Universidad de las Ciencias Informáticas

Facultad 2

Título: Configuración general del Sistema de Información Hospitalaria del Centro de Informática Médica

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

Autores: Yazmin Suárez Suárez
Fiax Lux Deschapelles La Rosa

Tutores: Ing. Anny Campos Cosme
Ing. Osmín Pérez Morales

La Habana, junio de 2015
"Año 57 de la Revolución"

"El aprendizaje es experiencia, todo lo demás es información."

Albert Einstein

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informática los derechos patrimoniales del mismo, con carácter exclusivo.

Para que así conste firmamos la presente a los _ días del mes de junio del 2015.

Yazmin Suárez Suárez

_____

Firma del autor

Fiax Lux Deschapelles La Rosa

_____

Firma del autor

Ing. Anny Campos Cosme

_____

Firma del tutor

Ing. Osmín Pérez Morales

_____

Firma del tutor

DATOS DE CONTACTO

Tutores:

Ing. Anny Campos Cosme: Ingeniera en Ciencias Informáticas, graduada en la Universidad de las Ciencias Informáticas en el año 2010. Ha participado en varios proyectos de desarrollo vinculados al perfil de salud. Actualmente labora en el Departamento de Sistemas de Gestión Hospitalaria del Centro de Informática Médica (CESIM), desempeñándose como Analista del mismo.

Correo electrónico: acampos@uci.cu

Ing. Osmín Pérez Morales: Ingeniero en Ciencias Informáticas, graduado en la Universidad de las Ciencias Informáticas en el año 2010. Se ha desempeñado como Desarrollador de Software en el Centro de Informática Médica (CESIM). Ha participado en varios proyectos de desarrollo vinculados al perfil de salud y ha sido tutor de otros trabajos de diploma.

Correo electrónico: opmorales@uci.cu

AGRADECIMIENTOS

Yazmin:

Quiero agradecer a mi mamá quien ha sido mi apoyo, mi fuerza, mi ejemplo a seguir, la luz de mis días. A mi papá que siempre me ha impulsado a lograr mis metas. A mi familia y amigos quienes me han apoyado en todo momento y soportado mis necedades. A mi compañero de tesis, a mis tutores y a todas las personas que me han brindado su apoyo a lo largo de mi formación profesional.

Fiax Lux:

Le agradezco a mi madre Edith, la reina y protectora de mi corazón. A mi paradigma de hombre, mi ídolo y consejero, mi padre. Le agradezco a mi hermana Jadelín, la piedra preciosa de la familia y madre del bebé más lindo que haya sostenido en mis manos, mi sobrino JJ. Y a mi mayor cómplice, mi hermano mayor Johans, por todos nuestros secretos, las lecciones de la vida y por ser torre de mis tempestades. A mi novia Nadiezka, por el apoyo incondicional y el amor brindado, a mis tutores, profesores, amigos y todos los que de una forma u otra contribuyeron a mi formación como profesional.

DEDICATORIA

Yazmin:

Dedicado a mi mamá y a mi hermanita, lo más hermoso que tengo en la vida.

Fiax Lux:

Dedicado a mi sobrino Jason Josiel, que le sirva de ejemplo y aliciente para su futuro tan incierto, como que hoy pueda leer estas palabras y tan prometedor como lo que pueda entender mañana de ellas.

RESUMEN

El Sistema de Información Hospitalaria del Centro de Informática Médica cuenta con el módulo Configuración para su administración y seguridad de acceso. Actualmente este gestiona la información de múltiples instituciones hospitalarias, pero su implementación no permite satisfacer todas las necesidades reales que demanda el mismo.

Con el objetivo de lograr una correcta gestión y administración de las configuraciones generales, se determina la necesidad de desarrollar el módulo Configuración general del Sistema de Información Hospitalaria del Centro de Informática Médica.

Para guiar el desarrollo de las funcionalidades se hace uso del Proceso Unificado de Desarrollo. Se utiliza Java como lenguaje de programación y JBoss Developer Studio como herramienta de desarrollo. El servidor de aplicaciones seleccionado es JBoss Server y como Sistema Gestor de Base de Datos PostgreSQL. Se incorporan a estas herramientas *frameworks* y librerías que facilitan el trabajo con las mismas.

El módulo desarrollado se espera que brinde centralización en la configuración. Además de interfaces visuales apropiadas e intuitivas en las funcionalidades y un mayor grado de usabilidad y facilidad de mantenimiento en la configuración general del sistema.

Palabras clave: configuración, Sistema de Información Hospitalaria, gestión de la información.

ÍNDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

El modelo de sociedad actual, está determinado por la consciente importancia de la construcción del conocimiento y por la preocupación de una efectiva gestión de la información. Este gran cambio, se debe en parte al desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC) que se ha convertido en uno de los recursos más importantes del hombre moderno. Las TIC favorecen la transmisión e intercambio de datos, información y conocimientos a los cuales se pueden acceder de manera rápida y sencilla. Por tales razones, en los últimos años, las instituciones del mundo han integrado estos avances para desarrollar todas las esferas de la sociedad. Las ciencias médicas es uno de los campos cuya proyección actual se ha ido orientando hacia una incorporación progresiva de las nuevas tecnologías. En los hospitales, se aplica cada vez más la tecnología médica digital, incluyendo los Sistemas de Información Hospitalaria (HIS, por sus siglas en inglés).

Un HIS es un sistema de gestión orientado a satisfacer las necesidades de generación de la información, almacenamiento, procesamiento e interpretación de datos de las instituciones hospitalarias, lo cual permite una mayor optimización de los recursos, tanto humanos como materiales (1). Estos sistemas están conformados por módulos que responden a cada una de las áreas que pertenecen a una institución hospitalaria. Entre los módulos que se pueden encontrar en un HIS figuran: Hospitalización, Emergencias, Admisión, Laboratorio, Almacén y Configuración.

El Centro de Informática Médica (CESIM) perteneciente a la Universidad de las Ciencias Informáticas (UCI) en Cuba, diseñó e implementó un HIS en el año 2010. En aquel entonces el HIS del CESIM estaba concebido para gestionar la información generada en una institución hospitalaria y el módulo Configuración que presentaba, permitía la administración del sistema, además de garantizar su seguridad de acceso y gestionar cada uno de los departamentos, servicios, habitaciones y camas.

La evolución del HIS del CESIM no se ha detenido, actualmente el sistema es capaz de gestionar la información de múltiples instituciones hospitalarias, pero la configuración que presenta no permite satisfacer todas las necesidades reales que demandan el sistema y sus usuarios, pues fue concebida para configurar una institución hospitalaria y no varias. A lo largo de los años la experiencia ha revelado, que el proceso de escalar de un sistema monoentidad a multientidad sobrecargó el módulo Configuración afectando la usabilidad y el mantenimiento del mismo.

La estructura de diseño del módulo Configuración mezcla en una misma interfaz visual, funcionalidades que permiten gestionar datos generales del sistema y otras que se asocian a configuraciones a nivel de entidad hospitalaria, siendo estas últimas totalmente dependientes de las configuraciones generales. Esta situación dificulta la administración y comprensión del sistema por parte de los usuarios responsables de esta tarea, pues deben identificar y conocer a priori el orden de las funcionalidades a configurar, restándole facilidad de interacción con el sistema.

Es válido señalar además, que la gestión de la seguridad en el sistema se realiza a través del control de acceso a los módulos y funcionalidades de las entidades, concediendo permisos a roles y usuarios. Estos permisos o reglas de seguridad se dividen en lógicos y físicos; donde los lógicos permiten visualizar o no una funcionalidad y los físicos controlan que se invoquen las páginas correspondientes.

Este proceso en la vista lógica se lleva a cabo en una interfaz visual inapropiada y requiere una alta comprensión de cómo opera la seguridad del sistema; ya que a la hora de permitir o denegar un permiso se listan de manera repetida las funcionalidades pertenecientes a cada entidad en forma de árbol jerárquico. Conceder un permiso sobre una funcionalidad en cada nivel de este árbol causa un efecto específico en la seguridad del sistema, además resulta contradictorio para el usuario administrador que se pueda permitir y denegar el acceso sobre una misma funcionalidad. También presentan problemas las reglas de seguridad que se muestran como información de los permisos configurados, pues no son lo suficiente detalladas y en ciertos casos incorrectas, lo cual provoca confusión acerca del estado de la seguridad del sistema.

Por otra parte, el módulo implementa una bitácora que se encarga de almacenar las acciones que se realizan sobre el sistema. Su desarrollo fue realizado dándole la responsabilidad al programador de garantizar que las funcionalidades registren automáticamente dichas acciones. Durante revisiones al sistema se constató, que estas acciones no eran almacenadas en su totalidad y en los casos que se cumplía con lo establecido, las acciones al no estar estandarizadas, eran registradas siguiendo diversos formatos. Se ha comprobado que estos inconvenientes ralentizan el control y dificultan la búsqueda de los accesos al sistema a la hora de realizar auditorías por los administradores. La bitácora además, no cuenta con un criterio de búsqueda por acciones realizadas e imposibilita localizar por cada entidad las operaciones registradas.

Una vez realizado el análisis de la situación actual del módulo Configuración se define como **problema a resolver**: El funcionamiento actual del módulo Configuración del Sistema de Información Hospitalaria del Centro de Informática Médica dificulta la gestión, administración y centralización de las configuraciones generales del sistema.

El **objeto de estudio** está constituido por el proceso de configuración en sistemas de gestión de información hospitalaria, enmarcado en el **campo de acción** el proceso de configuración general del Sistema de Información Hospitalaria del CESIM.

Para solucionar el problema identificado se propone el siguiente **objetivo general**: Desarrollar un módulo para la configuración general del Sistema de Información Hospitalaria del CESIM que permita la correcta gestión, administración y centralización de las configuraciones.

Para dar solución al objetivo planteado se definen las siguientes tareas de la investigación:

1. Analizar el proceso de configuración en sistemas de gestión de información hospitalaria.
2. Asimilar la arquitectura y pautas definidas para el desarrollo de los artefactos, el diseño y las aplicaciones del Sistema de Información Hospitalaria del CESIM.
3. Generar los artefactos correspondientes a las disciplinas de “Modelado de negocio”, “Gestión de requisitos”, “Diseño” e “Implementación”.
4. Implementar la configuración general del Sistema de Información Hospitalaria del CESIM.

De esta forma se espera que el desarrollo de la nueva configuración general del Sistema de Información Hospitalaria del CESIM, aporte los siguientes beneficios a partir de su puesta en práctica:

1. Configuración centralizada del Sistema de Información Hospitalaria del CESIM.
2. Interfaces visuales apropiadas e intuitivas en las funcionalidades de la configuración general.
3. Garantía de registro de todas las acciones realizadas en el sistema.
4. Mayor grado de usabilidad y facilidad de mantenimiento en la configuración general del Sistema de Información Hospitalaria del CESIM.

El presente documento se encuentra estructurado en cuatro capítulos. El primero de ellos, **“FUNDAMENTACIÓN TEÓRICA DE LA INVESTIGACIÓN”**, contiene un estudio del estado del arte así como el ambiente de desarrollo del módulo Configuración. Justificándose además

la tecnología que fue utilizada para su desarrollo. El capítulo 2, **"CARACTERÍSTICAS DEL MÓDULO CONFIGURACIÓN GENERAL"**, contiene un marco conceptual asociado a la información que es manipulada por el sistema. En este se llega a un acuerdo sobre las funcionalidades, requerimientos deseados y el objeto de automatización de la investigación, quedando explícitamente descritos mediante casos de uso del sistema. El tercer capítulo **"DISEÑO DEL MÓDULO CONFIGURACIÓN GENERAL"** se centra en la modelación detallada y la construcción de la estructura del módulo. En el cuarto y último capítulo, **"IMPLEMENTACIÓN DEL MÓDULO CONFIGURACIÓN GENERAL"**, se describe la implementación de las clases y subsistemas en términos de componentes. Se presenta la propuesta de solución para lograr una gestión más eficiente de la seguridad y configuración centralizada para el Sistema de Información Hospitalaria del CESIM.

CAPÍTULO 1. Fundamentación teórica de la investigación

En el presente capítulo se describe y analiza las principales características de la configuración de Sistemas de Información Hospitalaria a nivel mundial. También se especifica la tecnología utilizada en el proceso de desarrollo del módulo Configuración general del HIS del CESIM.

1.1 Sistemas automatizados existentes vinculados al objeto de estudio

Durante la investigación se realiza un análisis de distintos Sistemas de Información Hospitalaria con el objetivo de obtener una comprensión del proceso de configuración ejecutado en cada uno. Los sistemas analizados son:

SIHO

El Sistema Integral Hospitalario (SIHO) creado por el grupo Sistemas de Información Consultores (SIC) en México, es una solución modular que tiene como objetivo fundamental, dar soporte a la operación y administración de una institución hospitalaria. Se basa principalmente en el acoplamiento de las dos grandes áreas de un hospital: la clínica y la administrativa dentro de esta última se encuentra el módulo Sistemas y Seguridad.

El módulo Sistemas y Seguridad está concebido para que el departamento de sistemas controle el monitoreo de las transacciones que se realizan en los distintos departamentos. Cuenta con diferentes niveles de seguridad en el acceso a cada una de las opciones. Es posible configurar usuarios según sus niveles, funciones y responsabilidades. Cada vez que un usuario hace uso del SIHO, queda registrado quién realizó dicha transacción, la fecha y la hora, pudiendo de esta forma rastrear cada movimiento tanto clínico como administrativo. Cuenta con una doble seguridad para el expediente clínico, de tal forma que si un médico o una enfermera inician una captura, debe ser el mismo usuario quien lo termine. (2)

SIGH

El Sistema Integral de Gestión Hospitalaria (SIGH) desarrollado por Digitech, empresa aliada del Grupo Invercasa en Nicaragua, ofrece el módulo de Configuración y Seguridad. Este módulo comprende un alto control y manejo del acceso a cada uno de los módulos del sistema, garantizando mayor seguridad y control a la información médica de la institución. Dentro de sus funcionalidades se encuentra la configuración de clínicas, médicos y los catálogos generales del sistema.

La configuración de clínicas ofrece al usuario la opción de administrar la creación y manejo de clínicas con el objetivo de proteger y garantizar que las operaciones que realizarán los

diferentes usuarios sean específicamente en las clínicas donde tienen definidos sus privilegios. Esta subdivisión permite poder administrar la información que visualizarán los usuarios. (3)

PhPhClínica

Es un Sistema de Información Hospitalaria desarrollado en el Hospital Provincial Universitario "Arnaldo Milián Castro" en Cuba, el cual está conformado por diferentes módulos de los cuales dos son de configuración y administración del sistema. Además existen cinco niveles de seguridad de usuarios para que estos no tengas los mismos privilegios a la hora de interactuar con el sistema. Los niveles de seguridad son:

- Master-admin: El usuario tiene acceso pleno al sistema, puede efectuar y controlar todas las tareas que se realizan en el mismo.
- Super-admin: Este usuario tiene acceso pleno al sistema menos al módulo de administración que se utiliza para controlar los datos se seguridad del sistema.
- Admin: El usuario puede acceder a todo el sistema menos a los módulos de configuración y administración.
- User-avanzado: Este usuario puede acceder a todo el sistema menos a los módulos de configuración, administración y phphclinica-cmd este último se encarga de registrar todos los datos del uso de los medios de diagnósticos.
- User: El usuario solo puede observar y realizar búsqueda de informaciones de los pacientes.

En el módulo de Configuración se puede administrar el sistema para su uso, todos los datos que son configurados aquí son utilizados en los diferentes formularios de los módulos. Este permite además el registro de nuevos usuarios para el sistema.

Phphclínica deja guardado en la base de datos varias informaciones como es el usuario, el número de IP de la máquina y la fecha de la persona que registró ese dato específicamente. Estos datos solo son guardados en la computadora elegida como servidor. (4)

HIS del CESIM

El Sistema de Información Hospitalaria del Centro de Informática Médica se encarga del control de las actividades de salud orientadas a los pacientes, permitiendo además gestionar y controlar los recursos de cada una de las áreas de una institución de salud. Los módulos del

sistema son independientes y pueden ser elegidos para su instalación en dependencia de la necesidad de su uso en la entidad. Contiene además un conjunto de funcionalidades que permiten crear, modificar o eliminar determinada información, posibilita realizar búsquedas simples y avanzadas, así como generar reportes que consolidan la información gestionada por cada módulo.

Está concebido como una aplicación web de gestión hospitalaria que administra diferentes entidades y puede ser desplegado en cada uno de los centros de salud, con una base de datos local encargada del almacenamiento y reporte de los datos generados por los distintos módulos instalados. Además permite que todos los sistemas realicen réplica de la información hacia un Centro de Datos que permite contar con un repositorio centralizado de toda la información clínica del paciente, la cual puede ser accedida desde cualquier otro centro de salud autorizado.

Entre los módulos que presenta se encuentra el módulo Configuración, el cual permite a través del administrador la gestión de las configuraciones y de los privilegios de los usuarios en el sistema. Las principales funcionalidades de la configuración del sistema son:

- Configurar funcionalidades: permite visualizar todos los módulos y submódulos creados, mostrando de cada uno de ellos sus funcionalidades, las cuales pueden estar agrupadas por categorías. Brinda la posibilidad de adicionar, modificar, eliminar, editar el orden y cambiar la visibilidad de las funcionalidades y categorías. Estas funcionalidades son comunes para todas las entidades en el sistema.
- Administrar seguridad: facilita la definición de los permisos y niveles de acceso por roles o usuarios a las funcionalidades de cada uno de los módulos. Estos permisos o reglas de seguridad se dividen en lógicos y físicos; donde los lógicos permiten visualizar o no una funcionalidad y los físicos controlan que se invoquen las páginas correspondientes. En ambas vistas el procedimiento para permitir o denegar usuarios y roles es el mismo y las reglas de seguridad configuradas se muestran como información. Las funcionalidades en la vista lógica se listan de manera repetida en forma de árbol por cada entidad en donde los niveles dependen de la estructura de esta. Denegar el acceso a las funcionalidades en el último nivel causará efecto local en la entidad, para lograr un alcance en todo el sistema deben concederse los permisos en niveles superiores.
- Gestionar entidades: posibilita que se administren las entidades del sistema, los departamentos clínicos, otros departamentos y otras entidades conocidas. Además de crear, modificar, eliminar y asociar o desasociar las entidades del sistema, las cuales

incluyen la gestión innecesaria de las asignaciones financieras, las actividades de investigación y docencia así como las partidas presupuestarias.

- Configurar referencias: propicia que se gestionen las referencias de entidades del sistema para asociarlas con otras entidades y los servicios correspondientes.
- Seguridad avanzada: facilita gestionar el certificado raíz, los certificados de confianza, las identidades digitales y los servidores proxy y sello de tiempo.
- Usuarios y roles: brinda la posibilidad de gestionar los usuarios y roles del sistema, así como los médicos, enfermeros y bioanalistas. Además facilita relacionar los usuarios a las entidades que contiene el sistema, siendo este proceso defectuoso cuando no se han creados departamentos y servicios a la entidad asignada.
- Departamentos clínicos: propicia la creación de departamentos y servicios en el sistema, de cada uno de ellos se puede modificar la información asociada o eliminar la misma. Los servicios creados son asociados a los diferentes departamentos según sea la distribución de la institución.
- Gestionar camas: facilita crear las camas de la institución, especificando datos como descripción, tipo de cama, estado de cama, habitación y categoría. Esta información puede ser modificada o eliminada según el interés del usuario. Además brinda la posibilidad de organizar las camas por ubicaciones o servicios y muestra el mapa de las ubicaciones físicas de las camas existentes.
- La Bitácora de sucesos: posibilita visualizar y buscar las trazas de las acciones de los usuarios y las modificaciones de estos a los datos. Presenta criterios de búsqueda que no son exhaustivos y la información de las trazas incluye datos importantes que no se garantizan que sean registrados por el sistema.
- Configurar anillo: permite modificar los datos del anillo donde se encuentra el sistema.
- Configurar módulos: brinda la posibilidad de administrar los módulos pudiendo activar o desactivar aquellos que hayan sido seleccionados.

Todos estos Sistemas de Información Hospitalaria reflejan las tendencias actuales de los procesos que se manejan en la configuración. Tienen en común que son sistemas orientados al paciente y que la gestión de la información que se genera en las instituciones hospitalarias se realiza de manera similar. El SIHO y el SIGH al igual que el HIS del CESIM presentan un

módulo encargado de la configuración y control de la seguridad, donde se define a cada usuario su privilegio en el sistema de acuerdo a la responsabilidad que ocupa. Los dos primeros sistemas antes mencionados no describen el nivel que posee el usuario que realiza esta tarea y si ese nivel en cuestión es gestionable para otros usuarios. Por otra parte, PhPhClínica contiene dos módulos encargados de la administración del sistema y define cinco niveles de seguridad. Esta estructura permite administrar la información referente a las entidades y los datos generales del sistema de manera independiente, siendo apropiada para minimizar la sobrecarga existente en el módulo Configuración del HIS del CESIM.

1.2 Tecnología a utilizar

En este epígrafe se especifica la tecnología que se utiliza en el proceso de desarrollo de la solución, brindando una mejor comprensión de la misma. Esta fue definida para el desarrollo de las aplicaciones del Sistema de Información Hospitalaria del CESIM.

XHTML 1.0: El Lenguaje Extensible de Marcado de Hipertexto (XHTML por sus siglas en inglés) es la versión de Lenguaje de Etiquetado Extensible (XML por sus siglas en inglés) del Lenguaje de Marcado de Hipertexto (HTML por sus siglas en inglés), por lo que tiene básicamente las mismas funcionalidades pero cumple las especificaciones más estrictas de XML. Es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web. Algunas de las ventajas de XHTML sobre otros formatos son que un mismo documento puede adoptar diseños radicalmente distintos en diferentes aparatos y que proporciona facilidad de edición directa del código y de mantenimiento. (5)

JSF 1.2: Java Server Faces (JSF) es un marco de trabajo (*framework)* basado en el patrón Modelo Vista Controlador (MVC) para aplicaciones Java basadas en web, que simplifica el desarrollo de Interfaces de Usuario (IU) en aplicaciones de Java Enterprise Edition (JEE). JSF usa Java Server Pages (JSP) como la tecnología que permite hacer el despliegue de las páginas, facilita y agiliza el diseño de IU, pues implementa una serie de componentes, estado de los mismos y eventos del lado de servidor. (6)

Facelets 1.1.15.B1: Java Server Facelets es un *framework* para plantillas centrado en la tecnología JSF, lo cual permite que JSP y JSF puedan funcionar conjuntamente en una misma aplicación web ya que estos no se complementan naturalmente. JSP procesa los elementos de la página de arriba a abajo, mientras que JSF dicta su propia prestación (ya que su ciclo de vida está dividido en fases marcadas). Facelets constituye la complementación de JSP y JSF, siendo

una tecnología que se centra en crear árboles de componentes y está relacionado con el ciclo de vida JSF. (7)

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. Mediante este *framework* se puede variar el ciclo de vida de una petición JSF, recargar determinados componentes de la página sin necesidad de recargarla por completo, realizar peticiones automáticas al servidor, control de cualquier evento de usuario, entre otros. (8)

Richfaces 3.3.0.GA: Es una librería de componentes visuales para JSF. Posee un *framework* avanzado para la integración de funcionalidades Ajax en dichos componentes visuales, mediante el soporte de la librería Ajax4JSF. Se integra perfectamente en el ciclo de vida de JSF, incluye funcionalidades Ajax de modo que nunca se muestra el JavaScript y tiene un contenedor Ajax propio, además permite crear interfaces de usuario de manera eficiente y rápida, basado en componentes que están listos para usar y son altamente configurables. Provee varias librerías de componentes como son Core, Ajax y UI, así como administración avanzada de recursos como imágenes, código JavaScript y Hojas de Estilo en Cascada (CSS). (9)

EJB 3: Enterprise Java Bean (EJB) es una de las Interfaces de Programación de Aplicaciones (API por sus siglas en inglés) que permite construir aplicaciones de negocios portables, reutilizables y escalables usando el lenguaje de programación Java. Ofrece estándares para aplicaciones de negocios basadas en componentes, orientadas a objetos y distribuidas.

EJB proporciona un modelo distribuido y estándar de componentes que se ejecutan en el servidor. El objetivo de estos es dotar al programador de un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad, entre otros), para centrarse en el desarrollo de la lógica de negocio en sí. El hecho de estar basado en componentes permite que éstos sean flexibles y sobre todo reutilizables. (10)

Seam 2.1.1: Es un *framework* de aplicaciones que unifica los modelos de componentes de JSF y EJB, que proporciona un modelo de programación simplificado para las aplicaciones empresariales basadas en la Web. Seam permite enlazar sus componentes EJB directamente a páginas JSF. Se integra con librerías de controles de código abierto basadas en JSF, además de integrar múltiples tecnologías proporcionando un conjunto de herramientas útiles para

desarrollar cualquier aplicación. Seam también se integra perfectamente con otros *frameworks* como: RichFaces, ICE Faces, MyFaces, Hibernate y Spring. (11)

Hibernate 3.0: Es un *framework* que agiliza la relación entre la aplicación y la base de datos, es una herramienta de Mapeo objeto-relacional (ORM), que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML). Hibernate no sólo se encarga de la asignación de clases Java a tablas de bases de datos (y de tipos de datos Java con tipos de datos SQL), sino que también proporciona consulta de datos e instalaciones de recuperación.

Como todas las herramientas de su tipo, Hibernate busca solucionar el problema de la diferencia entre los dos modelos de datos coexistentes en una aplicación: el usado en la memoria de la computadora (orientación a objetos) y el usado en las bases de datos (modelo relacional). Le permite a la aplicación manipular los datos de la base de datos operando sobre objetos, es decir, con todas las características de la programación orientada a objetos. Hibernate convertirá los datos entre los tipos utilizados por Java y los definidos por SQL. (12)

JEE 6: Java Enterprise Edition (JEE) es la plataforma de programación para desarrollar y ejecutar software que permite utilizar arquitecturas de N capas distribuidas. Se basa ampliamente en componentes de software modulares y se ejecuta sobre un servidor de aplicaciones. Su lenguaje de programación Java es de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo. Esta plataforma cuenta además con una interfaz de aplicación para la persistencia y acceso a los datos. (13)

Java: El lenguaje de programación utilizado es Java, el cual es un lenguaje de propósito general, concurrente, orientado a objetos y basado en clases que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Gracias a su versatilidad y portabilidad, Java permite desarrollar software en una plataforma y ejecutarlo en otras. Además propicia la creación de programas que funcionen en un navegador y servicios web, así como desarrollar aplicaciones para servidores y procesamiento de formularios XHTML. (14)

Algunas de las características de Java son:

- Es un lenguaje de programación que ofrece la potencia del diseño orientado a objetos con una sintaxis fácilmente accesible y un entorno robusto y agradable.
- Proporciona un conjunto de clases potente y flexible.
- Facilita la utilización de aplicaciones que se pueden incluir directamente en páginas Web.
- Es gratis, universal y de fácil distribución.

1.3.1 Metodología de desarrollo

Una metodología es un proceso que engloba procedimientos, técnicas, documentación y herramientas que se utilizan en la creación de un producto de software. Se va indicando paso a paso lo que se debe hacer para lograr un producto informático. Además se debe especificar las personas que van a participar en el proceso así como el papel que van jugar en él (15). A continución de explica la metodología utilizada como guía en la confección del trabajo.

RUP: el Proceso Unificado de Desarrollo (RUP por sus siglas en inglés) es una forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo) en una organización o equipo de desarrollo de software. Junto con el Lenguaje Unificado de Modelado (UML por sus siglas en inglés), constituye una de las metodologías más utilizadas para el análisis, implementación y documentación de sistemas orientados a objetos.

Es un modelo de software que permite el desarrollo de software a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad. Este proceso de desarrollo de software lo conforman el conjunto de actividades necesarias para trasformar los requisitos funcionales de un usuario en un producto de software. (15)

En RUP se han agrupado las actividades en grupos lógicos en los que se definen nueve flujos de trabajo principales. Los seis primeros son conocidos como flujos de ingeniería y los tres últimos como flujos de apoyo. El ciclo de vida de RUP se caracteriza por ser dirigido por caso de uso, centrado en la arquitectura, iterativo e incremental.

Características de RUP:

- Unifica los mejores elementos de metodologías anteriores.
- Preparado para desarrollar grandes y complejos proyectos.

- Orientado a Objetos.
- Utiliza el UML como lenguaje de representación visual.

UML 1.0: es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. Las propiedades que han hecho de UML un estándar son: la concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividad actual y futura. Las estructuras que soporta tienen sus fundamentos en las tecnologías orientadas a objetos, tales como: clases, componentes y nodos. Representa el comportamiento del sistema a través de casos de uso, diagramas de secuencia y de colaboración. (16)

Este lenguaje de modelado formal permite tener un mayor rigor en la especificación, realizar una verificación y validación del modelo desarrollado, automatizar determinados procesos y generar código a partir de los modelos y a la inversa. Esto último permite que el modelo y el código estén actualizados.

1.3.2 Herramientas de desarrollo

Una herramienta de desarrollo de software es un programa informático que usa un programador para crear, depurar, gestionar o mantener un programa (17). A continuación se describen las herramientas utilizadas en el desarrollo de la solución.

JBoss Developer Studio 7.1: es uno de los entornos de desarrollo comúnmente utilizados, proporciona un rendimiento superior para todo el ciclo de vida de desarrollo. Incluye un amplio conjunto de capacidades de las herramientas y soporte para múltiples modelos y marcos de programación como Java Enterprise Edition 6 y 7, RichFaces, Java Server Faces, Enterprise Java Beans, Java Persistence API, Hibernate, JAX -RS con REST Easy, contextos inyección de dependencia, HTML5, y muchas otras tecnologías. Está totalmente probado y certificado para asegurar que todos sus aditamentos (plug-ins), componentes de tiempo de ejecución y sus dependencias son compatibles entre sí. (18)

PostgreSQL 8.4: es un sistema de gestión de bases de datos objeto-relacional, cuyo código fuente está disponible, se puede modificar y redistribuir libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado y en sus últimas versiones no tiene nada que envidiarle a otras bases de datos comerciales. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará

funcionando. PostgreSQL garantiza concurrencia, para lo cual utiliza la tecnología de Control de Concurrencia Multi-Versión (Multi-Version Concurrency Control (MVCC)), con lo que se logra que ningún lector sea bloqueado por un escritor. Es extensible, soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario (19).

PgAdmin III: es una aplicación gráfica para gestionar el Sistema de Gestión de Base de Datos PostgreSQL la cual cuenta con licencia de código abierto. Está diseñada para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. Incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor y un agente para lanzar scripts programados. La interfaz gráfica soporta todas las características de PostgreSQL y facilita enormemente la administración. La conexión al servidor puede hacerse mediante el protocolo TCP/IP o Unix Domain Sockets, y puede encriptarse mediante SSL (Secure Socket Layer) para mayor seguridad. (20)

Visual Paradigm para UML 8.0: es una herramienta de Ingeniería de Software Asistida por Computadora (CASE por sus siglas en inglés) que soporta el Lenguaje Unificado de Modelado y proporciona asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un sistema. Permite construir todo tipo de diagramas de clases, generar código desde diagramas y generar documentación. Apoya los estándares más altos de las notaciones de Java y de UML. Soporta aplicaciones web, es fácil de instalar y actualizar. Está orientado a la creación de diseños usando el paradigma de programación orientada a objetos. (21)

JBoss Server 4.2.2: es un servidor de aplicaciones implementado en Java por lo que puede ser utilizado en cualquier sistema operativo que soporte este lenguaje. Soporta todas las funcionalidades de J2EE 1.4 e incluye servicios adicionales como *clustering*, *caching* y persistencia. También soporta Enterprise Java Beans 3.0, lo que hace el desarrollo de las aplicaciones mucho más simple. (22)

Sus principales características son:

- Producto de licencia de código abierto sin coste adicional.
- Confiable a nivel de empresa.
- Orientado a arquitectura de servicios.
- Flexibilidad consistente.
- Servicios del middleware para cualquier objeto de Java.

- Soporte completo para Java Management Extensions (JMX).

1.3 Conclusiones del capítulo

- Los procesos gestionados por las funcionalidades del módulo Configuración son separados en dos módulos que administran la configuración referente a una entidad y general del sistema respectivamente.
- El proceso de configuración de los sistemas estudiados y en especial del HIS del CESIM es referencia para el desarrollo y la definición de las funcionalidades a implementar en la configuración general.
- El estudio de la tecnología definida para el desarrollo de las aplicaciones del Sistema de Información Hospitalaria que se utiliza en la implementación del módulo, permite un mejor entendimiento y manejo de la misma.

CAPÍTULO 2. Características del módulo Configuración general

El objetivo de este capítulo es detallar los diferentes elementos que brindan la base teórica y conceptual para el desarrollo de las funcionalidades dentro de los que se encuentra el flujo actual de los procesos. Debido a que los procesos de negocio no tienen una definición clara se resuelve desarrollar un modelo de dominio donde se abarcan los conceptos asociados a la solución y sus relaciones. Además se exponen los requisitos funcionales y no funcionales del sistema así como los diagramas de casos de uso correspondientes.

2.1 Modelo del dominio

Es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés, no describen clases u objetos del software. Consiste en un conjunto de diagramas de clases, sin definición de operaciones. También se le denomina modelo conceptual, modelo de objetos del dominio y modelo de objetos de análisis (23). Con el fin de brindar una mejor comprensión del diagrama del Modelo de dominio se proporcionará a continuación una breve descripción de los conceptos encontrados en el ámbito del problema.

En el sistema actual el administrador es el encargado de la configuración general y de gestionar todas las entidades, las cuales representan los centros destinados a brindar servicios de atención a la salud a una población. Estas pertenecen a una ubicación y tienen módulos que personalizan las diferentes áreas de una institución hospitalaria. Los módulos y submódulos contienen funcionalidades que pueden estar agrupadas en categorías que permiten gestionar la información médica a través de operaciones en el sistema. Las trazas de estas operaciones o acciones pueden ser visualizadas por el administrador si son registradas por los desarrolladores durante la implementación de cada funcionalidad. El administrador gestiona además todos los usuarios, roles y sus privilegios. A continuación se muestra el modelo de dominio con sus conceptos y relaciones.

Diagrama del modelo de dominio

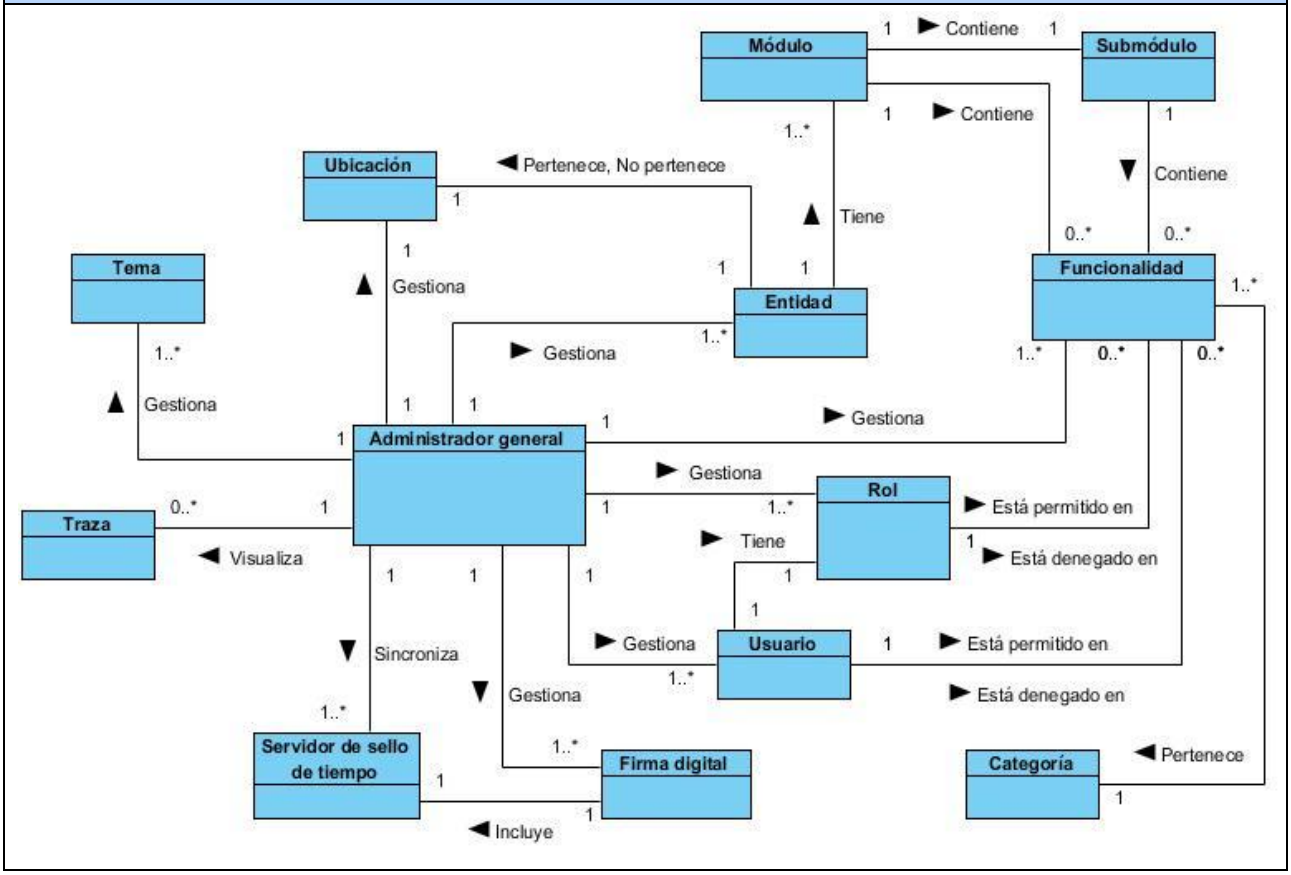

Figura 1: Diagrama del modelo de dominio

2.2 Propuesta del módulo Configuración general

El módulo Configuración general contiene las funcionalidades: Seguridad avanzada, Configurar anillo, Configuraciones personales, Configurar funcionalidades, Usuarios y roles, Administrar seguridad, Gestionar entidades, Configurar referencias y Bitácora.

La funcionalidad Gestionar entidades, permite la administración de estas y las referencias hacia otras instituciones hospitalarias y servicios que contengan. Además se asocian los usuarios administradores encargados de la configuración de cada entidad. El módulo a su vez permite que se definan y gestionen todas las funcionalidades de los módulos que se pueden incluir en dichas entidades.

Otra de las funcionalidades del módulo es Administrar seguridad en la cual se gestiona la permisología de los roles generales del sistema, posibilitando que los usuarios asociados a dichos roles solo tengan acceso sobre los módulos o funcionalidades definidos. Por otra parte se garantiza el registro de todas las trazas de las acciones del sistema y la bitácora cuenta con

criterios de búsquedas que incluye el tipo de operación ejecutado para localizar y visualizar con mayor rapidez dichas acciones y sus modificaciones.

Configuración general brinda la posibilidad de gestionar roles generales, usuarios administradores del sistema y de las entidades. Estos últimos solo tienen acceso a configurar las entidades que le sean asignadas y no pueden acceder a la configuración general; ya que solo están autorizados a este nivel los administradores del sistema.

2.3 Especificación de los requisitos de software

Es la descripción completa del comportamiento del sistema que se va a desarrollar. En esta deben recogerse tanto las necesidades de clientes y usuarios como los requisitos que debe cumplir el sistema a desarrollar para satisfacer dichas necesidades (24). Estos requisitos se dividen en 2 grupos, los funcionales y los no funcionales.

2.3.1 Requisitos funcionales del módulo Configuración general

Luego del análisis de las funcionalidades a desarrollar para cumplir con las necesidades de la configuración general del HIS del CESIM, se determinan los requisitos funcionales que no son más que declaraciones de los servicios que debe proporcionar el módulo, de la manera en que este debe reaccionar a entradas particulares y de cómo se debe comportar en distintas situaciones (25). Algunos de los requisitos funcionales definidos son:

- Administrar entidades del sistema: Permite gestionar todas las entidades del sistema.
- Administrar usuarios: Brinda la posibilidad de gestionar los usuarios administradores del sistema y administradores de entidad.
- Administrar roles: Propicia que se gestionen los roles generales del sistema.
- Configurar funcionalidades: Facilita la configuración de las funcionalidades del sistema.
- Administrar seguridad: Posibilita que se administre la permisología de los roles generales del sistema.
- Buscar traza: Permite buscar las trazas de acciones realizadas en el sistema.

Para visualizar los restantes requisitos funcionales ver anexo “Requisitos funcionales del módulo Configuración general”.

2.3.2 Requisitos no funcionales del módulo Configuración general

Para el correcto desarrollo del módulo Configuración general se deben tener en cuenta los requisitos no funcionales que no son más que restricciones de los servicios o funciones ofrecidos por un sistema. Son aquellos requisitos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la usabilidad, seguridad y hardware. Estos requisitos se agrupan en varios tipos dependiendo de dónde surjan las necesidades del software, de la organización que lo desarrolla o de fuentes externas (25). A continuación se explican los requisitos no funcionales del módulo.

Usabilidad

El módulo está diseñado de forma tal que brinda un acceso fácil y rápido a todas sus funcionalidades, el significado de los íconos y los textos son claros para la persona que interactúa con el mismo facilitando su uso. Las opciones que se proveen son comprensibles, sin explicaciones excesivas sobre su uso, y admiten flujos alternos.

Interfaz de usuario

Las interfaces contienen los datos legibles y bien organizados y están diseñadas siguiendo el documento “Pautas de diseño de interfaz” definido para el diseño del Sistema de Información Hospitalaria del CESIM, logrando una mayor uniformidad en la estructura del módulo. Dispone de menús desplegables que facilitan y aceleran su utilización. La entrada de datos incorrectos es detectada claramente y mostrada al usuario.

Seguridad

Las funcionalidades mantienen la seguridad y control a nivel de usuarios, garantizando el acceso de los mismos sólo a los niveles establecidos de acuerdo a la función o rol que desempeñan y a los permisos que le sean otorgados. Los administradores del sistema son los únicos que pueden acceder al módulo Configuración general. Estos y los administradores de las entidades son los autorizados de cambiar las contraseñas a petición del usuario. Toda entrada de información es validada, de forma que la introducción de datos incorrectos se muestra al usuario especificándole el tipo de error. Las acciones realizadas por los usuarios son registradas en todo momento permitiendo el control de las operaciones en el sistema.

Hardware

Los requerimientos mínimos de hardware para la instalación del HIS del CESIM se definen a continuación:

Servidores

Los servidores, deben tener alta capacidad de procesamiento y redundancia, garantizando movilidad, residencia de la información y las aplicaciones bajo esquemas seguros y confiables.

- Servidores de base de datos: disco duro: 80 GB, tarjeta de red: 100 Mbps, memoria RAM: 8 GB, procesador: 8 núcleos.
- Servidores de aplicaciones disco duro: 80 GB, tarjeta de red: 100 Mbps, memoria RAM: 16 GB, procesador: 8 núcleos.

Dependiendo de la cantidad de estaciones de trabajo a utilizar el sistema, las características de los servidores pueden aumentar. Los servidores de aplicaciones y base de datos inicialmente requerirá 80 GB pero con el tiempo será necesario contar con más espacio disponible para poder almacenar toda la información que genera el sistema dígase documentos clínicos, estudios y base de datos.

Software

Los requerimientos de software para la instalación del HIS del CESIM son: el sistema operativo del servidor debe ser de las distribuciones Linux de 64 bits Debian, Ubuntu u OpenSuse, utilizando la plataforma Java (Máquina Virtual de Java - Java Enterprise Edition), el servidor de aplicaciones JBoss AS y como sistema para la gestión de base de datos PostgreSQL. Los usuarios deben disponer de un navegador web, el cual puede ser Google Chrome , Firefox 3.6 o versiones superiores de estos para conectarse al sistema.

2.4 Modelo de Casos de Uso

Describe las funcionalidades propuestas del módulo. Un caso de uso es una descripción detallada de los pasos o las actividades que deberán realizarse para llevar a cabo un requisito funcional. Cada caso de uso representa un requisito funcional del módulo. Estos son iniciados por un actor que no es más que una entidad externa al sistema que guarda una relación con éste y que le demanda una funcionalidad. Esto incluye a los operadores humanos pero también incluye a todos los sistemas externos (24). Los actores definidos para el módulo son:

Tabla 1: Actores del sistema

Actor	Descripción
Administrador del sistema	Es el encargado de realizar todas las acciones de la configuración general del sistema.

Diagrama de actores


Figura 2: Diagrama de actores

2.4.1 Diagramas de Casos de Uso

Los Diagramas de Casos de Uso (DCU) son una forma de tener una visión general de los casos de uso, sus relaciones con los actores y con otros casos de uso (24). Algunos de los diagramas de casos de uso del módulo son:

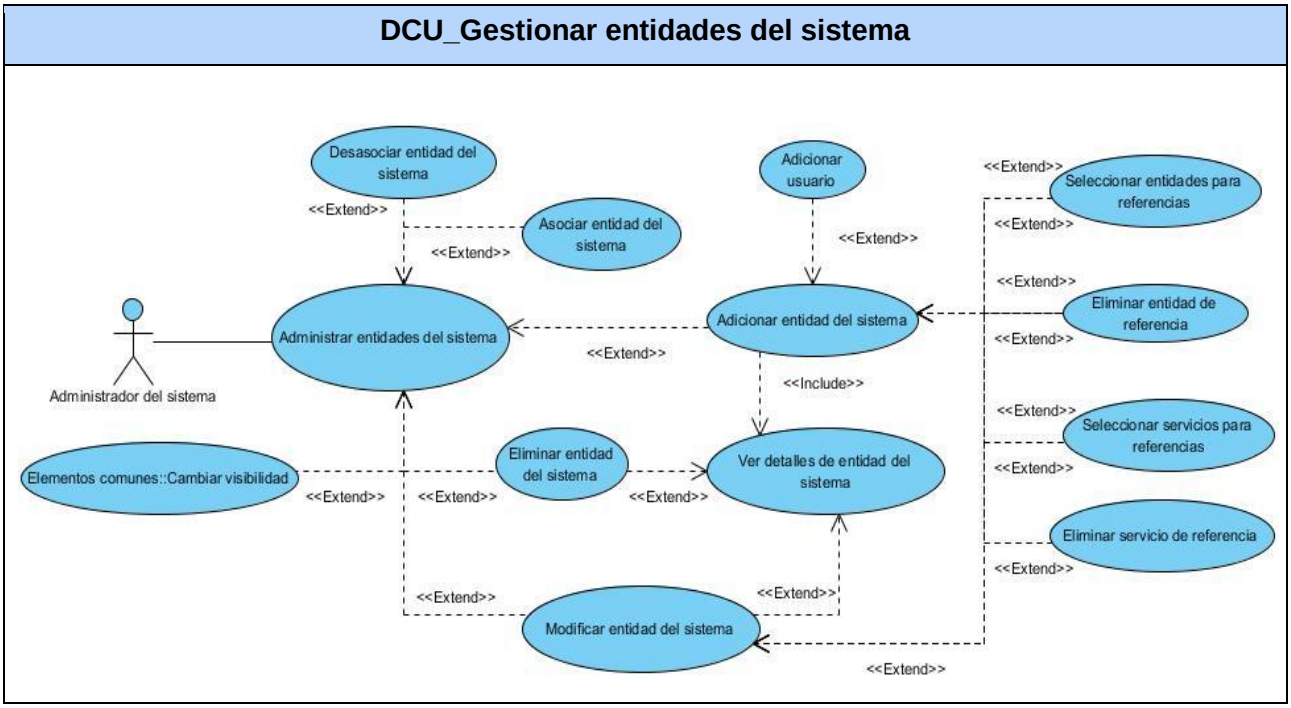

Figura 3: Diagrama de Casos de Uso - Gestionar entidades del sistema

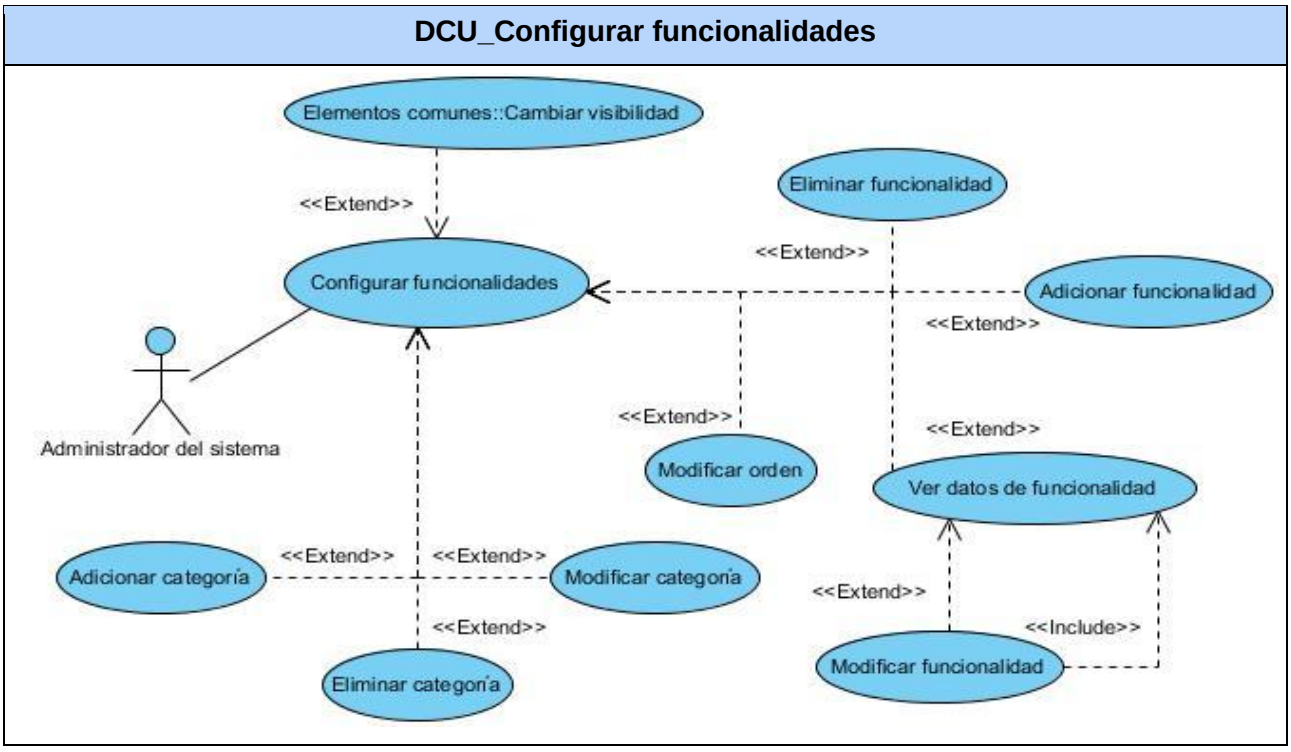

Figura 4: Diagrama de Casos de Uso - Configurar funcionalidades

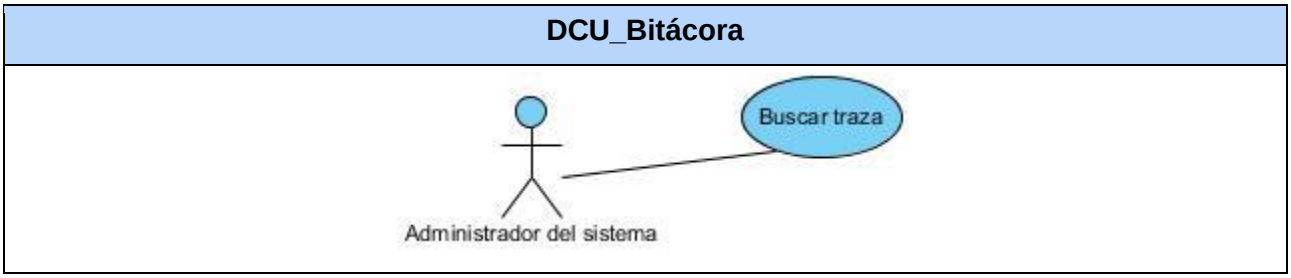

Figura 5: Diagrama de Casos de Uso – Bitácora

Para visualizar los restantes diagramas de casos de uso ver anexo “Diagramas de casos de uso del módulo Configuración general”.

2.4.2 Descripción textual de los casos de uso

A continuación se describen algunos de los casos de uso del módulo Configuración general brindando una mayor comprensión de la solución propuesta.

CU1: Adicionar entidad del sistema.

Tabla 2: Descripción de casos de uso – Adicionar entidad del sistema

Objetivo	Permite crear una Entidad en el sistema.
Actores	Administrador del sistema
Resumen	El caso de uso inicia cuando el actor accede a la opción Adicionar entidad del sistema, el sistema brinda la posibilidad de introducir los datos para crear la Entidad, el actor introduce los datos de la Entidad, el sistema crea la Entidad, el caso de uso termina.
Complejidad	Alta
Prioridad	Crítico
Precondiciones	No existen
Postcondiciones	Se creó una Entidad.

CU2: Configurar funcionalidades.

Tabla 3: Descripción de casos de uso – Configurar funcionalidades

Objetivo	Permite configurar Funcionalidades.
Actores	Administrador del sistema
Resumen	El caso de uso inicia cuando el actor accede a la opción Configurar funcionalidad, el sistema muestra una lista en forma de árbol con los módulos, los submódulos, las categorías y funcionalidades del sistema, el actor selecciona el modulo, submódulo, categoría o funcionalidad y brinda las opciones de adicionar, eliminar,

	cambiar visibilidad y cambiar orden de las categorías y las funcionalidades, el caso de uso termina.
Complejidad	Media
Prioridad	Secundario
Precondiciones	Debe existir al menos un módulo.
Postcondiciones	Se configuró una Funcionalidad por el actor.

CU3: Buscar traza.

Tabla 4: Descripción de casos de uso – Buscar traza

Objetivo	Permite buscar las trazas de las acciones realizadas en el sistema.
Actores	Administrador del sistema
Resumen	El caso de uso inicia cuando el actor accede a la opción Bitácora, el sistema brinda la posibilidad de introducir criterios de búsqueda para localizar las trazas de las acciones realizadas en el sistema, el actor introduce los datos que considera como criterios para realizar una búsqueda, el sistema busca y muestra un listado en forma de árbol con las trazas que cumplen con los criterios de búsqueda, el caso de uso termina.
Complejidad	Media
Prioridad	Secundario
Precondiciones	No existen
Postcondiciones	Se buscaron las trazas correctamente.

2.5 Conclusiones del capítulo

- La descripción del modelo de domino permite un mejor entendimiento del proceso de negocio, a partir del cual se identifican las funcionalidades necesarias para el desarrollo del módulo propuesto.
- El modelo de Casos de Uso descrito permite la identificación de los requisitos funcionales que mayor valor aportan al negocio, lo cual facilita la priorización del requerimiento.

CAPÍTULO 3. Diseño del módulo Configuración general

Este capítulo tiene como objetivo realizar una descripción detallada del módulo propuesto y explicar la arquitectura utilizada en el mismo. Se define el modelo de diseño con sus respectivos diagramas de clases, además de una explicación de los patrones de diseño que se ponen en práctica en el desarrollo del módulo.

3.1 Descripción de la arquitectura

La arquitectura de software de un programa o sistema de computadora, es la estructura de ese sistema, que incluye componentes de software, las propiedades visibles externas de esos componentes y las relaciones que existen entre estos. Consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información. Es considerada el nivel más alto en el diseño de un sistema puesto que establecen la estructura, funcionamiento e interacción entre las partes del software. (24)

Para el desarrollo de la aplicación se hace uso del patrón arquitectónico Modelo Vista Controlador (MVC) el cual se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar la tarea de desarrollo de aplicaciones y su posterior mantenimiento. Este patrón permite la separación de los datos de una aplicación, la interfaz de usuario y la lógica de control, en tres capas: el modelo, para la administración de los datos; la vista, transforma el modelo en una página HTML que permite al usuario interactuar con ella; y el controlador, que se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

El modelo o capa de datos contiene mecanismos para acceder a la información y también para actualizar su estado, se encarga de cargar, modificar, eliminar y persistir la información existente en la base de datos, además de validarla antes de persistirla. Para esto se utilizan los componentes de Hibernate por los que se encuentra constituida la capa, logrando abstraer al desarrollador del gestor de base de datos mediante el mapeo de tablas. Todo esto da la posibilidad de llevar las consultas a un lenguaje de objetos.

La vista o capa de presentación contienen el código del módulo que va a producir la visualización de las interfaces de usuario. Está conformada principalmente por páginas XHTML que obtienen y validan los datos que el usuario provee en cada una de las operaciones que realiza mediante controles JSF, Seam UI, Facelets y RichFaces, cuyo uso enriquece el diseño

de la interfaz de usuario. También realiza el envío y la carga de datos mediante los componentes Ajax4jsf logrando un efecto más agradable y natural al interactuar con el módulo.

El controlador o capa de negocio contiene el código necesario para responder a las acciones que se solicitan en el módulo. Está conformada por clases controladoras encargadas de definir la lógica del negocio, así como manejar y validar los datos capturados en la capa de presentación. Mediante anotaciones que provee el *framework* Seam se les puede especificar a estas clases el contexto en que se encuentran, los cuales definen el estado de los datos y las entidades que manejan. (26)

3.2 Integración con el módulo Configuración entidad

El módulo propuesto gestiona la información general del sistema y delega la responsabilidad de configurar las entidades al módulo Configuración entidad. Estos procesos no se hacen de forma separada, sino que se complementan para obtener finalmente una configuración más centralizada y organizada. La integración comienza cuando en el módulo Configuración general se crean las entidades del sistema, pues a cada una de estas se les hace corresponder un módulo Configuración entidad y al menos un usuario que la administre. Ambos módulos comparten vistas, clases e interactúan sobre mismas tablas en la base de datos. Los problemas de concurrencia son resueltos con el tratamiento de excepciones correspondientes y en el caso de la administración de la seguridad donde se opera sobre una misma variable se garantiza la concurrencia con la clase concurrentHashMap que implementa el lenguaje Java, la cual es segura a hilos (thread-safe) y permite un acceso ilimitado de lecturas de hasta 16 operaciones de escritura a la vez (27). De esta manera se puede plantear que el módulo Configuración entidad es una extensión necesaria del módulo Configuración general para obtener una configuración correcta en todo el sistema.

3.3 Modelo de diseño

La fase de diseño expande y detalla los modelos de análisis tomando en cuenta todas las implicaciones y restricciones técnicas. El propósito del modelo de diseño es especificar una solución que trabaje y pueda ser fácilmente convertida en código fuente y construir una arquitectura simple y fácilmente extensible. Para esto se utilizan patrones de diseño que son descripciones de clases y objetos comunicándose entre sí adaptadas para resolver un problema de diseño general en un contexto particular. Estos expresan esquemas para definir estructuras de diseño con las que construir sistemas de software. Contribuyen a reutilizar diseño gráfico, identificando aspectos claves de la estructura de un diseño que puede ser aplicado en una gran

cantidad de situaciones. Mejoran la flexibilidad, modularidad y extensibilidad, factores internos e íntimamente relacionados con la calidad del sistema percibida por el usuario. (24)

Para la realización del diseño del módulo se tienen en cuenta una serie de patrones entre los que se encuentran los Patrones Generales de Software para Asignación de Responsabilidades (GRASP por sus siglas en inglés), los cuales representan los principios básicos de la asignación de responsabilidades a objetos. (30)

El uso de los patrones Experto y Creador para asignar tareas a las clases según la información que posean, además de crear las instancias de otras en correspondencia con la responsabilidad dada. Se evidencian en la clase EntidadSistemaCrearControlador, la cual se encarga de crear instancias de Entidad_configuracion y de la clase Referencia_configuracion que es el experto de información para gestionar las referencias hacia otras instituciones hospitalarias y servicios que estas contengan.

Otros de los patrones que se utilizan son la Alta cohesión y el Bajo acoplamiento los cuales permiten que la información contenida en las clases sea coherente y esté relacionada con las mismas. Continuando el escenario anterior, estos principios se reflejan cuando la clase Referencia_configuración crea instancias de Servicio_configuracion para que los servicios se puedan asociar durante la gestión de las referencias en una institución hospitalaria. De esta manera, se minimiza la dependencia de la clase EntidadSistemaCrearControlador y se aporta un diseño más independiente que reduce el impacto de posibles cambios, así como clases más reutilizables. Durante el diseño se realizan el diagrama de paquetes para la organización de las clases y los diagramas de clases de diseño.

3.3.1 Diagrama de paquetes

Se usan para reflejar la organización de paquetes y sus elementos. Permiten dividir un modelo para agrupar y encapsular sus elementos en unidades lógicas individuales. Se pueden utilizar además para plantear la arquitectura del sistema de manera general (29). Comúnmente se usan para organizar diagramas de casos de uso y diagramas de clase.

A continuación se muestra el diagrama de paquetes del módulo Configuración general el cual basado en la arquitectura definida queda estructurado de la siguiente forma:

Todas las clases están agrupadas en el paquete Repositorio de clases. El paquete Sesiones contiene las clases controladoras Autogeneradas, Personalizadas y Propias del proceso. El paquete Entidades contiene a su vez las entidades Autogeneradas y Personalizadas. Estos

paquetes se relacionan entre sí, las vistas actualizan y consultan las entidades e invocan las controladoras y estas modifican a las entidades.

El paquete Configuración general contiene los paquetes de Realización de Casos de Uso (RCU) dónde se agrupan los casos de uso de acuerdo a las entidades que gestionan. Estos casos de uso son implementados haciendo uso de la estructura del paquete Repositorio de clases.

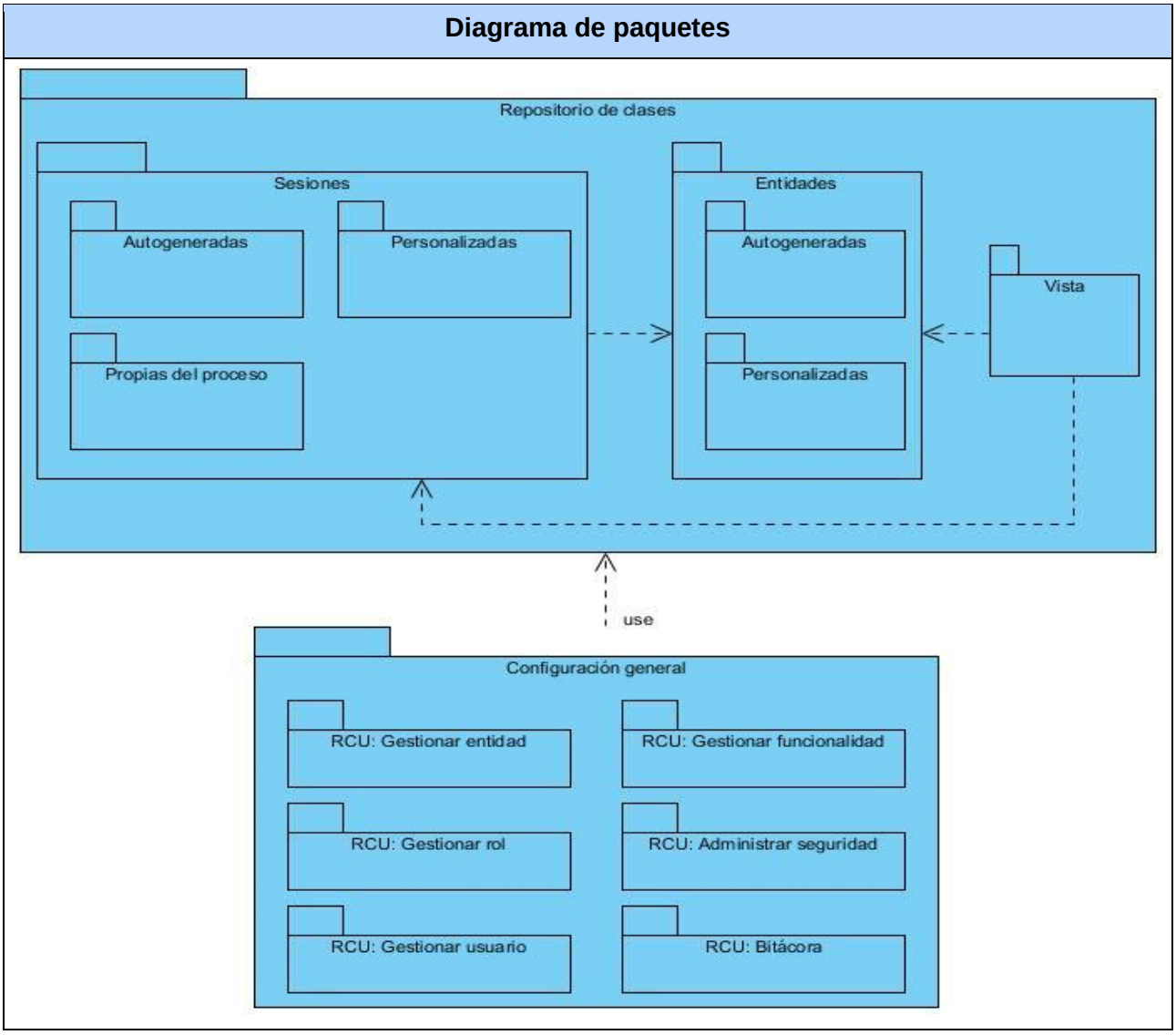

Figura 6: Diagrama de paquetes

3.3.2 Diagramas de clases de diseño

El Diagrama de Clase de Diseño (DCD) es el diagrama principal de diseño para un sistema. Este muestra las especificaciones para las clases de una aplicación y permite visualizar las

relaciones entre las clases que involucran el sistema, las cuales pueden ser asociativas, de herencia y de uso. (23)

Los diagramas de clases de diseño del módulo Configuración general siguiendo la estructura explicada en el epígrafe anterior quedan de la siguiente forma: Están compuestos por páginas clientes que son construidas por páginas servidoras y que a su vez contienen formularios que muestran y capturan toda la información enviándolas a las páginas servidoras. Las páginas servidoras invocan métodos o responsabilidades en la clase controladora que según la acción solicitada puede consultar o modificar las entidades.

A continuación se visualiza una muestra de los diagramas de clases de diseño del módulo, para una mejor comprensión de las mismas ver anexo “Descripción de las clases de diseño del módulo configuración general”.

DCD_Gestionar entidad

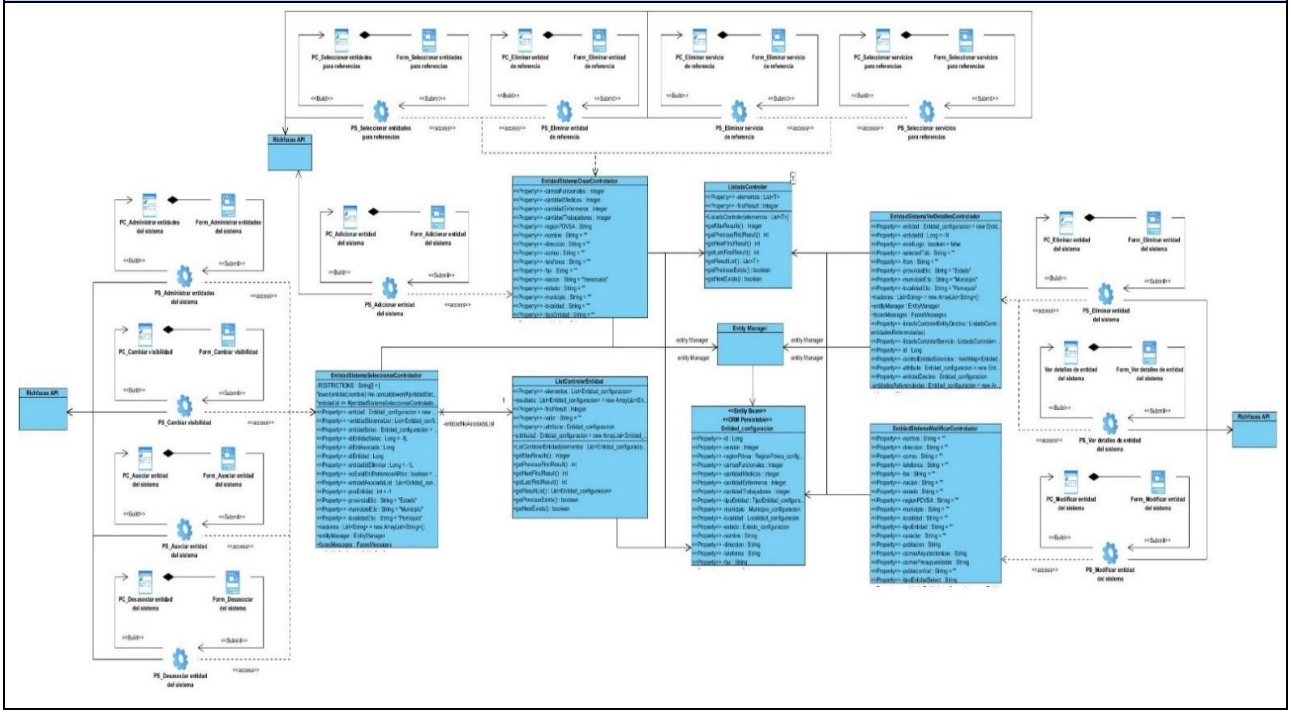

Figura 7: Diagrama de clases de diseño – Gestionar entidad

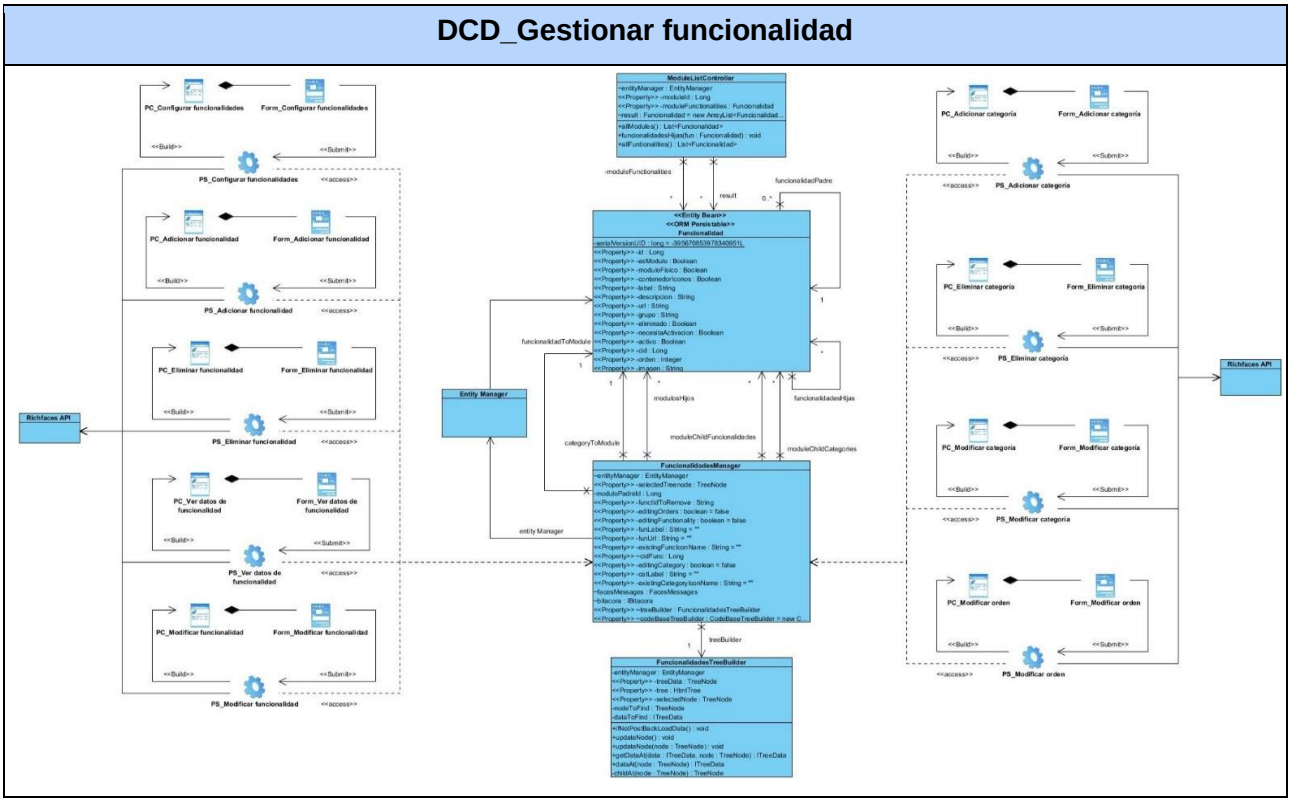

Figura 8: Diagrama de clases de diseño – Gestionar funcionalidad

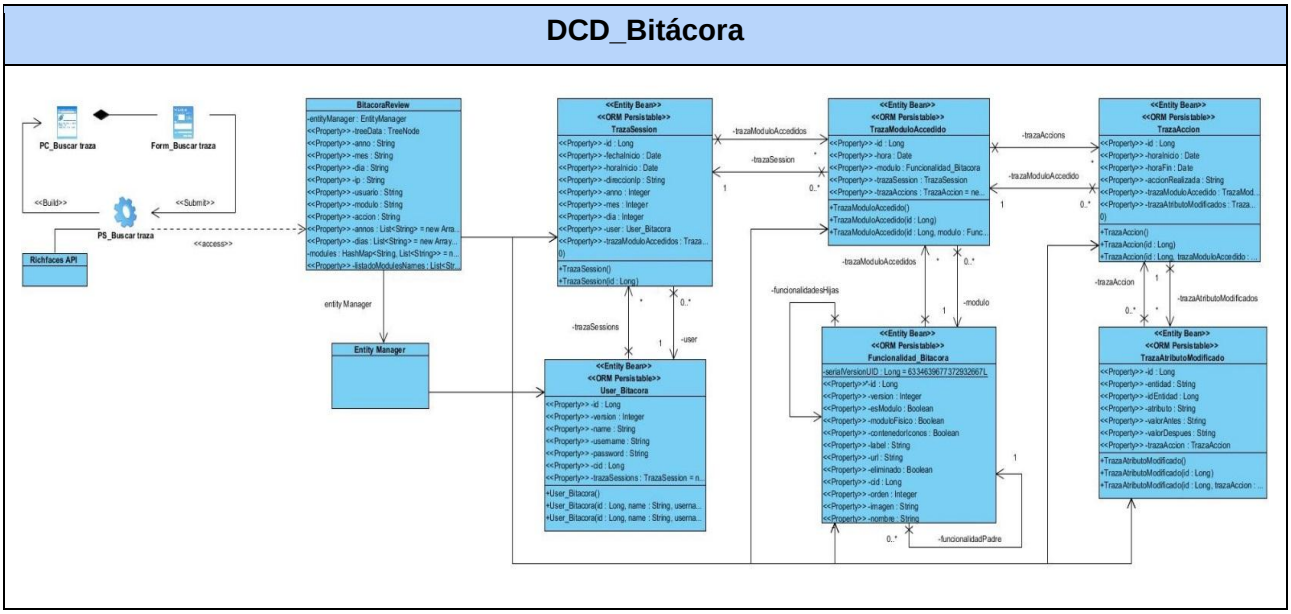

Figura 9: Diagrama de clases de diseño - Bitácora

3.4 Conclusiones del capítulo

- El diseño propuesto, hace uso del patrón Modelo Vista Controlador, permitiendo una mejor organización y comprensión de la estructura de la propuesta de solución.
- El empleo de los patrones de diseño GRASP permite una solución con objetos que se valgan de la información requerida y definiciones de clases sencillas más fáciles de comprender y de mantener con mejores oportunidades de reutilización.
- Los diagramas de clases del diseño facilitan la comprensión del funcionamiento del módulo Configuración general del HIS del CESIM.

CAPÍTULO 4. Implementación del módulo Configuración general

En este capítulo se exponen las principales características de la implementación del módulo Configuración general. Se detalla el Modelo de datos donde se puede apreciar la estructura de la información almacenada en el módulo. También se muestra el Modelo de implementación con sus respectivos diagramas, los cuales describen las clases y subsistemas en términos de componentes así como su organización, además de las dependencias entre los nodos físicos en los que funcionará el módulo. También se presenta la propuesta para lograr una gestión eficiente de la seguridad así como el tratamiento de errores en el módulo.

4.1 Modelo de datos

Es un conjunto de conceptos que permiten describir los datos de una aplicación o sistema de información. Este representa la estructura física de las tablas de la base de datos y es esencial para el desarrollo de una aplicación; pues a través de él se puede conseguir la compatibilidad necesaria para manejar grandes cantidades de datos. El diagrama que se muestra a continuación es un modelo entidad-relación el cual está compuesto por entidades y sus relaciones. Las entidades son objetos de los cuales el sistema guarda información, compuestas por atributos que no son más que características o propiedades de estas. Las relaciones por su parte muestran la asociación entre las entidades (25). A continuación se muestra el modelo de datos del módulo Configuración general, en este se representan las mismas tablas del módulo Configuración con las modificaciones necesarias para dar solución al problema planteado.

Modelo de datos

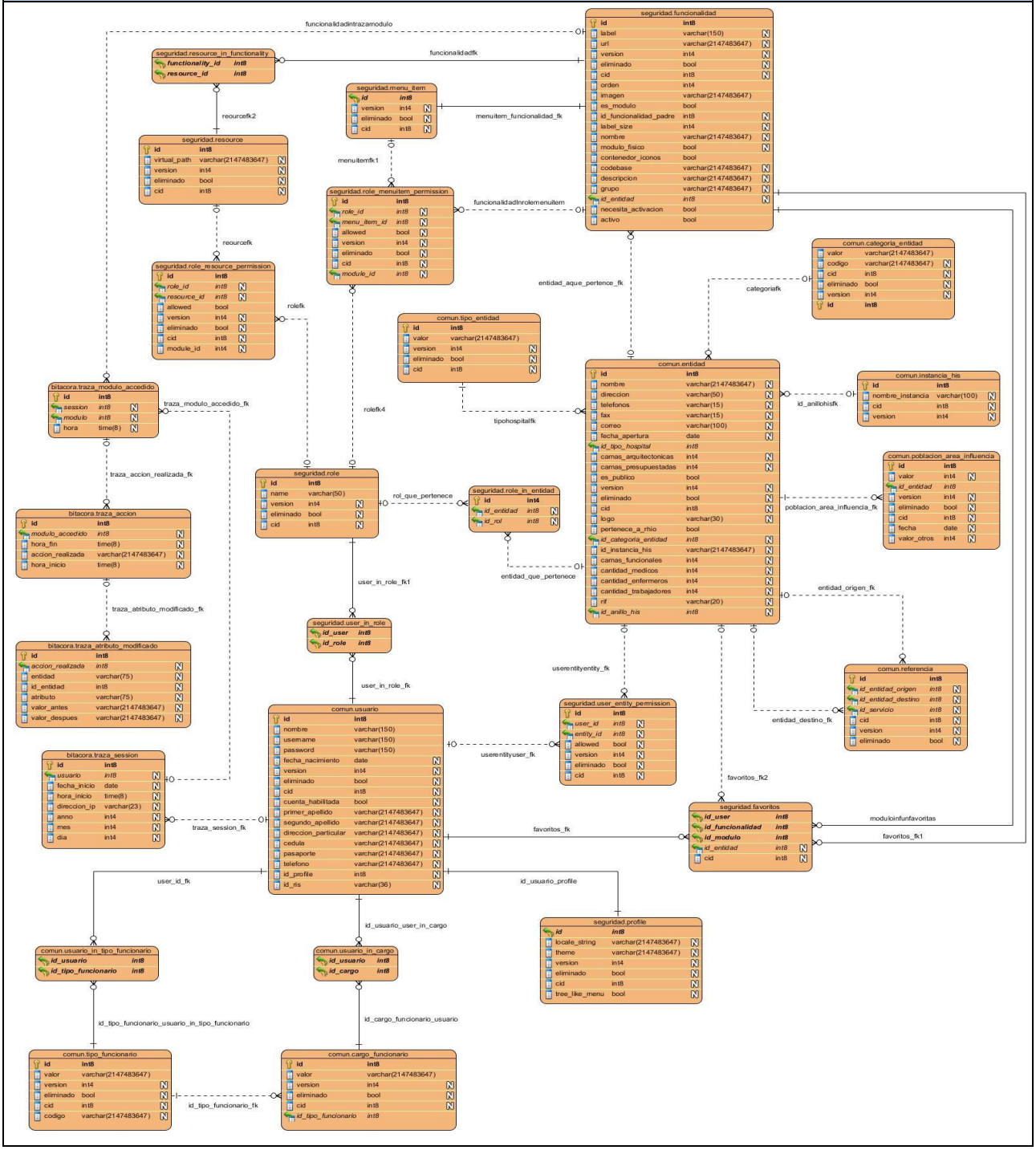

Figura 10: Modelo de datos

4.1.1 Descripción de las tablas de la base de datos

Tabla 5: Descripción de los atributos comunes en las tablas.

Atributos comunes en las tablas		
Atributos	**Tipo**	**Descripción**
id	bigint	Llave primaria de las tablas de la base de datos.
cid	bigint	Identificador asociado a la acción que se realiza.

Tabla 6: Descripción de la tabla entidad.

entidad		
Tabla que contiene toda la información de las entidades del sistema.		
Atributos	**Tipo**	**Descripción**
nombre	varchar	Nombre de la institución hospitalaria.
direccion	varchar	Dirección de la institución hospitalaria.
telefonos	varchar	Teléfonos de la institución hospitalaria.
fax	varchar	Fax de la institución hospitalaria.
correo	varchar	Correo de la institución hospitalaria.
fecha_apertura	date	Fecha de apertura de la institución hospitalaria.
id_tipo_hospital	bigint	Tipo de institución hospitalaria.
camas_arquitectonicas	integer	Cantidad de camas
camas_presupuestadas	integer	Cantidad de camas
es_publico	bool	Si es público o no.
version	integer	Indica con qué versión de la entidad se está trabajando. Es usado para garantizar que se está trabajando con la versión de la entidad más actualizada que existe en la base de datos.

eliminado	bool	Permite la eliminación lógica con que cuenta el sistema, cuando está en verdadero indica que la entidad está eliminada.
logo	varchar	Logo de la institución hospitalaria.
pertenece_a_rhio	bool	Si pertenece al anillo o no.
id_categoria_entidad	bigint	Identificador de la categoría de la entidad.
id_instancia_his	varchar	Identificador de la instancia del anillo.
camas_funcionales	integer	Cantidad de camas en funcionamiento.
cantidad_medicos	integer	Cantidad de médicos de la institución hospitalaria.
cantidad_enfermeros	integer	Cantidad de enfermeros de la institución hospitalaria.
cantidad_trabajadores	integer	Cantidad de trabajadores de la institución hospitalaria.
rif	varchar	Registro de Identificación Fiscal en Venezuela.
id_anillo_his	bigint	Identificador del anillo donde se encuentra la institución.

Tabla 7: Descripción de la tabla funcionalidad.

funcionalidad		
Tabla que contiene todos los datos de las funcionalidades del sistema.		
Atributos	**Tipo**	**Descripción**
label	varchar	Nombre de la funcionalidad.
url	varchar	Dirección de la página que es invocada por la funcionalidad.
version	integer	Indica con qué versión de la entidad se está trabajando. Es usado para garantizar que se está trabajando con la versión de la entidad más actualizada que existe en la base de datos.
eliminado	bool	Permite la eliminación lógica con que cuenta el sistema, cuando está en verdadero indica que la entidad está

		eliminada.
orden	integer	Orden en el que se encuentra la funcionalidad.
imagen	varchar	Imagen del icono de la funcionalidad.
es_modulo	bool	Si es módulo o no.
id_funcionalidad_padre	bigint	Identificador de la funcionalidad padre.
label_size	integer	Cantidad de caracteres que puede tener el nombre.
nombre	varchar	Nombre de la funcionalidad.
modulo_fisico	bool	Módulo contenedor de las funcionalidades.
contenedor_iconos	bool	Si contiene iconos o no.
codebase	varchar	Dirección física de la funcionalidad.
descripcion	varchar	Descripción del módulo.
grupo	varchar	Grupo al que pertenece.
id_entidad	bigint	Identificador de la entidad.
necesita_activacion	bool	Si necesita ser activado o no.
activo	bool	Si está activo o no.

Tabla 8: Descripción de la tabla traza_session.

traza_session		
Tabla que contiene la información de las sesiones abiertas en el sistema.		
Atributos	**Tipo**	**Descripción**
usuario	bigint	Usuario que inicia la sesión en el sistema.
fecha_inicio	date	Fecha en que se inicia la sesión.
hora_inicio	time	Hora en que se inicia la sesión.
direccion_ip	varchar	Dirección de IP de la máquina donde se inicia la sesión.
anno	integer	Año en que se inicia la sesión.
mes	integer	Mes en que se inicia la sesión.
dia	integer	Día en que se inicia la sesión.

Tabla 9: Descripción de la tabla traza_modulo_accedido.

traza_modulo_accedido		
Tabla que contiene toda la información de los módulos accedidos por los usuarios.		
Atributos	**Tipo**	**Descripción**
session	bigint	Identificador de la sesión abierta.
modulo	bigint	Módulo al que se accede.
hora	time	Hora a la que se accede al módulo.

Tabla 10: Descripción de la tabla traza_accion.

traza_accion		
Tabla que contiene toda la información de las acciones realizadas en el sistema.		
Atributos	**Tipo**	**Descripción**
modulo_accedido	bigint	Identificador del módulo accedido.
hora_fin	time	Hora en la que se finaliza la acción.
accion_realizada	varchar	Acción que se realiza en el sistema.
hora_inicio	time	Hora en la que se inicia la acción.

Tabla 11: Descripción de la tabla traza_atributo_modificado.

traza_atributo_modificado		
Tabla que contiene toda la información de los atributos modificados en el sistema.		
Atributos	**Tipo**	**Descripción**
acción_realizada	bigint	Identificador de la acción realizada.
entidad	varchar	Entidad en la que se modifica el atributo.
id_entidad	bigint	Identificador de la entidad en la que se modifica el atributo.
atributo	varchar	Atributo que es modificado.
valor_antes	varchar	Valor del atributo antes de modificarlo.
valos_despues	varchar	Valor del atributo después de modificarlo.

4.2 Modelo de implementación

El modelo de implementación representa a través de descripciones, cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue.

4.2.1 Diagrama de componentes

Un diagrama de componentes representa la organización y las dependencias entre los componentes de un software. Estos representan los elementos de software que entran en la

fabricación de aplicaciones informáticas, permitiendo conocer a los desarrolladores y clientes la estructura física que tiene el sistema y cómo se relacionan sus partes. Los componentes pueden ser librerías, archivos, documentos o ejecutables (29). A continuación se muestra el diagrama de los componentes del módulo Configuración general:

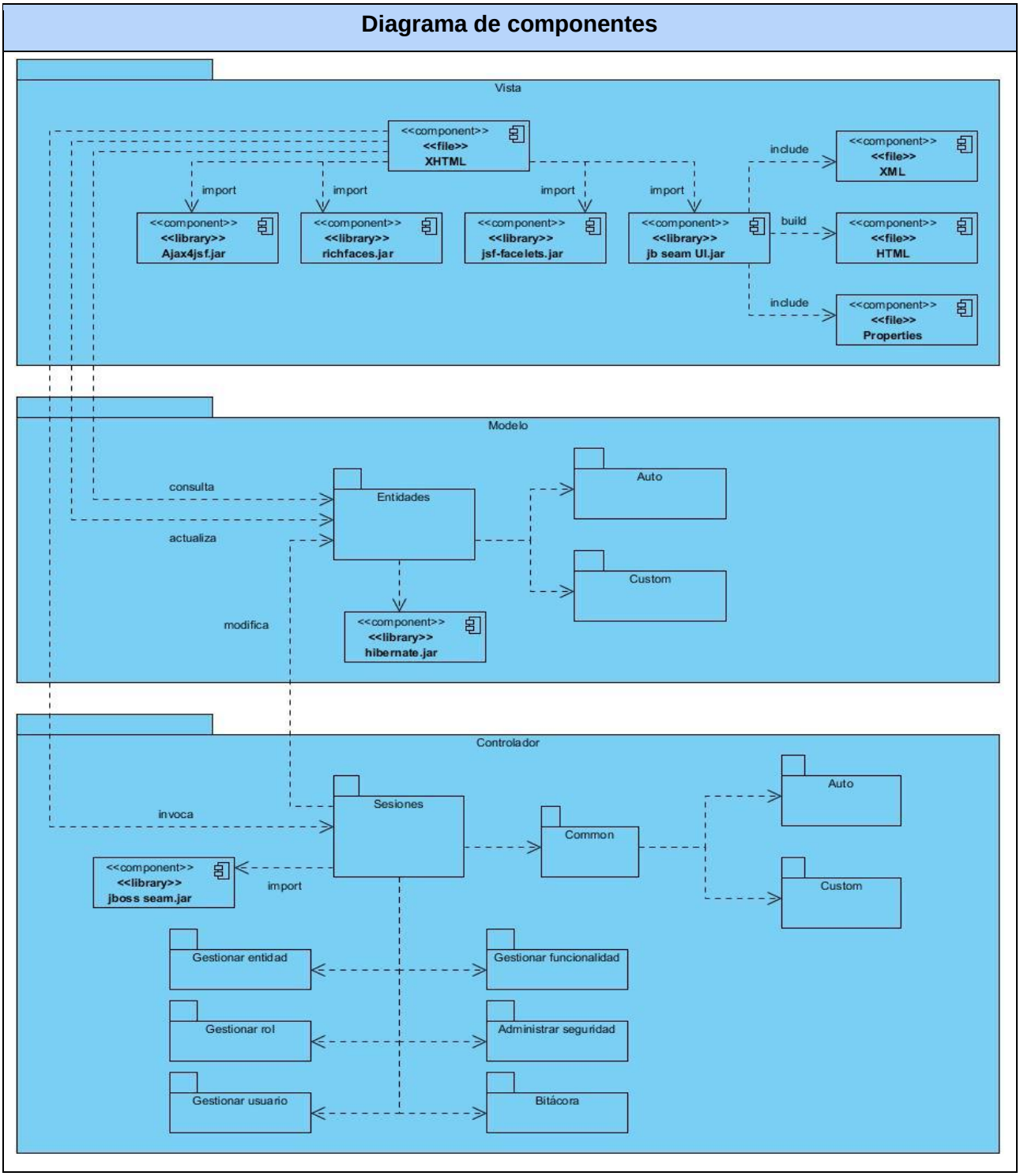

Figura 11: Diagrama de componentes

4.2.2 Diagrama de despliegue

En el diagrama de despliegue se define la estructuración, descripción y relaciones físicas de los distintos nodos que componen un sistema. Con los nodos se modela la topología del hardware sobre el que se ejecuta el módulo. Estos pueden representar un elemento de hardware o de software y son conectados por asociaciones de comunicación tales como enlaces de red y conexiones por el Protocolo de Control de Transmisión/Protocolo de Internet (TCP / IP por sus siglas en inglés). (29)

Para la implantación y utilización del sistema el usuario debe conectarse a este mediante una computadora cliente, haciendo uso de un navegador web. Las peticiones por el Protocolo Seguro de Transferencia de Hipertexto (HTTPS por sus siglas en inglés) son procesadas por el servidor de aplicaciones, y este a su vez envía las respuestas al cliente. El servidor de aplicaciones emite peticiones por el protocolo TCP/IP hacia el servidor de base de datos. El diagrama de despliegue del sistema queda definido de la siguiente forma:

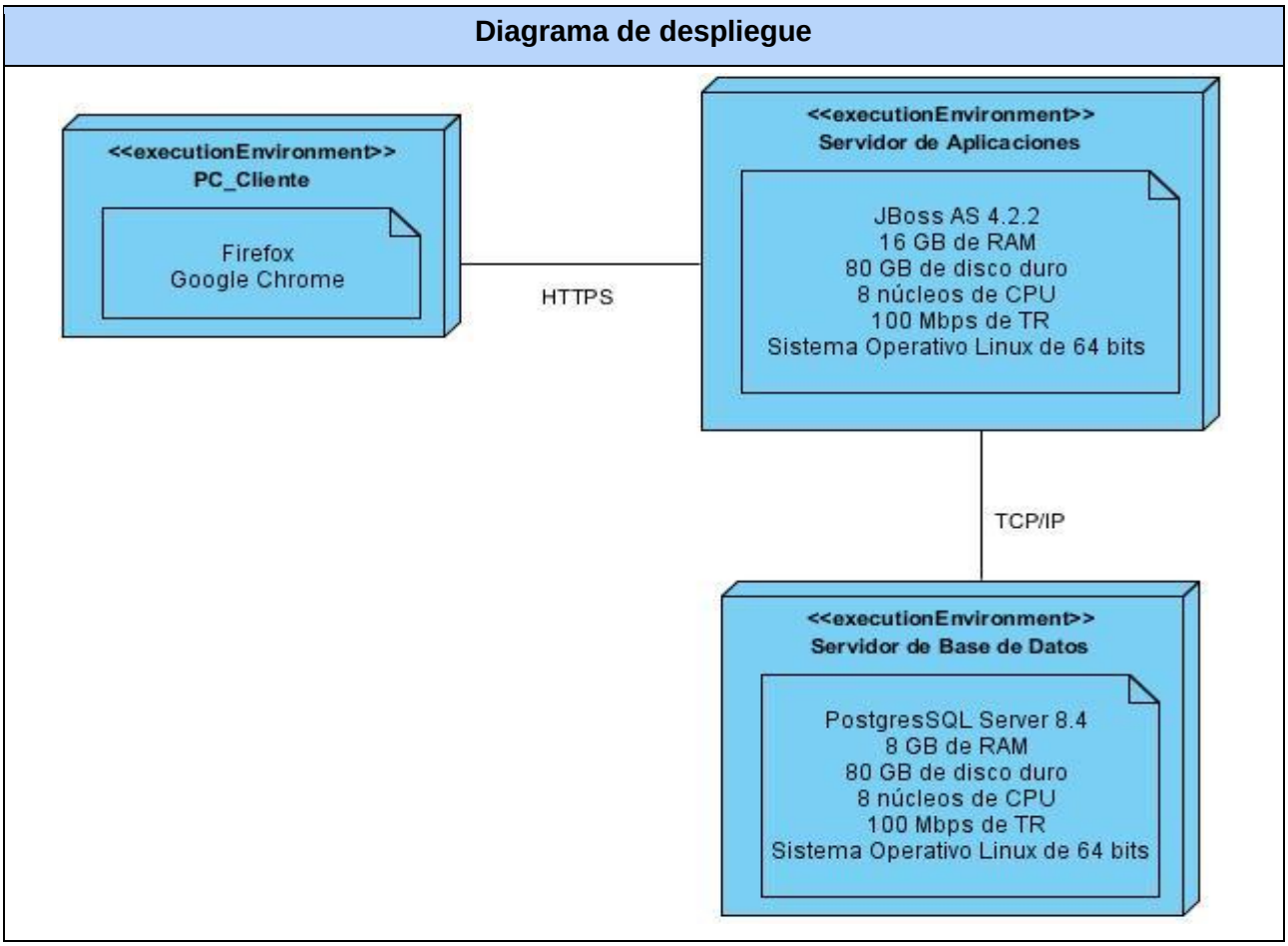

Figura 12: Diagrama de despliegue

4.3 Tratamiento de errores

El tratamiento de errores evita las interrupciones indeseadas y bloqueos de los programas, facilitando el localizar rápidamente la fuente de un determinado error, así como el darle un tratamiento adecuado manejándolo, acotándolo, haciendo un seguimiento o traza, o incluso ignorándolo. (30)

En el módulo Configuración general los errores de introducción de datos por parte del usuario son resueltos a través de mensajes e indicaciones apropiadas, que señalan la existencia de datos incorrectos o incompletos cumpliendo con las pautas definidas. Otros posibles errores que se puedan suscitar son manejados con el tratamiento de excepciones correspondientes y se implementa un disparador para garantizar el registro de las acciones realizadas en el sistema.

4.4 Seguridad

El sistema cuenta con dos tipos de administradores: los que gestionan las configuraciones generales y los encargados de la administración de una institución hospitalaria. Los privilegios para realizar estas actividades se conceden desde el módulo Configuración general donde el acceso es garantizado exclusivamente para los administradores generales. Estos gestionan también las reglas de seguridad y los roles generales, asegurando que cada usuario asociado a los roles creados tengan acceso a la información correspondiente en el sistema. Todas las operaciones realizadas por los usuarios son almacenadas en la Bitácora de sucesos posibilitando que los administradores puedan visualizar las trazas de estas acciones y sus modificaciones a la hora de realizar auditorías al sistema.

De igual forma, la información ingresada al sistema no será eliminada físicamente de la base de datos permitiendo al mismo recuperarse de posibles errores que puedan ocurrir durante su uso y recuperar la información de la base de datos a partir de los respaldos o salvas realizadas en el sistema. Los datos introducidos son validados en todo momento y se establecen mecanismos de control y verificación para los procesos.

4.5 Conclusiones del capítulo

- Los artefactos obtenidos garantizan la completa descripción de los datos, de los componentes y las dependencias entre estos, en el módulo desarrollado.
- Los métodos implementados para garantizar el tratamiento de errores y la seguridad en el sistema permiten la correcta administración y gestión de la información en la configuración general.

CONCLUSIONES

Con el desarrollo del módulo Configuración general del Sistema de Información Hospitalaria del Centro de Informática Médica se concluye que:

- El análisis de los sistemas estudiados determinan los procesos que se gestionan actualmente en la configuración de los HIS.
- La separación de los procesos que gestiona el módulo Configuración brinda mayor usabilidad y facilidad de mantenimiento en el sistema.
- El uso del patrón Modelo Vista Controlador propicia mayor organización y comprensión de la estructura del módulo Configuración general.
- La utilización de pautas en el proceso de desarrollo, garantiza la uniformidad y homogeneidad en los artefactos obtenidos.
- La implementación del módulo Configuración general permite la correcta gestión, administración y centralización de las configuraciones generales del sistema.

RECOMENDACIONES

Para un mejor funcionamiento del módulo Configuración general se recomienda.

- Definir un mecanismo que permita al sistema almacenar en la bitácora todas las acciones realizadas en el mismo, de manera que le reste al desarrollador la responsabilidad de implementar esta tarea en cada funcionalidad.

REFERENCIAS BIBLIOGRÁFICAS

1. **Guerrero Martínez, Juan F.** Universitat de Valencia, Open Course Ware. [En línea] 2011. [Citado el: 20 de Noviembre de 2014.] http://ocw.uv.es/ingenieria-y-arquitectura/1-5/ib_material/IB_T11_OCW.pdf.
2. **SiHO.** SiHO. [En línea] 2010. [Citado el: 17 de Enero de 2015.] http://www.sihosys.com/Sistema_Integral_Hospitalario_%28SiHO%29.pdf.
3. **DIGITECH.** Digitech. [En línea] 2012. [Citado el: 15 de Enero de 2015.] http://www.digitech.net.ni/sistema-integral.
4. **León Hernández, Pedro Luis y Santos Suárez, Adelys.** Portales Medicos. [En línea] 26 de Noviembre de 2012. [Citado el: 19 de Enero de 2015.] http://www.portalesmedicos.com/publicaciones/articles/4894/3/Phphclinica.-Sistema-de-informacion-hospitalaria.
5. **W3C.** W3C. [En línea] W3C, 1 de Agosto de 2002. [Citado el: 25 de Enero de 2015.] http://www.w3.org/TR/xhtml1/.
6. **Universidad de Alicante.** Universidad de Alicante. [En línea] Depto. Ciencia de la Computación e IA, 26 de Junio de 2014. [Citado el: 4 de Febrero de 2015.] http://www.jtech.ua.es/j2ee/publico/jsf-2012-13/sesion01-apuntes.html.
7. **Oracle.** Oracle Help Center. [En línea] 2013. [Citado el: 8 de Febrero de 2015.] https://docs.oracle.com/javaee/6/tutorial/doc/giepx.html.
8. **Red Hat.** JBoss Developer. [En línea] 2003. [Citado el: 10 de Febrero de 2015.] https://docs.jboss.org/richfaces/latest_3_3_X/en/tlddoc/a4j/tld-summary.html.
9. **Leathem, Brian, Fryc, Lukas y Rogers, Sean.** Jboss, Community Documentation. [En línea] 22 de Febrero de 2011. [Citado el: 6 de Febrero de 2015.] http://docs.jboss.org/richfaces/latest_4_2_X/Developer_Guide/en-US/html/chap-Developer_Guide-Introduction.html.
10. **Meroño Sánchez, Juan José.** Scribd. [En línea] 2009. [Citado el: 17 de Febrero de 2015.] https://es.scribd.com/doc/267237812/57/Introduccion.
11. **King, Gavin, y otros.** Javalobby. [En línea] 18 de Septiembre de 2005. [Citado el: 10 de Febrero de 2015.] www.javalobby.org/java/forums/t45690.html.

12. **Red Hat.** Hibernate, Community Documentation. [En línea] Red Hat, 25 de Abril de 2013. [Citado el: 18 de Febrero de 2015.] http://docs.jboss.org/hibernate/core/4.2/manual/en-US/html/pr01.html.

13. **Jendrock, Eric, y otros.** Oracle Help Center. [En línea] Oracle, Septiembre de 2014. [Citado el: 28 de Enero de 2015.] https://docs.oracle.com/javaee/7/JEETT.pdf.

14. **Oracle.** Oracle Help Center. [En línea] [Citado el: 27 de Febrero de 2015.] http://www.oracle.com/technetwork/java/index.html.

15. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo.* Madrid : Pearson Educación S.A, 2000.

16. **Object Management Group.** OMG. [En línea] Julio de 2005. [Citado el: 7 de Febrero de 2015.] http://www.uml.org/.

17. **Rivas, Lornel A., y otros.** Lisi, Laboratorio de investigación en Sistemas de Información. [En línea] 30 de Septiembre de 2014. [Citado el: 13 de Febrero de 2015.] http://www.lisi.usb.ve/publicaciones/05%20herramientas/herramientas_25.pdf.

18. **Rydahl Andersen, Max.** JBoss Developer. [En línea] Red Hat. [Citado el: 15 de Febrero de 2015.] https://www.jboss.org/products/devstudio/overview/.

19. **Martínez, Rafael.** PostgreSQL. [En línea] 2 de Octubre de 2010. [Citado el: 16 de Febrero de 2015.] http://www.postgresql.org.es/sobre_postgresql.

20. **PgAdmin.** PgAdmin PostreSQL Tools. [En línea] [Citado el: 20 de Febrero de 2015.] http://www.pgadmin.org/index.php.

21. **Visual Paradigm.** Visual Paradigm. [En línea] Visual Paradigm International Limited, 16 de Agosto de 2010. [Citado el: 19 de Febrero de 2015.] http://www.visual-paradigm.com/aboutus/newsreleases/vpuml80.jsp.

22. **Soldano, Alessio, y otros.** Jboss Community Documentation. [En línea] Enero de 2008. [Citado el: 16 de Febrero de 2015.] https://docs.jboss.org/jbossas/docs/Server_Configuration_Guide/4/html/.

23. **R. Schach, Stephen.** *Análisis y diseño orientado a objetos con UML y el Proceso Unificado.* México : Mc Graw Hill, 2005.

24. **Pressman, Roger S.** *Ingeniería del Software: Un Enfoque Práctico.* Madrid : Mc Graw Hill, 2005.

25. **Sommerville, Ian.** *Ingeniería del Software.* Madrid : Pearson Addison Wesley, 2005.

26. **Bascón Pantoja, Ernesto.** Acta Nova, Revista de Ciencias y Tecnología. [En línea] Diciembre de 2004. [Citado el: 15 de Marzo de 2015.] http://ucbconocimiento.ucbcba.edu.bo/index.php/ran/article/view/84/81.

27. **Oracle.** Oracle Help Center. [En línea] [Citado el: 20 de Mayo de 2015.] https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ConcurrentHashMap.html.

28. **Polo Usaola, Macario.** ESI Escuela Superior de Informática. [En línea] [Citado el: 21 de Marzo de 2015.] www.inf-cr.uclm.es/www/mpolo/asig/0304/0102/patronesgrasp.ppt.

29. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* Madrid : Addison Wesley, 2007.

30. **Sierra, Manuel.** Aprender a programar. [En línea] [Citado el: 25 de Mayo de 2015.] http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=678:gestion-de-excepciones-en-java-captura-con-bloques-try-catch-finally-ejemplos-resueltos-sencillos-cu00927c&catid=58:curso-lenguaje-programacion-java-nivel-avanzado-i&Itemid.

BIBLIOGRAFÍA

- **Alfatec Sistemas**. Alfatec Sistemas. [En línea] Alfatec Sistemas, 2012. [Citado el: 18 de Noviembre de 2014.] http://alfatecsistemas.es/soluciones/sistema-informacion-hospitalaria/.
- **Álvarez, Miguel Angel**. Desarrolloweb. [En línea] 2 de Enero de 2014. [Citado el: 16 de Marzo de 2015.] http://www.desarrolloweb.com/articulos/que-es-mvc.html.
- **Bascón Pantoja, Ernesto**. Acta Nova, Revista de Ciencias y Tecnología. [En línea] Diciembre de 2004. [Citado el: 15 de Marzo de 2015.] http://ucbconocimiento.ucbcba.edu.bo/index.php/ran/article/view/84/81.
- **DIGITECH**. Digitech. [En línea] 2012. [Citado el: 15 de Enero de 2015.] http://www.digitech.net.ni/sistema-integral.
- **Figueroa, Roberto G., Solis, Camilo J. y Cabrera, Armando A**. EVA. [En línea] [Citado el: 11 de Febrero de 2015.] http://eva.uci.cu/file.php/161/Documentos/Materiales_complementarios/UD_1_Procesos/Metodologias/METODOLOGIAS_TRADICIONALES_VS._METODOLOGIAS_AGILES.pdf.
- **Giusto Bilic, Denice**. Welivesecurity. [En línea] 9 de Julio de 2014. [Citado el: 5 de Diciembre de 2014.] http://www.welivesecurity.com/la-es/2014/07/09/factores-considerar-fortalecer-sistemas-informacion/.
- **Guerrero Martínez, Juan F.** Universitat de Valencia, Open Course Ware. [En línea] 2011. [Citado el: 20 de Noviembre de 2014.] http://ocw.uv.es/ingenieria-y-arquitectura/1-5/ib_material/IB_T11_OCW.pdf.
- **Gutiérrez, Demián**. Code Compiling. [En línea] Abril de 2011. [Citado el: 1 de Marzo de 2015.] http://www.codecompiling.net/files/slides/UML_clase_02_UML_casos_de_uso.pdf.
- **Jacobson, Ivar, Booch, Grady y Rumbaugh, James**. El Proceso Unificado de Desarrollo. Madrid : Pearson Educación S.A, 2000.
- **Jendrock, Eric, y otros**. Oracle Help Center. [En línea] Oracle, Septiembre de 2014. [Citado el: 28 de Enero de 2015.] https://docs.oracle.com/javaee/7/JEETT.pdf.
- **Junta de Andalucía**. Marco de Desarrollo de la Junta de Andalucía. [En línea] Junta de Andalucía. [Citado el: 19 de Febrero de 2015.] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/144.

- **Junta de Andalucía**. Marco de Desarrollo de la Junta de Andalucía. [En línea] Junta de Andalucía. [Citado el: 20 de Febrero de 2015.] http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/407.
- **King, Gavin, y otros**. Javalobby. [En línea] 18 de Septiembre de 2005. [Citado el: 10 de Febrero de 2015.] www.javalobby.org/java/forums/t45690.html.
- **Leathem, Brian, Fryc, Lukas y Rogers, Sean**. Jboss, Community Documentation. [En línea] 22 de Febrero de 2011. [Citado el: 6 de Febrero de 2015.] http://docs.jboss.org/richfaces/latest_4_2_X/Developer_Guide/en-US/html/chap-Developer_Guide-Introduction.html.
- **León Hernández, Pedro Luis**. Portales Medicos. [En línea] 26 de Noviembre de 2012. [Citado el: 19 de Enero de 2015.] http://www.portalesmedicos.com/publicaciones/articles/4894/3/Phphclinica.-Sistema-de-informacion-hospitalaria.
- **Martínez, Rafael**. PostgreSQL. [En línea] 2 de Octubre de 2010. [Citado el: 16 de Febrero de 2015.] http://www.postgresql.org.es/sobre_postgresql.
- **Mifsud, Elvira**. Observatorio Tecnológico. [En línea] 26 de Marzo de 2012. [Citado el: 24 de Mayo de 2015.] http://recursostic.educacion.es/observatorio/web/en/software/software-general/1040-introduccion-a-la-seguridad-informatica?showall=1.
- **Object Management Group**. OMG. [En línea] Julio de 2005. [Citado el: 7 de Febrero de 2015.] http://www.uml.org/.
- **Oracle**. Oracle Help Center. [En línea] [Citado el: 20 de Mayo de 2015.] https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ConcurrentHashMap.html.
- **Oracle**. Oracle Help Center. [En línea] [Citado el: 27 de Febrero de 2015.] http://www.oracle.com/technetwork/java/index.html.
- **Oracle**. Oracle Help Center. [En línea] 2013. [Citado el: 8 de Febrero de 2015.] https://docs.oracle.com/javaee/6/tutorial/doc/giepx.html.
- **PgAdmin**. PgAdmin PostreSQL Tools. [En línea] [Citado el: 20 de Febrero de 2015.] http://www.pgadmin.org/index.php.
- **Polo Usaola, Macario**. ESI Escuela Superior de Informática. [En línea] [Citado el: 21 de Marzo de 2015.] www.inf-cr.uclm.es/www/mpolo/asig/0304/0102/patronesgrasp.ppt.

- **Potencier, Fabien**. Librosweb. [En línea] [Citado el: 15 de Marzo de 2015.] http://librosweb.es/libro/jobeet_1_4/capitulo_4/la_arquitectura_mvc.html.
- **Pressman, Roger S**. Ingeniería del Software: Un Enfoque Práctico. Madrid : Mc Graw Hill, 2005.
- **R. Schach, Stephen**. Análisis y diseño orientado a objetos con UML y el Proceso Unificado. México : Mc Graw Hill, 2005.
- **Red Hat**. EJB3. [En línea] [Citado el: 17 de Febrero de 2015.] http://ejb3.jboss.org/overview.html.
- **Red Hat**. Hibernate, Community Documentation. [En línea] Red Hat, 25 de Abril de 2013. [Citado el: 18 de Febrero de 2015.] http://docs.jboss.org/hibernate/core/4.2/manual/en-US/html/pr01.html.
- **Red Hat**. JBoss Developer. [En línea] 2003. [Citado el: 10 de Febrero de 2015.] https://docs.jboss.org/richfaces/latest_3_3_X/en/tlddoc/a4j/tld-summary.html.
- **Rivas, Lornel A., y otros**. Lisi, Laboratorio de investigación en Sistemas de Información. [En línea] 30 de Septiembre de 2014. [Citado el: 13 de Febrero de 2015.] http://www.lisi.usb.ve/publicaciones/05%20herramientas/herramientas_25.pdf.
- **Rumbaugh, James, Jacobson, Ivar y Booch, Grady**. El Lenguaje Unificado de Modelado. Manual de Referencia. Madrid : Addison Wesley, 2007.
- **Rydahl Andersen, Max**. JBoss Developer. [En línea] Red Hat. [Citado el: 15 de Febrero de 2015.] https://www.jboss.org/products/devstudio/overview/.
- **Sierra, Manuel**. Aprender a programar. [En línea] [Citado el: 25 de Mayo de 2015.] http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=678:gestion-de-excepciones-en-java-captura-con-bloques-try-catch-finally-ejemplos-resueltos-sencillos-cu00927c&catid=58:curso-lenguaje-programacion-java-nivel-avanzado-i&Itemid.
- **SiHO**. SiHO. [En línea] 2010. [Citado el: 17 de Enero de 2015.] http://www.sihosys.com/Sistema_Integral_Hospitalario_%28SiHO%29.pdf.
- **Soldano, Alessio, y otros**. Jboss Community Documentation. [En línea] Enero de 2008. [Citado el: 16 de Febrero de 2015.] https://docs.jboss.org/jbossas/docs/Server_Configuration_Guide/4/html/.

- **Sommerville, Ian**. Ingeniería del Software. Madrid : Pearson Addison Wesley, 2005.
- **Universidad de Alicante**. Universidad de Alicante. [En línea] Depto. Ciencia de la Computación e IA, 26 de Junio de 2014. [Citado el: 4 de Febrero de 2015.] http://www.jtech.ua.es/j2ee/publico/jsf-2012-13/sesion01-apuntes.html.
- **Visual Paradigm**. Visual Paradigm. [En línea] Visual Paradigm International Limited, 16 de Agosto de 2010. [Citado el: 19 de Febrero de 2015.] http://www.visual-paradigm.com/aboutus/newsreleases/vpuml80.jsp.
- **W3C**. W3C. [En línea] W3C, 1 de Agosto de 2002. [Citado el: 25 de Enero de 2015.] http://www.w3.org/TR/xhtml1/.

ANEXOS

1. Requisitos funcionales del módulo Configuración general

En este anexo se muestran los requisitos funcionales definidos para el desarrollo del módulo Configuración general, las descripciones de los mismos se encuentran en el Expediente de proyecto “HIS_Configuración_general_Especificacion_de_requisitos_de_software”.

Tabla 12: Requisitos funcionales

Requisitos funcionales	
RF1. Administrar entidades del sistema.	RF33. Ver datos de funcionalidad.
RF2. Adicionar entidad del sistema.	RF34. Modificar funcionalidad.
RF3. Ver detalles de entidad del sistema.	RF35. Eliminar funcionalidad.
RF4. Modificar entidad del sistema.	RF36. Cambiar visibilidad de funcionalidad.
RF5. Eliminar entidad del sistema.	RF37. Administrar seguridad.
RF6. Cambiar visibilidad de la entidad.	RF38. Buscar traza.
RF7. Asociar entidad del sistema.	RF39. Configuraciones personales.
RF8. Desasociar entidad del sistema.	RF40. Configurar anillo.
RF9. Seleccionar entidades para referencias.	RF41. Ver detalles de certificado raíz.
RF10. Eliminar entidad asociada.	RF42. Modificar certificado raíz.
RF11. Seleccionar servicios para referencias.	RF43. Administrar certificados de confianza.
RF12. Eliminar servicio de referencia.	RF44. Buscar certificado de confianza.
RF13. Administrar roles.	RF45. Eliminar certificado de confianza.
RF14. Adicionar rol.	RF46. Importar certificados de confianza.
RF15. Ver detalles de rol.	RF47. Exportar certificado de confianza.
RF16. Buscar rol.	RF48. Administrar identidades digitales.
RF17. Modificar rol.	RF49. Buscar identidad digital.

RF18. Eliminar rol. RF19. Administrar usuarios. RF20. Adicionar usuario. RF21. Ver detalles de usuario. RF22. Buscar usuario. RF23. Modificar usuario. RF24. Ver datos de usuario. RF25. Eliminar usuario. RF26. Configurar funcionalidades. RF27. Adicionar categoría. RF28. Modificar categoría. RF29. Eliminar categoría. RF30. Cambiar visibilidad de categoría. RF31. Modificar orden. RF32. Adicionar funcionalidad.	RF50. Administrar servidores proxy. RF51. Adicionar servidor proxy. RF52. Ver detalles de servidor proxy. RF53. Buscar servidor proxy. RF54. Ver datos de servidor proxy. RF55. Modificar servidor proxy. RF56. Eliminar servidor proxy. RF57. Seleccionar servidor proxy. RF58. Administrar servidor de sello de tiempo. RF59. Adicionar servidor de sello de tiempo. RF60. Ver detalles de servidor de sello de tiempo. RF61. Buscar servidor de sello de tiempo. RF62. Ver datos de servidor de sello de tiempo. RF63. Modificar servidor de sello de tiempo. RF64. Eliminar servidor de sello de tiempo.

2. Diagramas de casos de uso del módulo Configuración general

Para una mejor comprensión de los casos de uso del módulo Configuración general ver el Expediente de producto “HIS_Configuracion_general_Especificacion_de_casos_de_uso”.

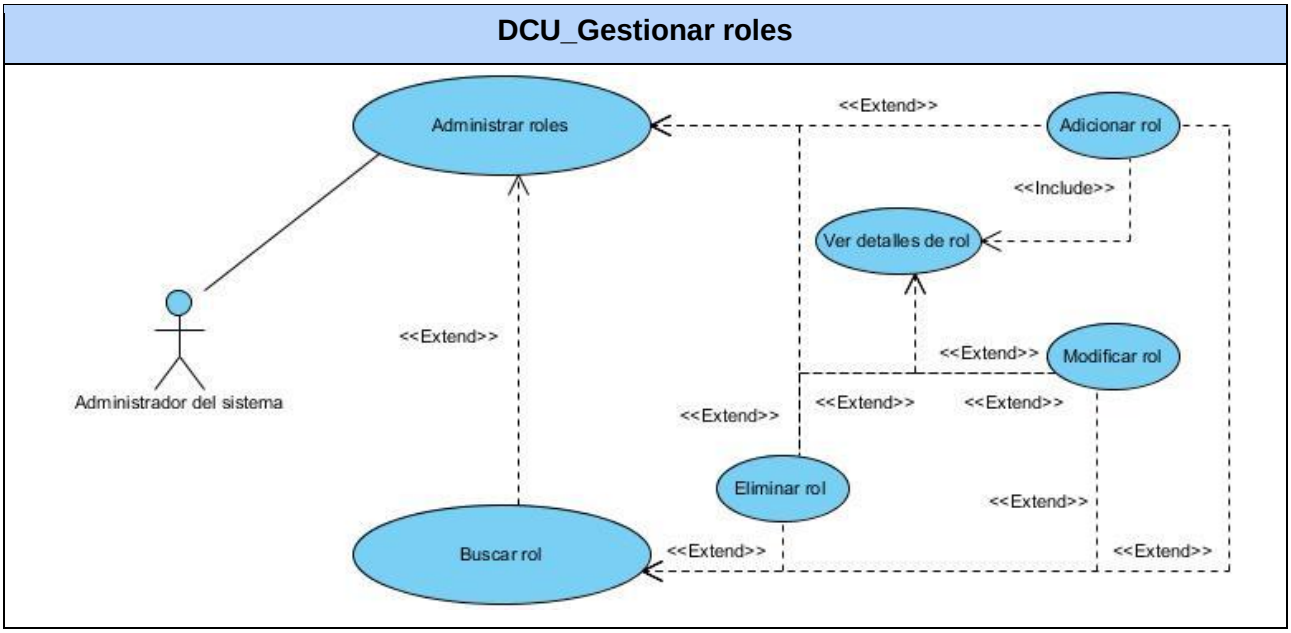

Figura 13: Diagrama de Casos de Uso – Gestionar roles

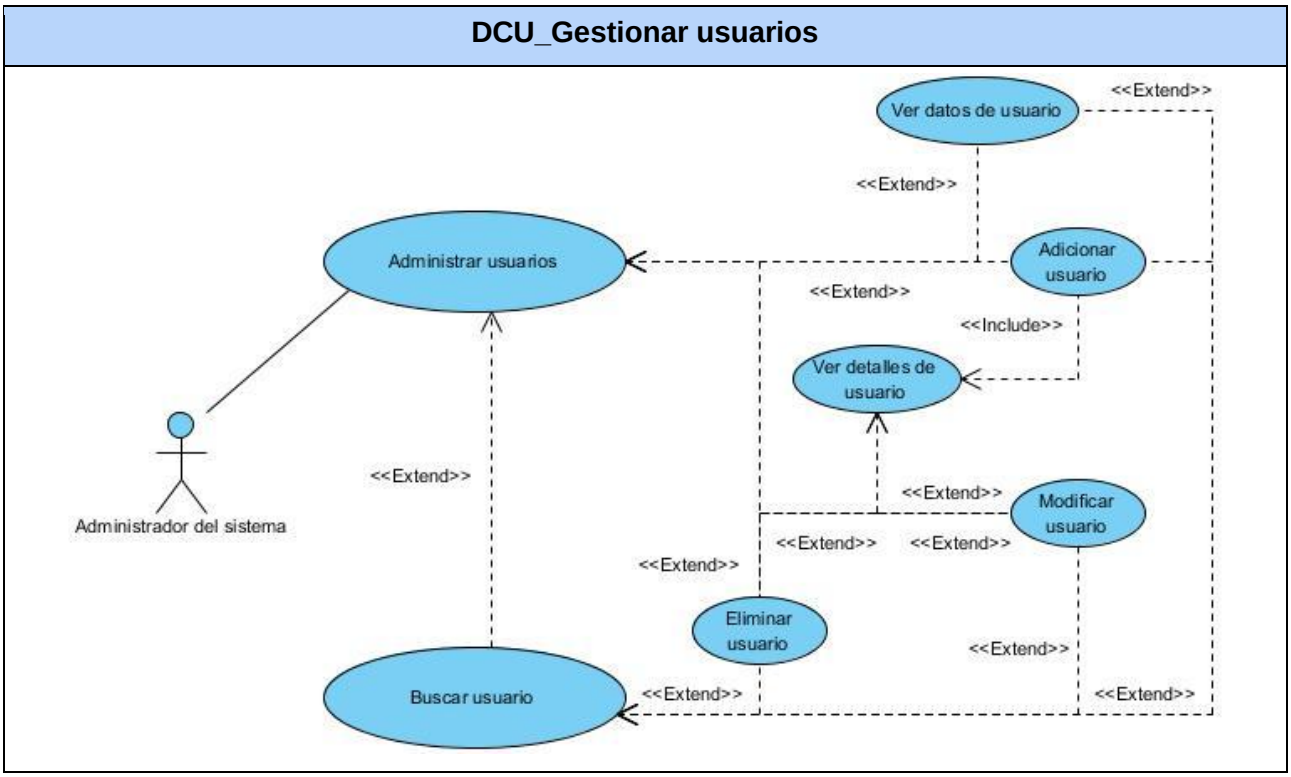

Figura 14: Diagrama de Casos de Uso – Gestionar usuarios

Figura 15: Diagrama de Casos de Uso – Administrar seguridad

3. Descripción de las clases de diseño del módulo Configuración general

A continuación se describen algunos de las clases de la solución propuesta. Las restantes descripciones se encuentran en el Expediente de producto "HIS_Configuración_general_Modelo_de_diseño".

EntidadSistemaSeleccionarControlador

Propósito: Esta clase controladora es la encargada de cambiar el estado de las entidades en visible y no visible. También se encarga de asociarlas y desasociarlas del anillo donde se encuentra el servidor.

Figura 16: Clase EntidadSistemaSeleccionarControlador

EntidadSistemaCrearControlador

Propósito: Esta clase controladora es la encargada de crear las entidades del sistema, así como referenciar tanto las entidades como los servicios de estas.

Figura 17: Clase EntidadSistemaCrearControlador

FuncionalidadesManager

Propósito: Esta clase controladora es la encargada de gestionar las funcionalidades del sistema, crear, eliminar, y modificar categorías y funcionalidades, así como ver los datos de las funcionalidades.

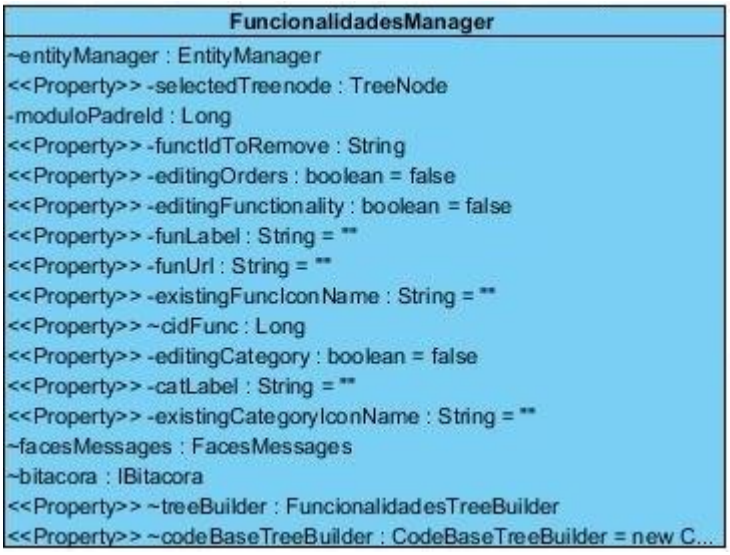

Figura 18: Clase FuncionalidadesManager

BitacoraReview

Propósito: Esta clase controladora es la encargada de buscar y mostrar las trazas de acciones del sistema.

Figura 19: Clase BitacoraReview

GLOSÁRIO DE TÉRMINOS

Ajax: Acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones. Estas aplicaciones se ejecutan en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre las páginas sin necesidad de recargarlas, lo que significa aumentar la interactividad, velocidad y usabilidad en las aplicaciones.

API: Acrónimo de Application Programming Interface (interfaz de programación de aplicaciones) es un conjunto de funciones que permite al programador acceder a servicios de una aplicación a través del uso de un lenguaje de programación.

Aplicación: Es un tipo de programa informático diseñado como herramienta para permitir a un usuario realizar uno o diversos tipos de trabajos.

Bitácora: Se le llama bitácora a un conjunto de registros de lo que sucede en un proyecto, sistema, etcétera. Puede registrar mucha información acerca de eventos relacionados con el sistema que la genera. Se usa para recuperar información ante incidentes de seguridad, detección de comportamiento inusual, información para resolver problemas, evidencia legal, es de gran ayuda en las tareas de cómputo forense.

Caching: Su principal funcionalidad es almacenar temporalmente los datos frecuentemente accedidos más cerca del solicitante de los mismos. El caching se ha usado desde hace mucho tiempo en sistemas de ordenadores y redes para aumentar las prestaciones.

Clustering: Es la división de los datos en grupos de objetos similares. Este proceso ayuda a la agrupación estructural de un conjunto de datos.

Código: Cifras, clave. En informática se utiliza para referirse a un conjunto de instrucciones en un lenguaje de programación.

Configuración: Es lo que determine cómo, a través de qué medios y con qué recursos funcionará un elemento.

Framework: Estructura predefinida para la creación de aplicaciones. Puede estar formado por un conjunto de librerías y clases o por una arquitectura que facilita el desarrollo de software.

HTTPS: Es un protocolo de aplicación basado en el protocolo HTTP, destinado a la transferencia segura de datos de hipertexto. Se usa en cada transacción de datos en la web. Es

usado por gran cantidad de entidades pues la información que manejan es de suma confidencialidad e importancia y por ello se necesita mantener cifrado el canal de transferencia.

Java Script: Es un lenguaje de programación interpretado, es decir, que no requiere compilación, utilizado principalmente en páginas web.

Patrón: Es el esqueleto de las soluciones a problemas comunes en el desarrollo de software. Definen una estructura común debido al aprendizaje pasado.

Persistencia: Es la acción de preservar la información de un objeto de forma permanente o puede referirse a recuperar la información del mismo para que pueda ser utilizada nuevamente, lo cual se interpreta como guardar y leer respectivamente.

Software: Término genérico que designa al conjunto de programas que posibilitan realizar una tarea específica en un ordenador.

TCP/IP: Acrónimo de Transmission Control Protocol/Internet Protocol (Protocolo de Control de Transmisión/Protocolo de Internet) es un sistema de protocolos que hacen posibles servicios FTP, E-mail, y otros entre ordenadores que no pertenecen a la misma red.