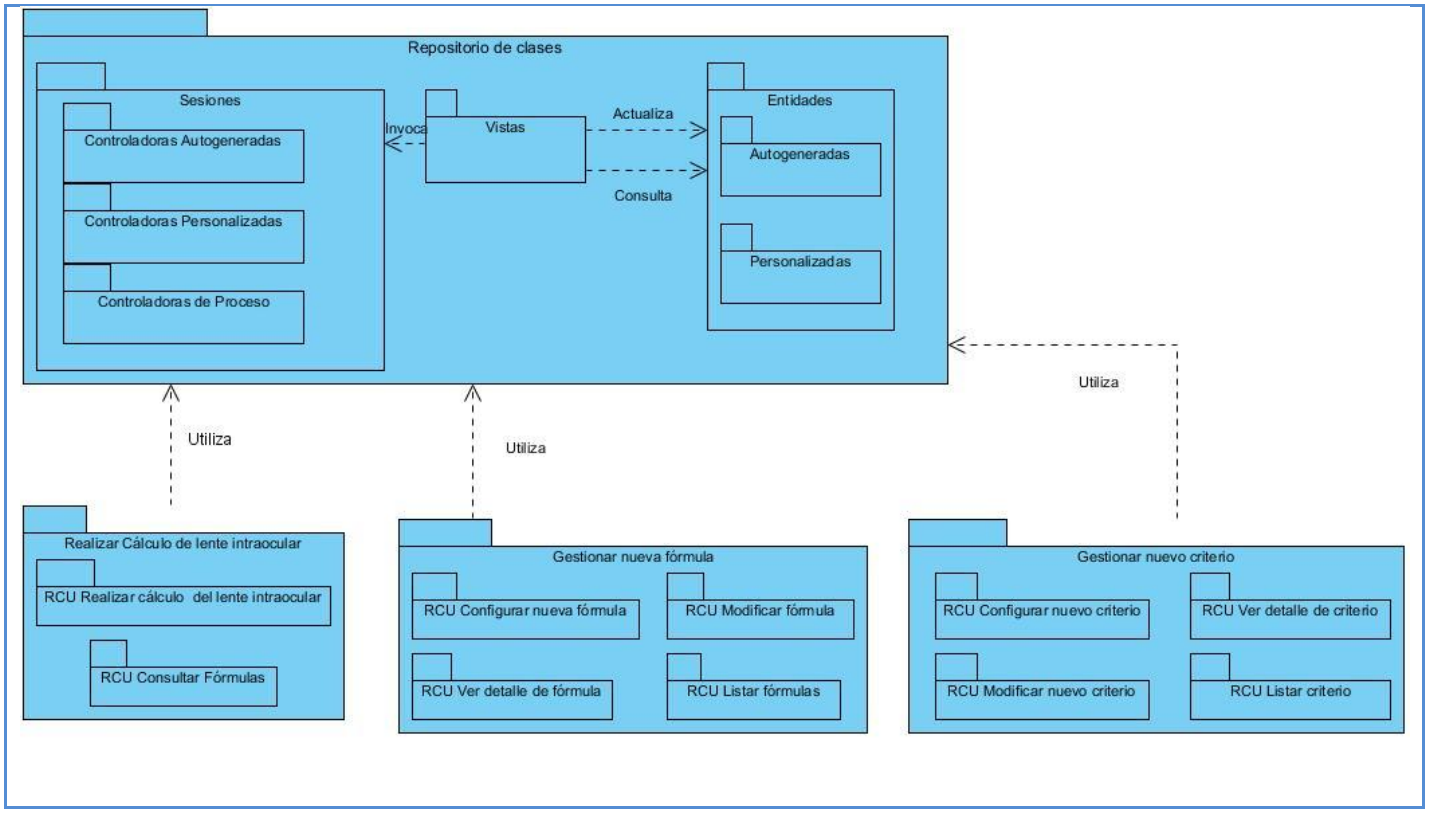

Figura 3.7 Diagrama de Paquetes

3.2 Descripción de las nuevas clases u operaciones necesarias

Tabla 1: Descripción de la clase Introducir Fórmula

Nombre: introducirFormula	
Tipo de Clase: Controladora	
Atributo	**Tipo**
idFormulaDioptria	Integer
descripcion	String

formula	String
Para cada responsabilidad:	
Nombre	**Descripción**
introducirFormula()	Permite agregar una nueva fórmula al sistema.

Tabla 3.1 Descripción de la clase controladora: *IntroducirFormula*

Tabla 2: Descripción de la clase Ver Detalles de la fórmula.

Nombre: verDetallesFormula	
Tipo de Clase: Controladora	
Atributo	**Tipo**
idFormulaDioptria	Integer
descripcion	String
formula	String
Para cada responsabilidad:	
Nombre	**Descripción**
verDetallesFormula ()	Permite ver los detalles de la nueva fórmula agregada al sistema.

Tabla 3.2 Descripción de la clase controladora: *verDetallesFormula*

Tabla 3: Descripción de la clase Listar nuevas fórmulas.

Nombre: listarFormula
Tipo de Clase: Controladora

Atributo	**Tipo**
idFormulaDioptria	Integer
descripcion	String
formula	String
Para cada responsabilidad:	
Nombre	**Descripción**
listarFormula ()	Permite listar la nueva fórmula y las existentes.

Tabla 3.3 Descripción de la clase controladora: *listarFormula*

Tabla 4: Descripción de la clase Modificar nueva fórmula.

Nombre: modificarFormula	
Tipo de Clase: Controladora	
Atributo	**Tipo**
idFormulaDioptria	Integer
descripcion	String
formula	String
Para cada responsabilidad:	
Nombre	**Descripción**
modificarFormula ()	Permite modificar la nueva fórmula y las existentes.

Tabla 3.4 Descripción de la clase controladora: *modificarFormula*

Tabla 5: Descripción de la clase Introducir nuevo criterio de trabajo.

Nombre: introducirCriterio
Tipo de Clase: Controladora

Atributo	**Tipo**
idCriterio	Integer
descripcion	String
formula	String
Nombre	string
Para cada responsabilidad:	
Nombre	**Descripción**
introducirCriterio()	Permite introducir un nuevo criterio de trabajo al sistema.

Tabla 3.5 Descripción de la clase controladora: *introducirCriterio*

Tabla 6: Descripción de la clase Ver detalles del nuevo criterio

Nombre: verDetallesCriterio	
Tipo de Clase: Controladora	
Atributo	**Tipo**
idCriterio	Integer
descripcion	String
formula	String
Para cada responsabilidad:	
Nombre	**Descripción**
verDetallesCriterio()	Permite ver los detalles del criterio agregado al sistema.

Tabla 3.6 Descripcion de la clase controladora: *verDetallesCriterio*

Tabla 7: Descripción de la clase Listar nuevos criterios existentes

Nombre: listarCriterio	
Tipo de Clase: Controladora	
Atributo	**Tipo**
idCriterio	Integer
descripcion	String
formula	String
Para cada responsabilidad:	
Nombre	**Descripción**
listarCriterio()	Permite listar los criterios agregados y los existentes en el sistema.

Tabla 3.7 Descripción de la clase controladora: *listarCriterio*

Tabla 8: Descripción de la clase Modificar nuevo criterio

Nombre: modificarCriterio	
Tipo de Clase: Controladora	
Atributo	**Tipo**
idCriterio	Integer
descripcion	String
formula	String
Para cada responsabilidad:	
Nombre	**Descripción**
modificarCriterio()	Permite modificar los criterios agregados y los existentes en el sistema.

Tabla 3.8 Descripción de la clase controladora: *modificarCriterio*

Tabla 9: Descripción de la clase Ver datos del criterio

Nombre: verDatosCriterio	
Tipo de Clase: Controladora	
Atributo	**Tipo**
idCriterio	Integer
descripcion	String
formula	String
Para cada responsabilidad:	
Nombre	**Descripción**
verDatosCriterio()	Permite ver los datos del criterio agregado.

Tabla 3.9 Descripción de la clase controladora: *verDatosCriterio*

Tabla 10: Descripción de la clase Realizar calculo

Nombre: realizarCalculo	
Tipo de Clase: Controladora	
Atributo	**Tipo**
idCriterio	Integer
Ojo_operar	String
Tipo_lente_implantar	String
Tipo_lente_especifico	String
Para cada responsabilidad:	
Nombre	**Descripción**
realizarCalculo()	Permite realizar el cálculo del Lente Intraocular.

Tabla 3.10 Descripción de la clase controladora: *realizarCalculo*

Tabla 11: Descripción de la clase Personalizar constantes

Nombre: personalizarConstantes	
Tipo de Clase: Controladora	
Atributo	**Tipo**
dioptría	Integer
Cámara_anterior	interger
Cámara_posterior	interger
Para cada responsabilidad:	
Nombre	**Descripción**
personalizarConstantes()	Permite modificar las constates utilizadas para el cálculo del lente.

Tabla 3.11 Descripción de la clase controladora: *personalizarConstante*

3.3 Modelo de Datos

El modelo de datos es un conjunto de conceptos que permiten describir, a distintos niveles de abstracción, los datos de una aplicación o sistema de información. Este tiene gran importancia en el ciclo de desarrollo de software, y de manera particular para la fase de implementación, pues define formalmente las estructuras permitidas y las restricciones que se aplican, con el fin de representar los datos del dominio de la aplicación. Está compuesto por entidades, atributos y relaciones. (48)

El modelo de datos describe la representación lógica y física de la información persistente manejada por el sistema. Puede ser inicialmente creado a través de la ingeniería inversa de un almacenamiento de datos persistentes que ya exista o puede ser inicialmente creado a partir de un conjunto de clases del diseño persistentes en el modelo de diseño. Es usado para definir el mapeo entre las clases del diseño y las estructuras de datos.

Modelo de Datos

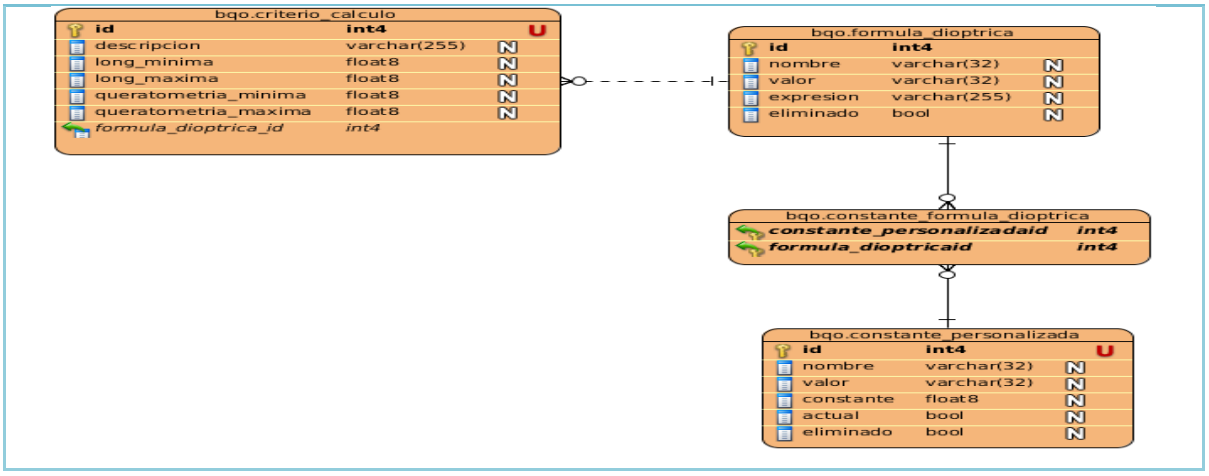

Figura 3.8 Modelo de datos

3.4 Valoración de las técnicas de validación

Durante el desarrollo de la aplicación el tratamiento de errores fue un aspecto importante a medir. La validación de los datos es el proceso de comprobar los valores introducidos a la aplicación. Una buena práctica de programación es la comprobación de estos datos para garantizar la valides y fidelidad de los mismos, elevando la calidad de la aplicación.

Para la validación de los datos se pueden utilizar clases validadoras, las cuales a través de métodos específicos se encargan de comprobar los datos. Otra vía para llevar a cabo la validación es la utilización del Framework JSF, el cual permite la manipulación y gestión de las validaciones mediante los controles de formularios o desde un componente. La utilización de la captura y tratamiento de errores mediante el bloque reservado de Java try/catch también fue utilizado para garantizar la calidad de la aplicación.

3.4.1 *Descripción de las tablas de la base de datos*

Nombre: cfg_criterio_cálculo		
Entidad que recoge los datos que se guardan al insertar un nuevo criterio.		
Atributo	**Tipo**	**Descripción**
id_criterio	INTEGER	Identificador del nuevo criterio.
descripción	VARCHAR	Nombre del nuevo criterio.

Long_min	float	Longitud axial mínima
Long_max	Float	Longitud axial máxima
Queratometría_min	float	Queratometría mínima
Queratometría_max	Float	Queratometría máxima
Fómula dióptrica	Interger	Fórmula asociada al criterio

Tabla 3.12 bqo_criterio_cálculo

Nombre: bqo_fórmula_dióptrica		
Entidad que recoge los datos que se guardan al insertar una nueva fórmula.		
Atributo	**Tipo**	**Descripción**
id_formula_dioptrica	INTEGER	Identificador de la nueva fórmula.
descripción	VARCHAR	Nombre de la nueva fórmula.
Formula	STRING	Expresión de la nueva fórmula.
Eliminado	Bool	

Tabla 3.13 bqo_fórmula_dióptrica

Nombre: cfg_constante_personalizada		
Entidad que recoge los datos que se guardan al seleccionar una nueva constante.		
Atributo	**Tipo**	**Descripción**
id_constante	INTEGER	Identificador de la nueva constante.
Constante	DOUBLE	Valor de la constante.
Actual	BOOLEAN	Especifica si es la constante actual.
Nombre	VARCHAR	Nombre de la constante.

Tabla 3.13 cfg_constante_personalizada

3.5 Vista de Implementación

Un diagrama de componentes es utilizado para estructurar el modelo de implementación en términos de subsistemas y modelar la vista estática de un sistema, describiendo sus elementos físicos y las relaciones

entre estos. Representa cómo un sistema de software es dividido en componentes y muestra la organización y las dependencias lógicas entre estos. Los componentes de software son una parte física de un sistema, y se encuentran en la computadora. Pueden ser: código fuente, librerías, binarios o ejecutables. (49)

A continuación se muestra una representación que describe la estructura de los componentes del sistema agrupados por paquetes lógicos:

Diagrama de Componentes

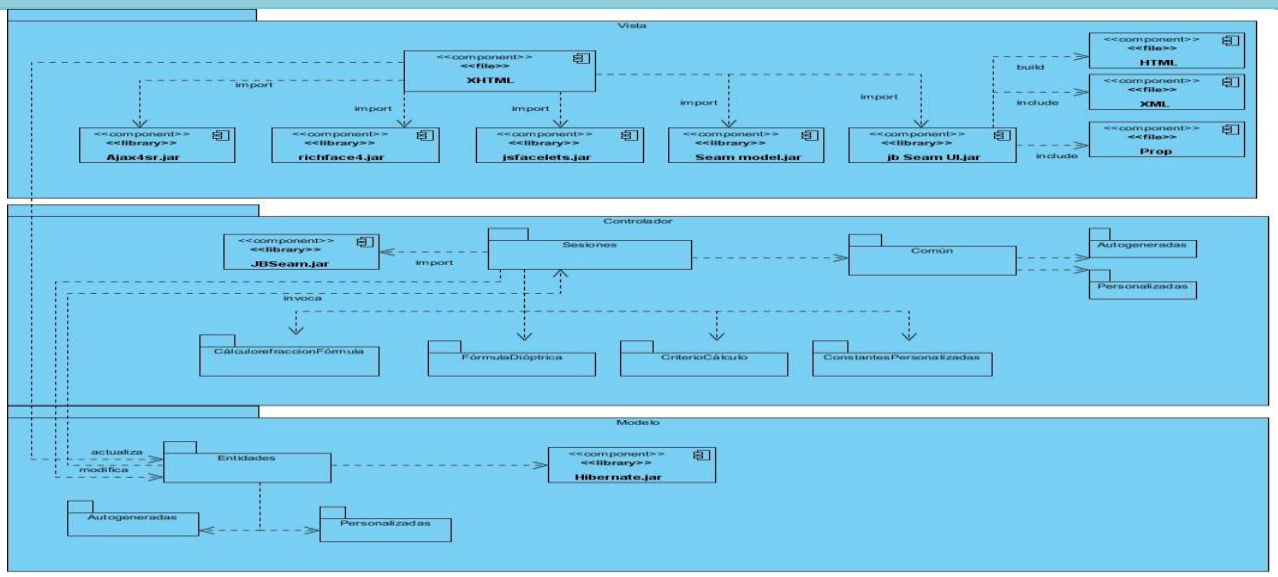

Figura 3.9 Diagrama de Componentes

CONCLUSIONES

Una vez finalizada la presente investigación y después de haber cumplido con el objetivo y las tareas propuestas, se arribaron a las siguientes conclusiones:

1. El análisis a los sistemas informáticos relacionados con el campo de acción no constituyen una solución a la problemática planteada. La gran mayoría de los sistemas estudiados no responden a las necesidades, dependiendo de las funcionalidades definidas para el desarrollo de la investigación.
2. Para el desarrollo de las funcionalidades se asimiló la arquitectura definida por el departamento de Sistemas de Gestión Hospitalaria utilizándose como patrón arquitectónico el MVC, así como las tecnologías y herramientas propuestas y utilizadas por el mismo.
3. La adopción de las pautas de diseño de interfaz de usuario definidas en el Departamento de Sistemas de Gestión Hospitalaria permitió que los nuevos casos de uso estén visualmente homogéneos a los existentes, obteniendo una aplicación visualmente uniforme.
4. El análisis del diseño propuesto por el analista, permitió realizar el refinamiento del diseño y de la base de datos, lo que posibilitó iniciar el flujo de implementación con la mínima cantidad de errores, evitando de esta manera la pérdida de tiempo y recursos.

RECOMENDACIONES

1. Incorporar funcionalidades que permitan realizar la gestión del control del Lente Intraocular en el módulo Consulta Externa del sistema alas HIS.

REFERENCIAS BIBLIOGRÁFICAS

1. **Font, Lic. Jaime Cruz.** Informática Médica. [En línea] Agosto de 2009. [Citado el: 20 de Noviembre de 2012.] www.cocmed.sld.cu/no44/n44ori1.htm.

2. **Hoyos, Dr. Fernando Bergaz de.** Kateron Systems. [En línea] [Citado el: 22 de Noviembre de 2012.] http://www.kanteron.com..

3. *Automatización del cálculo del poder dióptrico de los lentes intraoculares para las operaciones oftalmológicas.* **Ing. Alexander Rodríguez Rabelo, Ing Joselín Miló Pérez, Ing Mauricio Espinosa Robaina y Ing Miriam Rodríguez Reyes.** La Habana : s.n.

4. **Nodales, Edith María Ballate.** La Oftalmología. [En línea] 31 de Octubre de 2009. [Citado el: 22 de Enero de 2013.] http://www.articulo.org/articulo/9408/la_oftalmologia.html..

5. **Dr.Zaldívar.** Oftalmología avanzada para el mundo. [En línea] [Citado el: 25 de Enero de 2013.] http://www.institutozaldivar.com/cirugia-intraocular/.

6. **Essilor.** Para ver mejor el mundo. [En línea] [Citado el: 25 de Enero de 2013.] http://www.essilor.es/Paginas/Glossary.aspx?l=g.

7. **SALUD.** Salud Bio.Medicina Natural. [En línea] [Citado el: 25 de Enero de 2013.] http://saludbio.com/tratamientos-de/30586.

8. **Pérez, Miguel Ángel Puig.** Revista Cubana de Oftalmología. [En línea] [Citado el: 25 de Enero de 2013.] http://scielo.sld.cu/scielo.php?pid=S0864-21761999000200001&script=sci_arttext.

9. **Ing.Joselín Milo Pérez, Ing.Yandy Hernández.** *Implementación de los servicios de cirugía del cristalino, cirugía refractiva y pterigium del bloque quirúrjico oftalmológico y los módulos de administración y configuración.* La Habana : s.n.

10. **Rabelo, Ing. Alexander Rodríguez.** *CESIM_BQO_0120_Arquitectura de Software v2.0. Bloque Quirúrgico Oftalmológico.* La Habana : s.n., 2011.

11. **Emetropía.** Emetropía. [En línea] [Citado el: 27 de Enero de 2013.] http://www.emetropia.com/caracteristicas.

12. **VisionDat.** VisionDat. [En línea] [Citado el: 27 de Enero de 2013.] http://www.visiondat.com.

13. **Médica.** Tecno.Canaria.SL. [En línea] [Citado el: 27 de Enero de 2013.] http://www.oftalclinic.com.

14. **Arquitectura.** [En línea] [Citado el: 26 de Mayo de 2013.] http://www.ecured.cu/index.php/Arquitectura_Cliente_Servidor.

15. **Servidor.** [En línea] [Citado el: 26 de Mayo de 2013.] http://www.techpoint.cl/servidor.php.

16. Arquitectura Cliente-Servidor. [En línea] 22 de Octubre de 2008. [Citado el: 31 de Enero de 2013.] http://tombasededatos.wordpress.com/2010/08/22/1-6-arquitectura-cliente-servidor.

17. **González, Carlos Sanchez.** Proyecto de fin de carrera. [En línea] 28 de Septiembre de 2004. [Citado el: 27 de Enero de 2013.] http://oness.sourceforge.net/proyecto/html/ch03s02.html.

18. [En línea] http://www.fdi.ucm.es/profesor/jpavon/poo/2.14.mvc.pdf.

19. Arquitectura 3 capas. [En línea] http://www.desarrolloweb.com/wiki/mvc-modelo-vista-controlador.html.

20. Arquitectura 3 capas. [En línea] http://davidjguru.com/2010/02/08/la-arquitectura-de-tres-capas-introduccion.

21. Curso Java. [En línea] http://tikal.cifn.unam.mx.

22. Arquitectura 3 capas. [En línea] http://davidjguru.com/2010/02/08/la-arquitectura-de-tres-capas-introduccion.

23. Rich Faces. [En línea] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro.

24. Ajax. [En línea] 2007. http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=Ajax4Jsf.

25. Facelets. [En línea] 2008. http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=migrateJSF2Facelets.

26. **Domínguez, Orestes.** Seam. [En línea] 4 de Febrero de 2008. http://seamcity.madeinxpain.com/archives/que-es-jboss-seam.

27. Hibernate. [En línea] http://www.dosideas.com/wiki/Hibernate.

28. Enterprise. [En línea] 2008. http://j2ee-manununuwi.blogspot.com/2008/01/ejb3.html.

29. Curso. Java. [En línea] 14 de Septiembre de 2008. http://www.coplec.org/2008/09/14/java-persistence-api-jpa.

30. Java. [En línea] http://enciclopedia.us.es/index.php/Java_2_Enterprise_Edition.

31. Java Runtime. [En línea] http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre.

32. JBoss Aplication. [En línea] Octubre de 2011. http://unaaldia.hispasec.com/2011/10/un-gusano-se-aprovecha-de-servidores_24.html.

33. **Pressman., Roger.** Ingeniería de Software.Un enfoque práctico. 2002.

34. RUP. [En línea] Julio de 2007. http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm.

35. Lenguaje Unificado de Modelado. [En línea] http://profesores.fi-b.unam.mx/carlos/aydoo/uml.html.

36. Eclipse Foundation. [En línea] 2004. http://wiki.eclipse.org/FAQ_What_is_Eclipse%3f.

37. JBoss Tools. [En línea] http://amap.cantabria.es/confluence/display/DEV/JBoss+Tools.

38. Descarga, descubre y comparte. [En línea] http://postgresql.uptodown.com.

39. pgAdmin PostgresSQL Tools. [En línea] http://www.pgadmin.org..

40. Visual Paradigm. [En línea] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p.

41. Requisitos. [En línea] http://www.ecured.cu/index.php/Requisitos_no_funcionales.

42. Requerimiento. [En línea] http://www.ecured.cu/index.php/Flujo_de_Trabajo_Requerimiento.

43. **Campa, María Karina.** [En línea] 10 de Marzo de 2009. http://mood.itdurango.edu.mx/mod/forum/discuss.php?d=50.

44. Estándar de codificación. [En línea] http://informatica-y-tecnologia-es.blogspot.com/2011/05/importancia-de-la-codificacian-de-las.html.

45. **Lezcano, German.** Patrones de Software. [En línea] http://germanlescano.wordpress.com/2010/03/10/patrones-software.

46. Patrones. [En línea] http://ingenieriasw2.blogspot.com/p/patrones-de-diseno-y-frameworks.html.

47. **Curbelo, Ing. Dunia Santos.** *Desarrollo de las funcionalidades asociadas a las consultas Oculoplastia y Cirugía Implanto-Refractiva para el producto alas BQO.* La Habana : s.n., 2012.

48. **Quintero, Juan Bernardo.** La ingeniería de software aplicada a las bases de datos. [En línea] [Citado el: 5 de Marzo de 2013.] http://docencia.udea.edu.co/ingenieria/ArquitecturaSoftware/documentos/LaingenieriadesoftwareenlasBD.pdf .

49. **Liset Maria Barreras, Leydis Aguilera Morales.** *Componente para grabado en media de estudios imagenológicos.* La Habana : s.n.

50. **Domínguez, Orestes.** Seam. [En línea] 4 de Febrero de 2008. http://seamcity.madeinxpain.com/archives/que-es-jboss-seam.

BIBLIOGRAFÍA

- **Ajax.** [En línea] 2007. http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=Ajax4Jsf.
- **Arquitectura.** [En línea] [Citado el: 26 de Mayo de 2013.] http://www.ecured.cu/index.php/Arquitectura_Cliente_Servidor.
- **Arquitectura 3 capas**. [En línea] http://www.desarrolloweb.com/wiki/mvc-modelo-vista-controlador.html.
- **Arquitectura 3 capas**. [En línea] http://davidjguru.com/2010/02/08/la-arquitectura-de-tres-capas-introduccion.
- **Arquitectura Cliente-Servidor.** [En línea] 22 de Octubre de 2008. [Citado el: 31 de Enero de 2013.] http://tombasededatos.wordpress.com/2010/08/22/1-6-arquitectura-cliente-servidor.
- **Campa, María Karina.** [En línea] 10 de Marzo de 2009. http://mood.itdurango.edu.mx/mod/forum/discuss.php?d=50.
- **Curbelo, Ing. Dunia Santos.** *Desarrollo de las funcionalidades asociadas a las consultas Oculoplastia y Cirugía Implanto-Refractiva para el producto alas BQO.* La Habana : s.n., 2012.
- **Curso. Java**. [En línea] 14 de Septiembre de 2008. http://www.coplec.org/2008/09/14/java-persistence-api-jpa.
- **Curso Java**. [En línea] http://tikal.cifn.unam.mx.
- Descarga, descubre y comparte. [En línea] http://postgresql.uptodown.com.
- **Domínguez, Orestes.** Seam. [En línea] 4 de Febrero de 2008. http://seamcity.madeinxpain.com/archives/que-es-jboss-seam.
- **Dr.Zaldívar.** Oftalmología avanzada para el mundo. [En línea] [Citado el: 25 de Enero de 2013.] http://www.institutozaldivar.com/cirugia-intraocular/.

- **Eclipse Foundation**. [En línea] 2004. http://wiki.eclipse.org/FAQ_What_is_Eclipse%3f.
- **Enterprise**. [En línea] 2008. http://j2ee-manunuwi.blogspot.com/2008/01/ejb3.html.
- **Emetropía.** Emetropía. [En línea] [Citado el: 27 de Enero de 2013.] http://www.emetropia.com/caracteristicas.
- **Essilor.** Para ver mejor el mundo. [En línea] [Citado el: 25 de Enero de 2013.] http://www.essilor.es/Paginas/Glossary.aspx?l=g.
- **Estándar de codificación**. [En línea] http://informatica-y-tecnologia-es.blogspot.com/2011/05/importancia-de-la-codificacian-de-las.html.
- **Facelets**. [En línea] 2008. http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=migrateJSF2Facelets.
- **Font, Lic. Jaime Cruz.** Informática Médica. [En línea] Agosto de 2009. [Citado el: 20 de Noviembre de 2012.] www.cocmed.sld.cu/no44/n44ori1.htm.
- **González, Carlos Sanchez.** Proyecto de fin de carrera. [En línea] 28 de Septiembre de 2004. [Citado el: 27 de Enero de 2013.] http://oness.sourceforge.net/proyecto/html/ch03s02.html.
- **Hibernate**. [En línea] http://www.dosideas.com/wiki/Hibernate.
- **Hoyos, Dr. Fernando Bergaz de.** Kateron Systems. [En línea] [Citado el: 22 de Noviembre de 2012.] http://www.kanteron.com..
- **Ing. Alexander Rodríguez Rabelo, Ing Joselín Miló Pérez, Ing Mauricio Espinosa Robaina y Ing Miriam Rodríguez Reyes.***Automatización del cálculo del poder dióptrico de los lentes intraoculares para las operaciones oftalmológicas.* La Habana : s.n.
- **Ing.Joselín Milo Pérez, Ing.Yandy Hernández.** *Implementación de los servicios de cirugía del cristalino, cirugía refractiva y pterigium del bloque quirúrjico oftalmológico y los módulos de administración y configuración.* La Habana : s.n.

- **Java**. [En línea] http://enciclopedia.us.es/index.php/Java_2_Enterprise_Edition.
- **Java Runtime**. [En línea] http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre.
- **JBoss Aplication**. [En línea] Octubre de 2011. http://unaaldia.hispasec.com/2011/10/un-gusano-se-aprovecha-de-servidores_24.html.
- **Lenguaje Unificado de Modelado**. [En línea] http://profesores.fi-b.unam.mx/carlos/aydoo/uml.html.
- **Liset Maria Barreras, Leydis Aguilera Morales.** *Componente para grabado en media de estudios imagenológicos.* La Habana : s.n.
- **Lezcano, German.** Patrones de Software. [En línea] http://germanlescano.wordpress.com/2010/03/10/patrones-software.
- **Médica.** Tecno.Canaria.SL. [En línea] [Citado el: 27 de Enero de 2013.] http://www.oftalclinic.com.
- **Nodales, Edith María Ballate.** La Oftalmología. [En línea] 31 de Octubre de 2009. [Citado el: 22 de Enero de 2013.] http://www.articulo.org/articulo/9408/la_oftalmologia.html.
- **Patrones**. [En línea] http://ingenieriasw2.blogspot.com/p/patrones-de-diseno-y-frameworks.html.
- **Pérez, Miguel Ángel Puig.** Revista Cubana de Oftalmología. [En línea] [Citado el: 25 de Enero de 2013.] http://scielo.sld.cu/scielo.php?pid=S0864-21761999000200001&script=sci_arttext.
- **pgAdmin** PostgresSQL Tools. [En línea] http://www.pgadmin.org.
- **Pressman., Roger.** Ingeniería de Software.Un enfoque práctico. 2002.
- **Quintero, Juan Bernardo.** La ingeniería de software aplicada a las bases de datos. [En línea] [Citado el: 5 de Marzo de 2013.]

http://docencia.udea.edu.co/ingenieria/ArquitecturaSoftware/documentos/LaingenieriadesoftwareenlasBD.pdf .

- **Rabelo, Ing. Alexander Rodríguez.** *CESIM_BQO_0120_Arquitectura de Software v2.0. Bloque Quirúrgico Oftalmológico.* La Habana : s.n., 2011.
- **Requisitos**. [En línea] http://www.ecured.cu/index.php/Requisitos_no_funcionales.
- **Requerimiento**. [En línea] http://www.ecured.cu/index.php/Flujo_de_Trabajo_Requerimiento.
- **RichFaces**.[Enlínea] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=richFacesJsfIntro.
- **RUP.** [En línea] Julio de 2007. http://www.utvm.edu.mx/OrganoInformativo/orgJul07/RUP.htm.
- **SALUD.** Salud Bio.Medicina Natural. [En línea] [Citado el: 25 de Enero de 2013.] http://saludbio.com/tratamientos-de/30586.
- **Servidor.** [En línea] [Citado el: 26 de Mayo de 2013.] http://www.techpoint.cl/servidor.php.
- **VisionDat.** VisionDat. [En línea] [Citado el: 27 de Enero de 2013.] http://www.visiondat.com.
- **VisualParadigm**.[Enlínea] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p.

ANEXOS

Anexo 1: Realizar cálculo

Figura A 1: Realizar cálculo

Anexo 2: Personalizar fórmula

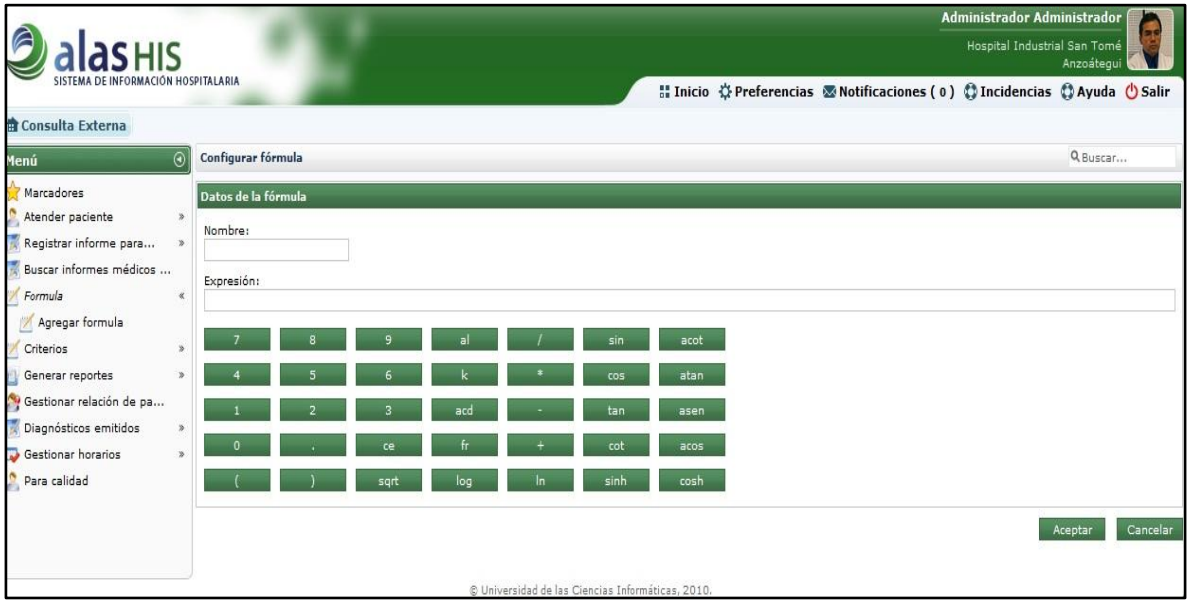

Figura A 2: Personalizar fórmula

Anexo 3: Personalizar criterio

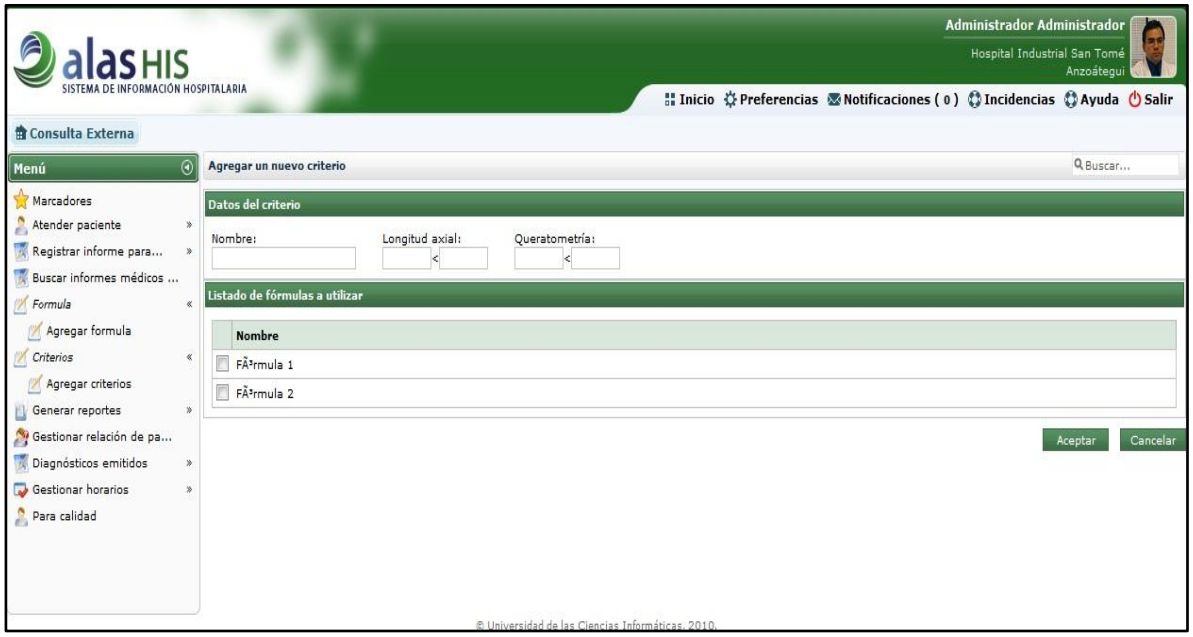

Figura A 3: Personalizar criterio

GLOSARIO TÉRMINOS

1. **Lente Intraocular:** es el lente que se coloca en el ojo en sustitución del cristalino cuando se realiza una cirugía de extracción del mismo y el oftalmólogo determina la necesidad de este. El tipo de lente y el poder dióptrico del mismo son determinados por el especialista a partir de un conjunto de reglas y fórmulas preestablecidas con este fin.
2. **Poder dióptrico:** Poder de refracción de una lente óptica, medido en dioptrías.
3. **Globo ocular:** Órgano de la vista, de color blanquecino y forma esférica: el globo ocular está formado por el iris, la pupila, la retina y el cristalino.
4. **Cristalino:** El cristalino es un componente del ojo con forma de lente biconvexa que está situado tras el iris y delante del humor vítreo. Su propósito principal consiste en permitir enfocar objetos situados a diferentes distancias. Es el más débil de los elementos refractivos que conforman el sistema óptico humano, su poder dióptrico equivale a unas veintiuna dioptrías positivas.
5. **Consulta de Microcirugía del Cristalino:** En esta consulta se atienden todos los pacientes que presentan dificultades con el correcto funcionamiento de este órgano, encargado de enfocar los rayos de luz que entran al globo ocular sobre la retina.
6. **TCP/IP:** Protocolo de control de transmisión /Protocolo de Internet o un conjunto de protocolos. Representa las reglas de comunicación para internet y se basa en las direcciones IP de cada equipo en la red para poder enrutar paquetes de datos.
7. **GRASP:** Patrones de software para la asignación general de responsabilidades.