Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas.

Título: Desarrollo del componente de notificaciones del Sistema de Información Hospitalaria alas HIS.

Autores: Lianny Hernández De la Paz

Alexei Darias Jojorina

Tutores: Ing. Nadiezka Milan Cristo

Ing. Diuber Estanque Díaz

Co-tutor: Ing. Alejandro Luis Ortega Díaz

La Habana, julio 2013

"Año 55 de la Revolución"

Declaración de autoría

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.
Para que así conste firmamos la presente a los __ días del mes de ___ del año ____.

Lianny Hernández De la Paz

_____

Firma del autor

Alexei Darias Jojorina

_____

Firma del autor

Ing. Nadiezka Milan Cristo

_____

Firma del tutor

Ing. Diuber Estanque Díaz

_____

Firma del tutor

Ing. Alejandro Luis Ortega Díaz

_____

Firma del co-tutor

DATOS DE CONTACTO

Ing. Nadiezka Milan Cristo

Graduado en el año 2007 de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Jefe del Departamento de Sistemas de Gestión Hospitalaria. Ha impartido las asignaturas Inteligencia Artificial, Práctica Profesional y Gestión de Software.

Correo electrónico: nmilan@uci.cu

Ing. Diuber Estanque Díaz

Graduado en la Universidad de Ciencias Informáticas (UCI) como Ingeniero en Ciencias Informáticas en el año 2010. Especialista del departamento Sistemas de gestión Hospitalaria del Centro de Informática Medica (CESIM). Se ha desempeñado como líder del equipo de desarrollo del módulo de Facturación. Ha tutorado tres trabajos de diploma hasta el momento obteniéndose en cada uno de ellos la calificación máxima de 5 puntos.

Correo electrónico: destanque@uci.cu

Ing. Alejandro Luis Ortega Díaz

Graduado de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias informáticas en el año 2012. Actualmente trabaja como desarrollador del Departamento de Gestión Hospitalaria del centro de desarrollo CESIM en la Universidad de las Ciencias Informáticas. Sus líneas de trabajo se centran en tecnologías web, aplicaciones para la informática médica y la gestión hospitalaria.

Correo electrónico: alortega@uci.cu

AGRADECIMIENTOS

De Lianny:

Quiero agradecer a todas aquellas personas que desde un principio confiaron en mí y me alentaron a seguir siempre adelante con la frente en alto.

A mi abuela, por soportarme cuando los malos momentos cubrían mi mente y sobresalía mi mal genio, por tratar de entender lo que era la programación y por sufrir el desespero cuando la nota de una prueba tardaba más de lo acostumbrado.

A mi abuelo, mi papá y mi hermano porque han demostrado que mis cosas les importan y por escucharme cuando me pongo a hablar temas de mi carrera que a veces ni entienden.

A mis tíos, por ser los mejores que tengo, por malcriarme pero a la vez tratarme con mano ruda en todos los aspectos de la vida, por hacerme sentir que soy una hija y no una simple sobrina para ellos.

A mi tata, por ser la luz que siempre he seguido y sin saberlo me ha guiado por el mejor de los caminos.

A mi madre, por ser mi amiga, por abrirme los ojos, por darme ese carácter tan suyo que me impulsa a insistir para alcanzar aquello que realmente me interesa, por considerarme su Michy y por ser mi punto de apoyo.

A mis amigas de Matanzas, porque la distancia nunca fue motivo de separación entre nosotras.

A Diana, por decirme siempre la verdad por cruda que fuese, por ser más que una amiga y por hacerme saber que puedo contar con ella para lo que sea.

A Fara, por compartir los ataques de estrés y tolerar mi carácter.

A todas las amistades que hice en estos 5 años y a mi grupo, por ser especial.

A los profes, por contribuir a mi formación como ingeniera.

A mis tutores, por brindarme su apoyo, sus conocimientos y darme los instrumentos esenciales para comenzar a construir el trabajo donde se resumen 5 años de esfuerzo.

A Alexei, por ser mi maestro, mi compañero, mi amigo, mi confidente y hasta mi dolor de cabeza en estos últimos tiempos. Por sacarme una sonrisa, una lágrima y un bello sentimiento. Por ser el mejor compañero de tesis y por ayudarme, desde primer año, a cumplir este gran sueño.

A los que me escucharon, me ayudaron, me indicaron, me dieron su ejemplo y me criticaron para que mejorara.

De Alexei:

DEDICATORIA

De Lianny:

A mi hermano, por ser el pequeño que quiere seguir estos mismos pasos, espero que te haya servido de ejemplo y veas que todo esfuerzo vale la pena. Solo trázate una meta y verás que eres capaz de alcanzarla. Confío en ti.

De Alexei:

RESUMEN

Los correos electrónicos, los dispositivos de radio-búsqueda o radio-mensajería (beepers) y los teléfonos celulares, ofrecen potencialidades que pueden ser utilizadas en función de hacer más atractiva, organizada y eficiente la gestión hospitalaria, dando lugar a un intercambio mucho más rápido de la información. El Centro de Informática Médica (CESIM) desarrolla el Sistema de Información Hospitalaria alas HIS, solución informática para gestionar la información de los diferentes procesos de una institución hospitalaria. Dicho sistema no cuenta con un mecanismo para el envío de notificaciones a pacientes, o al personal médico que no se encuentre interactuando con la aplicación, lo cual puede provocar situaciones desfavorables que limitan la eficacia de los servicios del hospital. La presente investigación tiene como objetivo elaborar un componente de notificaciones basado en el uso del correo electrónico, el beeper y la telefonía celular, para el Sistema de Información Hospitalaria alas HIS. En el diseño del componente se utilizó la metodología de software Proceso Unificado de Desarrollo, la cual se apoyó en el Lenguaje de Modelado Unificado. Fue utilizado el entorno de desarrollo Eclipse, unido a Java como lenguaje de programación, PostgreSQL como Sistema Gestor de Base de Datos y Visual Paradigm para el modelado. Junto a estas herramientas se incorporaron frameworks y librerías. Todos los elementos anteriores giraron alrededor del patrón Modelo-Vista-Controlador. Luego de concluir el desarrollo se obtuvo como resultado un componente funcional y con atributos de seguridad.

Palabras claves: beeper, celular, componente, correo electrónico, notificaciones, Sistema de Información Hospitalaria.

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Las Tecnologías de la Información y las Comunicaciones, también conocidas como TIC, son el conjunto de tecnologías desarrolladas para gestionar información y enviarla de un lugar a otro. (1) Estas han introducido un nuevo paradigma respecto a cómo el hombre ve y hace las cosas, convirtiéndose en herramientas indispensables para el progreso eficiente de casi todas las áreas del conocimiento y la actividad humana, como son: la gestión de negocios, la industria, la investigación científica, la medicina, la física, la biología, la química, la ingeniería y muchas otras.

En estrecho vínculo con las TIC, la informática se ha manifestado en los diversos sectores sociales, aportando grandes beneficios en situaciones donde se hace necesario el almacenamiento y manipulación de considerables volúmenes de datos, una ejecución de cálculos matemáticos a grandes velocidades o un fácil y rápido acceso a determinada información.

En la actualidad, el sector de la salud es uno de los más favorecidos con el avance de la informática. La necesidad de aumentar la calidad de atención a los pacientes y mejorar el desempeño del personal médico en los diferentes niveles de atención, hace indispensable la puesta en práctica de nuevas tecnologías en este campo. Dentro de este gran grupo cabe resaltar medios tecnológicos como el correo electrónico, los dispositivos de radio-búsqueda o radio-mensajería (beepers) y la telefonía celular, cuyas potencialidades pueden ser utilizadas en función de hacer más atractiva, organizada y eficiente la gestión de la información hospitalaria, dando lugar a un intercambio mucho más rápido de la misma.

De igual manera se destacan los llamados Sistemas de Información Hospitalaria (HIS, por sus siglas en inglés). Estos están orientados a satisfacer las necesidades de generación de información para almacenar, procesar y reinterpretar datos médico-administrativos dentro de cualquier institución hospitalaria (2), facilitando la toma de decisiones y la realización de las actividades en los diferentes niveles de dichas entidades y la optimización de los recursos humanos y materiales.

El Centro de Informática Médica (CESIM) de la Universidad de las Ciencias Informáticas desarrolla el Sistema de Información Hospitalaria alas HIS, con el objetivo de gestionar los datos de los diferentes procesos de una institución hospitalaria. Este sistema cuenta con varios módulos que interconectan en la aplicación las distintas áreas de un centro de salud tales como: Banco de Sangre, Epidemiología, Anatomía Patológica, Bloque Quirúrgico, Emergencia, Hospitalización, Consulta Externa, entre otros.

Estos manejan información referente a las especialidades para las cuales han sido diseñados y responden a las necesidades específicas de cada una de ellas.

Actualmente el Sistema de Información Hospitalaria alas HIS no cuenta con un mecanismo para el envío de notificaciones al personal de salud, basado en el uso del correo electrónico, la radio-mensajería o la telefonía celular, reduciendo los espacios de intercambio con los usuarios solamente a la aplicación web. Por otro lado, no existe una forma de notificar a los pacientes desde el sistema. Esta situación puede afectar el buen funcionamiento de los procesos hospitalarios, puesto que para hacer llegar determinada información desde el alas HIS, los médicos y demás trabajadores deben encontrarse interactuando directamente con el mismo. De fallar la comunicación con el personal de salud o los pacientes, podrían surgir situaciones desfavorables, como la presencia de una persona en un centro asistencial con el objetivo de adquirir los resultados de sus exámenes y estos no encontrarse disponibles, afectaciones tanto a médicos como pacientes por la imposibilidad de informarles, en tiempo real, cambios de horario o una reprogramación de consulta. Además, se puede dar lugar a una atención tardía a pacientes con enfermedades contagiosas por el hecho de obtener los resultados de los análisis una vez retirada la persona enferma del hospital y no poseer la vía inmediata para citarla a la institución de salud. Por otra parte, a los médicos en horario de descanso que dan seguimiento a dolientes internos, no se les puede hacer saber rápidamente que la situación de esas personas pudo haber empeorado, lo que propicia que al tener que atenderlas de manera urgente otro doctor, no se les ponga en práctica el tratamiento que más se ajusta a sus condiciones de salud.

El presente trabajo surge para darle solución a la situación antes expuesta, por lo que el ***problema a resolver*** queda formulado de la siguiente forma: ¿Cómo notificar, en tiempo real, a pacientes que no se encuentren en el hospital y al personal de salud que no esté interactuando directamente con el sistema alas HIS?

Considerando como ***objeto de estudio*** los procesos para notificar información en tiempo real. Enmarcado en el ***campo de acción*** los procesos para notificar información en tiempo real basados en el uso del correo electrónico, los dispositivos de radio-búsqueda o radio-mensajería (beepers) y la telefonía celular.

Para solucionar el problema identificado se define como ***objetivo general:*** Elaborar un componente de notificaciones basado en el uso del correo electrónico, los dispositivos de radio-búsqueda o radio-mensajería (beepers) y la telefonía celular para el Sistema de Información Hospitalaria alas HIS.

Para dar cumplimiento al objetivo planteado, se precisan las siguientes ***tareas de la investigación:***

1. Estudiar las tendencias actuales de las soluciones informáticas relacionadas con el envío de correos electrónicos, radio-mensajes y el Servicio de Mensajes Cortos (SMS, por sus siglas en inglés).
2. Asimilar las tecnologías y la arquitectura definidas para el Sistema de Información Hospitalaria alas HIS.
3. Obtener la documentación correspondiente al componente de notificaciones, según la metodología Proceso Unificado de Desarrollo (RUP).
4. Realizar el Modelo de Dominio con los conceptos presentes en el ámbito del sistema.
5. Describir los requisitos del componente a desarrollar.
6. Diseñar las clases que permitan cumplir con los requerimientos descritos.
7. Proponer una solución basada en estándares y patrones.
8. Implementar las funcionalidades necesarias para desarrollar el componente de notificaciones para el Sistema de Información Hospitalaria alas HIS.

Con el desarrollo de las funcionalidades asociadas al componente de notificaciones del sistema alas HIS se esperan obtener los siguientes ***beneficios***:

1. Existencia de un componente capaz de notificar a los pacientes, cuando los mismos se encuentren fuera del centro de salud, sobre la disponibilidad de los resultados de sus análisis, la confirmación de una cita, un cambio de horario repentino, la disponibilidad de un medicamento específico en la farmacia, la alerta sobre un virus o epidemia que se haya detectado, un nuevo plan de vacunación a la población, entre otras informaciones que pueden ser suministradas.
2. Disponibilidad de una forma rápida de comunicación con el personal de salud cuando se encuentre fuera de la institución hospitalaria y ante situaciones de urgencia.
3. Provisión de vías alternativas para el flujo de información entre las áreas del hospital, lo que contribuye a lograr independencia de los usuarios con el sistema a la hora de recibir notificaciones.
4. Aumento de la competitividad del sistema alas HIS en el mercado internacional, al contar con un mecanismo de notificaciones que hace uso de algunos de los medios tecnológicos más empleados

y adquiridos por la población en la actualidad y que no se encuentra integrado a todos los Sistemas de Información Hospitalaria existentes.

El documento se encuentra dividido en cuatro capítulos, estructurados de la siguiente manera:

Capítulo 1: Fundamentación Teórica: Se presentan los principales conceptos que constituyen las bases para el desarrollo del componente de notificaciones. Se realiza un estudio preliminar de varios sistemas automatizados de avisos, vinculados a las instituciones de salud. Se fundamentan las tecnologías, metodologías y herramientas de desarrollo a utilizar.

Capítulo 2: Características del sistema: Se describen las principales características del sistema. Se presenta el Modelo de Dominio de la aplicación, conjuntamente con la especificación de los requerimientos funcionales y no funcionales y el Modelo de Casos de Usos del Sistema.

Capítulo 3: Análisis y diseño del sistema: Se realiza una descripción y análisis de la estructura de la solución que se propone para dar respuesta a la problemática planteada. Se fundamenta la arquitectura empleada, así como las estrategias de integración a tener en cuenta.

Capítulo 4: Implementación: Se introduce el flujo de trabajo de implementación, partiendo de los resultados obtenidos en el diseño. Se detalla el Modelo de Datos, en el que se ve la estructura donde se almacena toda la información requerida en el sistema y se exponen aspectos referentes a la seguridad del sistema, las estrategias de codificación, así como la forma en que se tratarán los errores.

CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA DEL COMPONENTE DE NOTIFICACIONES

En el presente capítulo se realiza un estudio de los principales conceptos y características que constituyen las bases para la implementación del componente de notificaciones, se muestran breves descripciones de sistemas vinculados al campo de acción del problema a resolver y se fundamenta la selección de tecnologías, metodologías y herramientas de software con las que se llevará a cabo el proceso de desarrollo, a partir de la arquitectura definida para el Sistema de Información Hospitalaria alas HIS.

1.1 Conceptos básicos relacionados con el dominio del problema

En el ámbito de la informática, el proceso de notificación consiste en la acción de comunicar formalmente una resolución o noticia con propósito cierto, advirtiendo sobre determinadas situaciones a los usuarios del sistema. (3)

Existen diversas vías para realizar notificaciones de forma rápida y segura, como pueden ser: el correo electrónico, el beeper y el teléfono celular.

La notificación por correo electrónico es aquella comunicación orientada a las direcciones electrónicas de los usuarios. Esta vía para notificar posibilita que la persona a la cual se le ha enviado el mensaje lo reciba rápidamente y acceda al contenido del mismo desde la comodidad de su hogar, centro laboral o cualquier lugar con acceso a la red, sin necesidad de trasladarse hacia donde se encuentra el emisor de dicha información.

El beeper, o también llamado buscapersonas, generalmente es un dispositivo muy sencillo que incluye una pantalla de cristal líquido, una alerta vibratoria y/o sonora y botones de control. Este facilita la localización de las personas para fines laborales y sociales, mediante un sistema de radio-mensajes, que viajan a través de un canal radioeléctrico predeterminado para este fin. (4) La notificación hacia beepers consiste en el envío de mensajes a dichos dispositivos, garantizando un aviso inmediato de cualquier eventualidad.

El Sistema Global de Comunicaciones Móviles (GSM, por sus siglas en inglés), es un estándar de telefonía móvil digital con un rendimiento máximo de 9.6 kbps, con lo cual se pueden realizar transmisiones de voz y de datos digitales, por ejemplo, mensajes de texto o mensajes multimedia. En una

red GSM, la terminal del usuario se llama estación móvil, la cual está constituida por una tarjeta SIM (Módulo de Identificación de Abonado), que permite identificar de manera única al usuario y a la terminal móvil, o sea, al dispositivo del usuario (normalmente un teléfono móvil o un módem). (5)

Un teléfono móvil, también conocido como celular, es un dispositivo inalámbrico que se puede trasladar de manera fácil. El funcionamiento de este tipo de teléfono está dado por ondas de radios que le permiten acceder a las antenas que conforman la red de la telefonía móvil. (6) El envío de notificaciones a celulares consiste en la remisión de mensajes SMS hacia dichos dispositivos (cadenas alfanuméricas de hasta 160 caracteres).

Un módem GSM puede verse como un teléfono celular al cual se le ha adaptado una interfaz serial RS232, con el objetivo de ser controlado por una computadora. A través del módem GSM pueden realizarse enlaces para la trasmisión de voz, fax, datos y mensajes SMS. También existen módems GSM que poseen una interfaz USB (Universal Serie Bus) para su conexión. Una de las aplicaciones de estos dispositivos es el envío programado de mensajes SMS a teléfonos móviles. (7)

En cada una de las formas de comunicación anteriores (vía correo electrónico, beeper, celular o módem GSM), intervienen un emisor y uno o varios receptores. Particularmente, en el componente de notificaciones para el sistema alas HIS, se entiende como emisores a los médicos, laboratoristas, almaceneros, farmacéuticos, directivos, administradores y demás usuarios registrados en la aplicación, mientras que los receptores son los pacientes y el propio personal de salud de las instituciones hospitalarias.

1.2 Sistemas automatizados de notificaciones

Los Sistemas de Información Hospitalaria se utilizan en varios países del mundo y presentan diversas características y funcionalidades. Aunque todos tienen como objetivo principal lograr una gestión eficiente de la información de las instituciones de salud, algunos integran soluciones que los hacen más útiles y completos, como son los componentes de notificaciones. A continuación se relacionan algunos HIS y otros sistemas vinculados al área de la salud, en cuyas funcionalidades se incluyen mecanismos para el envío automatizado de avisos.

Michigan es un Sistema de Información Hospitalaria desarrollado por la empresa Michigan Ingeniería Informática S.A., utilizado en algunos centros de salud de Argentina y que cuenta con una serie de

módulos dedicados a la gestión hospitalaria. Desde el módulo Control de Accesos y Horarios se permite la generación y envío de correos electrónicos automáticos a los destinatarios necesarios, con información de las llegadas tarde y las ausencias del personal médico. Además, posibilita la remisión de correos electrónicos a modo de notificaciones a trabajadores y pacientes. También, desde el módulo Compras, se facilita la emisión de órdenes de compra de medicamentos vía e-mail. (8)

SALUS es un software para la gestión integral de clínicas, centros médicos y hospitales, desarrollado por la empresa española QSOFT, que permite gestionar de forma global el conjunto de áreas de gestión de un centro de salud. Para facilitar la organización y comunicación entre profesionales del mismo centro, SALUS implementa un sistema de mensajería entre usuarios del sistema. Cada mensaje puede tratarse como una tarea a realizar de forma que quien la envía siempre tiene confirmación de cuándo ha sido vista por el receptor y cuándo éste marca la tarea como realizada. Permite además comunicar datos de manera automática a los pacientes, como por ejemplo, el recordatorio de una cita próxima o el resultado de una prueba diagnóstica, mensajes que se envían directamente a su teléfono móvil. (9)

Net Clínicas es un sistema de gestión clínica creado por el departamento de desarrollo de software español NetClínicas Software. Brinda un programa configurable y adaptable a todo tipo de clínica y es muy utilizado en la actualidad. Permite gestionar distintas áreas que van desde el control de historias clínicas hasta el registro de facturas. Contiene una agenda de excelencia para archivar citas y eventos, gestiona toda la documentación del centro, tiene editores de formularios y bases de datos de los pacientes. Dispone además del envío automático de mensajes SMS a modo de recordatorio de los eventos a realizar. El costo de adquisición de Net Clínicas (versión multi-puesto) es de 455 euros. (10) Presenta licencia de software propietario y es una aplicación de escritorio, compatible únicamente con el sistema operativo Windows.

Sistema de recordatorio SMS es un sistema de software desarrollado por Esendex, empresa líder en el mercado europeo de mensajes SMS, incluido en los centros hospitalarios del Instituto Catalán de la Salud (ICS) en España. Brinda un servicio de recordatorios de forma automatizada desde una plataforma compartida por los hospitales del ICS, donde los usuarios reciben un mensaje de texto al móvil con información sobre la fecha, la hora y el lugar de su consulta, así como un número de teléfono de contacto para cualquier duda o cambio de cita. Por otra parte, los usuarios del sistema de salud son informados de cuándo pueden ir a recoger los resultados de distintas pruebas sanitarias. En la red de atención primaria

de dichos centros se potencia además la notificación vía correo electrónico. (11) Las aplicaciones de Esendex son de uso gratuito, por lo que no se tiene que pagar por derechos de licencia para su empleo, sin embargo, está establecido un sistema de precios para los envíos de SMS.

Mithos es un sistema de solicitud y asignación de turnos vía SMS que se implementa en el hospital Bouquet Roldán de la provincia Neuquén, en Argentina. Este servicio ofrece una rápida solución a la problemática que genera la solicitud de turnos médicos, siendo una alternativa más para los usuarios, quienes también pueden recibir respuesta de forma personal o telefónica. La persona interesada en registrar una cita envía un mensaje de texto a un número fijo con su identificación, la especialidad en la que requiere ser atendida y la fecha en la que quiere reservar. Una vez que se procesa el mensaje, el sistema de turnos responderá con el texto "Solicitud Realizada Correctamente". Luego el solicitante recibirá las opciones con los turnos disponibles para elegir y finalmente, le llegará un mensaje con los datos de la confirmación de turno y un número de transacción. Entre las ventajas de su utilización se encuentran las siguientes: facilita la obtención de turnos por los pacientes, libera recursos tanto de personal como de líneas telefónicas, incrementa la tasa de asistencia mediante recordatorios automáticos y la reserva de turnos es inmediata e instantánea, posibilitando tener una “ventanilla virtual” las 24 horas del día. Constituye un ejemplo de aplicación E-Mobile, siendo este último un conjunto de herramientas y procesos orientados a facilitar el desarrollo de sistemas que interactúan con dispositivos móviles. La plataforma tecnológica de E-Mobile comprende la utilización de Oracle como Sistema Gestor de Base de Datos, el lenguaje de programación Java, GlassFish como servidor de aplicaciones y el uso de servicios web. (12)

Sistema de Información Clínico-Hospitalaria del hospital Padua, Italia. Posee un componente computarizado de notificaciones, envío de alertas y SMS, que comenzó a utilizarse en enero del 2008 y fue desarrollado por la Sociedad de Procesamiento de Datos Médicos (GMD, por sus siglas en alemán). Este sistema gestiona todas las peticiones realizadas por los médicos y reúne la información procedente de todos los servicios de diagnóstico, incluyendo el laboratorio clínico y los departamentos de imagen. En tiempo real, se notifican a los médicos, de forma automática, los resultados del laboratorio, mediante un mensaje de correo electrónico. Para el envío de dichos resultados, se generan además dos mensajes: un SMS hacia el teléfono móvil del médico de turno, y otro a nivel de departamento, que llega al monitor del clínico que ordenó el análisis. Ambos tipos de mensaje (el SMS y la alerta) incluyen los códigos

apropiados para la identificación del paciente y especifican el número de teléfono móvil del médico de guardia en el laboratorio. El sistema de información de la institución lleva una constancia del éxito o el fracaso de cada proceso de notificación. (13)

Con la realización de un estudio de los diferentes sistemas existentes que dan solución total o parcial al problema relacionado con la notificación automatizada en los centros hospitalarios, se arriban a las siguientes conclusiones:

La mayoría de los sistemas analizados no brindan información suficiente sobre las funcionalidades con que cuentan o las herramientas con las que fueron desarrollados, por lo que resulta difícil decidir si se ajustan o no a las necesidades reales del sistema alas HIS. Algunos son desarrollados con una combinación de tecnologías libres y privativas, y poseen licencia de software propietario, lo que dificulta obtener versiones funcionales de los mismos para su estudio. Se debe resaltar además que muchos de estos sistemas notifican únicamente por la vía del teléfono celular.

Por las razones anteriores se decide desarrollar un componente de notificaciones basado en el uso de medios tecnológicos como el correo, el beeper y el teléfono celular para el Sistema de Información Hospitalaria alas HIS, con el fin de obtener una solución propia, desarrollada sobre tecnologías libres. De esta forma, la información que se transmite será gestionada por el sistema alas HIS y no por empresas o aplicaciones informáticas extranjeras, lo que contribuye a ganar en confidencialidad y seguridad.

1.3 Tendencias y tecnologías actuales a considerar

Un paso fundamental en el inicio del desarrollo de un proyecto es la realización de un estudio de las tecnologías en dependencia del lenguaje de programación que se va a utilizar, definiendo cuáles serán las herramientas y técnicas con las que se va a trabajar para crear un producto con la calidad requerida.

Para el desarrollo de un mecanismo de notificaciones integrable al sistema alas HIS, se realizó un estudio de la arquitectura del mismo, por lo que se asumió el uso de herramientas de software libre con el objetivo de obtener un componente sin licencias propietarias, minimizando los costes y maximizando los resultados. Igualmente se aceptó la utilización de tecnologías que garanticen que el componente sea configurable y multiplataforma.

1.3.1 Arquitectura cliente-servidor

El sistema alas HIS utiliza la arquitectura cliente-servidor. Esta arquitectura intenta proveer usabilidad, flexibilidad, interoperabilidad y escalabilidad en las comunicaciones. Permite a los usuarios finales obtener acceso a la información de forma transparente, desde diferentes lugares y aún en entornos multiplataforma. Este modelo consiste básicamente en que el cliente envía un mensaje solicitando un determinado servicio a un servidor (hace una petición), que envía uno o varios mensajes con la respuesta (provee el servicio solicitado). (14) El modelo cliente/servidor se recomienda para redes que requieran un alto grado de fiabilidad. Presenta las ventajas de: recursos centralizados (se administran a nivel de servidor los recursos comunes a todos los usuarios, por ejemplo, una base de datos centralizada), seguridad mejorada y red escalable (es posible quitar o agregar clientes sin afectar el funcionamiento de la red y sin la necesidad de realizar mayores modificaciones). (15)

1.3.2 Arquitectura en capas

La arquitectura en capas es un estilo de diseño cuyo objetivo principal es la separación y agrupamiento de los componentes del software atendiendo a la función que cumplen en el mismo. Proporciona una centralización de los aspectos de seguridad y transaccionalidad. Para realizar el asociamiento se tienen en cuenta las funcionalidades relacionadas con el usuario del sistema, así como la información que éste gestiona y las operaciones que realiza sobre la misma en dependencia de la complejidad que se necesita que tenga el sistema. La división mayormente se realiza en tres capas: la capa de presentación, la capa de negocio y la capa de datos. (16)

1.3.3 Patrones de Arquitectura y Diseño

Un patrón es un esquema o modelo genérico que permite solucionar un problema particular y que provee un vocabulario común y comprensible. Los patrones de arquitectura expresan un esquema organizativo estructural para los sistemas de software, mientras que los de diseño, definen modelos para detallar estructuras de diseño (o sus relaciones), con los que construir una aplicación. En general, facilitan el desarrollo de programas con propiedades definidas (propiedades particulares). Para la realización del componente de notificaciones del sistema alas HIS, se utilizó el patrón de diseño de arquitectura Modelo-Vista-Controlador.

1.3.3.1 Modelo-Vista-Controlador (MVC)

El patrón MVC está diseñado para reducir el esfuerzo de programación necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos. Hace a los sistemas más flexibles y adaptables. Su característica principal es que el Modelo, las Vistas y los Controladores se tratan como entidades separadas, lo que permite su desarrollo independiente, garantizando así la actualización y mantenimiento del software de forma sencilla y en un reducido espacio de tiempo. Además presenta la ventaja de que la conexión entre el Modelo y sus Vistas es dinámica, o sea, se produce en tiempo de ejecución, no en tiempo de compilación. (17)

Modelo: El Modelo es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El Modelo no tiene conocimiento específico de los Controladores o de las Vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el Modelo y sus Vistas, y notificar a las Vistas cuando cambia el Modelo. (17)

Vista: La Vista es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del Modelo y muestra los datos al usuario. Interactúa preferentemente con el Controlador, pero es posible que trate directamente con el Modelo a través de una referencia al propio Modelo. (17)

Controlador: El Controlador es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo, centra toda la interacción entre la Vista y el Modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo. (17)

1.3.4 Tecnologías utilizadas en el proceso de desarrollo

A partir de la necesidad de integrar un componente de notificaciones al sistema alas HIS y siguiendo la línea de desarrollo del Departamento de Sistemas de Gestión Hospitalaria, se propone utilizar tecnologías y herramientas que permitan su uso sin necesidad del pago por su licencia. A continuación se describe el lenguaje de programación empleado y se presentan las tecnologías asumidas, según su ubicación dentro de la arquitectura en capas (presentación, negocio, acceso a datos). Se mostrará además una relación de las herramientas propuestas para el desarrollo del proyecto en cuestión.

1.3.4.1 Java

Java es un lenguaje de programación multiplataforma y orientado a objetos, fue diseñado para crear software altamente fiable para lo que proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Es altamente seguro e indiferente a la arquitectura pues está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red. (18) Logra cumplir el principal objetivo que persigue el sistema alas HIS que es estar libre del costo relacionado con patentes de software**,** asociadas al servidor de aplicaciones, al servidor de base de datos, al sistema operativo huésped u otras herramientas o tecnologías utilizadas para su desarrollo.

1.3.4.2 Capa de presentación

La capa de presentación es la que exhibe el sistema al usuario, le comunica la información y captura la que este introduce en un mínimo de procesos. Esta capa se comunica únicamente con la capa de negocio. (19)

1.3.4.2.1 Java Server Faces (JSF)

JSF es una tecnología y un ambiente de desarrollo o corrida (framework) que facilita y agiliza la construcción de interfaces de usuario, pues realiza la programación a través de componentes y es basada en eventos. (20) Los componentes JSF facilitan la construcción de interfaces web del lado del servidor, se conectan a fuentes de datos y relacionan de forma transparente eventos del cliente con manejadores en el servidor. (21)

1.3.4.2.2 RichFaces

RichFaces es una librería de componentes enriquecidos para JSF y además posee un framework avanzado para la integración sencilla de las funcionalidades Ajax dentro del desarrollo de las aplicaciones del negocio. RichFaces incluye ciclo de vida, validaciones, conversiones y la gestión de recursos estáticos y dinámicos. Los componentes de RichFaces están construidos con soporte Ajax, que puede ser fácilmente incorporado dentro de las aplicaciones JSF. (22)

1.3.4.2.3 Ajax4jsf

Ajax4jsf es una librería de código abierto que se integra totalmente a la arquitectura de JSF y extiende la funcionalidad de sus etiquetas, dotándolas con tecnología Ajax de forma limpia y sin necesidad de añadir código JavaScript. Mediante este framework se puede variar el ciclo de vida de una petición JSF, recargar determinados componentes de la página sin necesidad de refrescarla por completo, realizar peticiones automáticas al servidor y controlar cualquier evento de usuario. (23)

1.3.4.2.4 Facelets

JavaServer Facelets es un framework para plantillas. Es de código abierto, distribuido bajo la licencia Apache y constituye una tecnología alternativa de controlador de vista de JSF. No depende de un contenedor web. Soporta todos los componentes de interfaz de usuario JSF y construye su propio árbol de componentes. (24)

1.3.4.2.5 XHTML

Lenguaje de Marcado de Hipertexto Extensible (XHTML, por sus siglas en inglés), es una versión más estricta y limpia del Lenguaje de Marcado de Hipertexto (HTML, por sus siglas en inglés), que nace precisamente con el objetivo de remplazar a HTML ante su limitación de uso con las herramientas basadas en el Lenguaje de Marcado Extensible (XML). Su objetivo es avanzar en el proyecto del World Wide Web Consortium (W3C) de lograr una web semántica, donde la información y la forma de presentarla estén claramente separadas. (25)

1.3.4.2.6 Seam UI

Seam UI constituye una serie de controles JSF altamente integrables con JBoss Seam que añaden mejoras a JSF, desde validación, expresiones Extended EL, hasta la integración de la navegación en la interfaz de usuario basada en flujo de páginas o procesos del negocio.

1.3.4.3 Capa de negocio

La capa de negocio es donde residen los programas que se ejecutan, recibiendo las peticiones del usuario y enviando las respuestas tras el proceso. Se denomina capa de negocio porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación,

para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al Sistema Gestor de Base de Datos almacenar o recuperar información. (19)

1.3.4.3.1 Seam

Seam es una potente plataforma de desarrollo de código abierto utilizada para construir aplicaciones de Internet en Java. Integra tecnologías como Ajax, JSF, Java Persistence API (JPA), Enterprise Java Beans (EJB 3.0) y Business Process Management (BPM) en un sistema unificado con sofisticadas herramientas.

1.3.4.3.2 JavaMail

JavaMail es una expansión de Java que facilita el envío y recepción de correos electrónicos desde código Java de forma rápida y segura. Sus funcionalidades pueden ser utilizadas desde cualquier plataforma. Contiene el paquete javax.mail que define clases que son comunes a todos los sistemas de correo. Por otra parte, el paquete javax.mail.internet provee clases específicas para los sistemas de correo basados en los estándares de Internet, como son: MIME (Multipurpose Internet Mail Extensions o Extensiones Multipropósito de Correo de Internet), SMTP (Simple Mail Transfer Protocol o Protocolo para la transferencia simple de correo electrónico), POP3 (Post Office Protocol o Protocolo de Oficina de Correo) e IMAP (Internet Message Access Protocol o Protocolo de acceso a mensajes electrónicos). De esta forma, JavaMail está diseñada mediante una arquitectura en capas: la Capa de Abstracción declara clases, interfaces y métodos abstractos destinados a apoyar las funciones de manejo de todos los sistemas de correos soportados, mientras que la Capa de Implementación desarrolla parte de los elementos de la Capa de Abstracción usando las normas RFC822 y MIME. JavaMail utiliza el JavaBeans Activation Framework (JAF) para encapsular los datos de los mensajes, y para manejar comandos destinados a interactuar con esos datos. Permite crear un mensaje de correo electrónico a partir de una colección de atributos de cabecera y un bloque de datos, utiliza la interfaz Part y la clase Message para definir dicho mensaje. Posibilita la creación de un objeto Session que autentica al usuario y controla el acceso al depósito de mensajes y el transporte de estos. De igual forma, permite enviar un mensaje a una lista de receptores y recuperar un mensaje que se encuentre almacenado. JavaMail no tiene mecanismos que soporten el estado de entrega del mensaje, operaciones de desconexión, servicio de directorios o funciones de filtro. (26)

1.3.4.3.3 SMSLib

SMSLib es una librería Java que permite enviar y recibir mensajes SMS a través de un módem o teléfono GSM. La misma se distribuye bajo los términos de la licencia Apache v2, la cual está certificada por la organización Iniciativa para el Código Abierto (OSI, por sus siglas en inglés) y es libre según las Directrices de Software Libre de Debian. Puede ser usada desde los sistemas operativos Windows y Linux. Posibilita el envío de mensajes de texto desde los operadores internacionales BulkSMS, Clickatell y EzTexting, usando el Protocolo de Transferencia de Hipertexto (HTTP) o el Protocolo de Transferencia de Hipertexto Seguro (HTTPS). Los módems o teléfonos GSM pueden ser conectados a través de interfaces de puerto serie o interfaces IP, o por puerto USB. La librería viabiliza además el envío y recepción de mensajes encriptados y de gran tamaño y ofrece información básica de los dispositivos GSM conectados (fabricante, nivel de señal, modelo, entre otros datos). SMSLib da tratamiento a los SMS modo texto y modo PDU (Protocol Description Unit) y soporta el protocolo SMPP (Short Message Peer-to-Peer). Aunque la tecnología GSM tiene un límite de envío de alrededor de 6 mensajes por minuto, la librería ofrece la ventaja de poder configurar múltiples pasarelas de salida. Este problema también se resuelve con el uso de los operadores internacionales compatibles. (27)

1.3.4.4 Capa de acceso a datos

La capa de acceso a datos contiene las clases que interactúan con la base de datos, estas clases, utilizando los procedimientos almacenados (funciones para interactuar con la base de datos) que se generan, realizan todas las operaciones con la base de datos de forma transparente para la capa de negocio.

1.3.4.4.1 Hibernate

Hibernate es una herramienta de mapeo objeto-relacional para la plataforma Java, 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) que permiten establecer estas relaciones. Es una tecnología de software libre distribuida bajo los términos de la licencia GNU Lesser General Public License (LGPL). Esta herramienta ofrece también un lenguaje de consulta de datos llamado Hibernate Query Language (HQL),

al mismo tiempo que una Interfaz de Programación de Aplicaciones (API, por sus siglas en inglés) para construir las consultas programáticamente. (28)

1.3.4.4.2 Enterprise Java Beans (EJB3)

La tecnología Enterprise JavaBeans (EJB) es un componente de arquitectura del lado del servidor para la plataforma Java Enterprise Edition (Java EE). El estándar EJB3 es desarrollado por Java Community Process (JCP). Encapsula la lógica del negocio de una aplicación y permite construir aplicaciones de negocio portables, reutilizables y escalables usando el lenguaje de programación Java. EJB3 puede residir en diferentes servidores y puede ser invocado por un cliente remoto. La lógica del negocio reside en los Enterprise Beans y no en el lado del cliente, permitiendo que el desarrollo del lado del cliente esté desacoplado de la lógica del negocio. (29).

1.3.4.4.3 Java Persistence API (JPA)

Java Persistence API (JPA), es la API de persistencia desarrollada para la plataforma Java EE y está incluida en el estándar EJB3. Esta API busca unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue su diseño es no perder las ventajas de la orientación a objetos al interactuar con una base de datos, como sucedía con EJB2, y permitir usar objetos regulares conocidos como POJO (Plain Old Java Object). (30)

1.4 Tecnologías horizontales

1.4.1 Java Enterprise Edition 5 (JEE 5)

Es la plataforma que provee Sun Microsystem para dar soporte y desarrollar software para las empresas. JEE está enfocado en la creación de aplicaciones empresariales, las cuales se caracterizan por resolver problemas corporativos, lo que supone almacenamiento seguro, concurrencia, seguridad, escalabilidad y procesamiento distribuido. Dichos sistemas están construidos sobre una infraestructura base: los servidores de aplicaciones. Se puede definir a JEE como una arquitectura distribuida y multicapa, con especificaciones para empaquetar componentes redistribuibles para despliegue. La quinta edición dio un gran paso hacia la excelencia, pues incorporó nuevas tecnologías de punta que le proporcionaron a la plataforma gran robustez y simplicidad a la hora de trabajar. (21)

1.4.2 JBoss Application Server

JBoss Application Server es un servidor multiplataforma de aplicaciones J2EE, de código abierto, implementado en Java puro y orientado a la arquitectura de servicios. Por ser una plataforma certificada J2EE, soporta todas las funcionalidades de J2EE 1.4 e incluye servicios adicionales como clustering y persistencia. JBoss es ideal para aplicaciones Java y aplicaciones web. También soporta EJB 3.0, lo que hace el desarrollo de las aplicaciones mucho más simple. Con su utilización se pueden implementar sistemas con facilidad, únicamente es necesario colocar los archivos en el directorio /server/default/deploy. (31)

1.4.3 Java Virtual Machine

La Máquina Virtual Java es el núcleo del lenguaje de programación Java. De hecho, es imposible ejecutar un programa Java sin ejecutar alguna implantación de la misma. Es la clave de muchas de las características principales de Java, como la portabilidad, la eficiencia y la seguridad. Se encarga de interpretar todo el código java y convertirlo al lenguaje nativo del sistema operativo en uso. (32)

1.5 Metodologías de desarrollo de software

Para obtener un producto que cumpla con todas las especificaciones y esté en el tiempo estimado, se necesita una metodología que guíe el proceso de desarrollo de software. Una metodología es la que define quién debe hacer qué, cuándo y cómo. Es un proceso donde se realizan tareas para obtener un resultado a través de un conjunto de actividades definidas previamente. En la actualidad no existe una metodología universal que se le pueda aplicar a todos los proyectos por lo que su selección depende de las características de cada uno de ellos.

Para el desarrollo del componente de notificaciones del sistema alas HIS, se propone seguir las pautas de la metodología Proceso Unificado de Desarrollo (más conocida por RUP). Además, se sugiere el uso del Lenguaje Unificado de Modelado (UML, por sus siglas en inglés) como lenguaje de modelado.

1.5.1 Proceso Unificado de Desarrollo

RUP es una forma disciplinada de asignar tareas y responsabilidades en una empresa de desarrollo (quién hace qué, cuándo y cómo). Los procesos de RUP estiman tareas y horario del plan, midiendo la velocidad de iteraciones concerniente a sus estimaciones originales. Busca detectar defectos en las fases iniciales. Intenta reducir al número de cambios tanto como sea posible. Realiza el análisis y diseño, tan

completo como sea posible. Se caracteriza por ser guiado por casos de uso, centrado en la arquitectura e iterativo e incremental.

1.5.2 Lenguaje Unificado de Modelado

El Lenguaje Unificado de Modelado prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, describe la semántica esencial de lo que estos diagramas y símbolos significan. Se utiliza para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software.

Este lenguaje de modelado permite tener un mayor rigor en la especificación, realizar una verificación y validación del modelo desarrollado, automatizar determinados procesos, así como generar código a partir de los modelos y viceversa. (33)

1.6 Herramientas

1.6.1 Eclipse

Eclipse es un Entorno de Desarrollo Integrado (IDE, por sus siglas en inglés) de código abierto (su diseño permite una fácil extensión por parte de terceros) y multiplataforma. Proporciona herramientas para gestionar espacios de trabajo, construir, lanzar y depurar aplicaciones y compartir objetos con el equipo de desarrollo. Está construido sobre un mecanismo para descubrir, integrar y ejecutar módulos (en inglés plug-ins). (34) Esto lo diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos permite a Eclipse extenderse, usando otros lenguajes de programación como C/C++, Python y Java.

1.6.2 JBoss Tools

JBoss Tools es un conjunto de plug-ins de Eclipse que tienen como objetivo ayudar a los desarrolladores a crear aplicaciones web de forma rápida y sencilla. Los módulos de JBoss Tools son:

- **RichFaces VE:** Editor visual que incluye soporte para las librerías de componentes JSF incluyendo JBoss RichFaces. Brinda el apoyo para la edición visual de páginas HTML, JSF, JSP (Java Server Pages) y Facelets.

- **Seam Tools:** Incluye soporte para la vinculación de los componentes del framework Seam.

- **Hibernate Tools:** Soporta el mapeo de archivos, anotaciones y JPA con ingeniería inversa, completamiento de código, asistentes de proyecto, refactorización, ejecución interactiva de HQL/JPA-QL (JPA Query Language)/Criteria.
- **JBoss AS Tools:** Contiene funciones para el despliegue eficiente de cualquier tipo de proyecto en el IDE.

1.6.3 Sistemas de Gestión de Base de Datos.

1.6.3.1 PostgreSQL

Es un Sistema Gestor de Base de Datos (SGBD) relacional de código abierto, muy poderoso, con una arquitectura probada. Puede ser ejecutado sobre la mayoría de los sistemas operativos que existen en la actualidad. Posee características sofisticadas como fiabilidad, Control de Concurrencia Multi-Versión (MVCC, por sus siglas en inglés), replicación asíncrona, transacciones anidadas, realización de respaldo de datos en línea, optimizador o planificador de consultas y soporta internacionalización. Es altamente escalable en cuanto a la cantidad de información que puede manejar y al número de usuarios concurrentes que puede alojar. Presenta características que no tienen otros SGBD como son los tipos de datos definidos por el usuario, la herencia y el uso de normas. El desarrollo de PostgreSQL es realizado por un equipo de desarrolladores, en su mayoría voluntarios, extendido por todo el mundo, que se comunican vía Internet. Se trata de un proyecto comunitario y no está controlado por compañía alguna. (35) Se utiliza PostgreSQL en su versión 8.4 dadas las características y ventajas que brinda.

1.6.4 Visual Paradigm

Es una herramienta de Ingeniería de Software Asistida por Computadora (CASE, por sus siglas en inglés), que utiliza como lenguaje de modelado UML y que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Se integra con varios IDE como Eclipse, NetBeans, VisualStudio, entre otros. Provee un generador de mapeo de objetos relacionales para los lenguajes de programación Java, PHP y algunos lenguajes del ambiente .Net. (36) Se utiliza Visual Paradigm en su versión 6.4 dadas las características y ventajas que brinda.

1.6.5 iReport

iReport es una herramienta visual que sirve para generar ficheros XML (plantillas de informe), que se puedan utilizar con la herramienta de generación de informes JasperReports. Este diseñador de informes está escrito en Java y es gratuito. Permite a los usuarios la corrección visual de informes complejos. Soporta internacionalización. Maneja una gran parte de las etiquetas de JasperReports. Tiene asistentes para generar los sub-reportes. (37)

Conclusiones

Como resultado del estudio realizado en el presente capítulo, se puede concluir que los sistemas de notificaciones existentes, vinculados a las instituciones de salud, no cumplen con todas las funcionalidades deseadas, ni permiten su integración al sistema alas HIS. Por tal motivo, se evidencia la necesidad de elaborar un componente de notificaciones para el Sistema de Información Hospitalaria alas HIS que sea capaz de comunicar información en cortos períodos de tiempo a través del correo electrónico, hacia dispositivos de radio-búsqueda o hacia teléfonos celulares y que además, se encuentre debidamente documentado para facilitar su comprensión durante el desarrollo de versiones futuras. Se llevó a cabo el análisis de las tecnologías y herramientas que serán utilizadas para la construcción del componente.

CAPÍTULO 2 CARACTERÍSTICAS DEL COMPONENTE DE NOTIFICACIONES

En este capítulo, con motivo de la poca estructuración de los procesos del negocio y para poder comprender el contexto en el cual se desarrolla el sistema, se muestra el Modelo de Dominio, donde se exponen conceptos relacionados con el sistema y los vínculos entre estas definiciones. Por otra parte, se enumeran los requerimientos funcionales y no funcionales, agrupándose los primeros en casos de uso, con el objetivo de estructurar el Diagrama de Casos de Uso del Sistema.

2.1 Modelo de Dominio

El Modelo de Dominio o Modelo Conceptual es una representación visual de los conceptos u objetos que se manejan en el dominio del sistema. Los objetos o conceptos incluidos en el Modelo de Dominio no describen clases u objetos del software; sino entidades o conceptos del mundo real que están asociadas al problema en cuestión. Dicho modelo podrá ser utilizado como una base de las abstracciones relevantes en el proceso de construcción del sistema.

2.1.2. Conceptos fundamentales del dominio

A continuación se proporcionará una breve descripción de los conceptos encontrados en el ámbito del problema, con el fin de facilitar una mejor comprensión del Diagrama del Modelo de Dominio, recordando que dichos conceptos estarán, en gran escala, asociados a los conceptos existentes en mecanismos o componentes de notificaciones actuales. Estos son:

Usuario: Persona que por medio de un ordenador puede acceder a los recursos y servicios que ofrece el sistema, de acuerdo con los privilegios, permisos y roles asignados.

Administrador: Usuario del sistema encargado de realizar la configuración general del mismo.

Notificación: Contenido a transmitir entre un emisor (médico, almacenero, farmacéutico y demás usuarios del sistema) y un receptor (paciente, médico, farmacéutico, entre otros) en una situación comunicacional.

Mensaje a correo: Contenido a transmitir entre un emisor y un receptor a través del correo electrónico.

Mensaje a beeper: Contenido a transmitir por un emisor, desde el sistema hasta el beeper de un receptor.

Mensaje a teléfono móvil: Contenido a transmitir por un emisor, desde el sistema hasta el teléfono móvil de un receptor.

Adjunto: Archivo o fichero que va o está unido al cuerpo o contenido del mensaje de correo electrónico.

Traza: Registro oficial de eventos durante un rango de tiempo en particular, que pueden ser recolectados y analizados con herramientas y técnicas especiales, lo que deja al descubierto la actividad registrada en el mismo.

2.1.3. Diagrama del Modelo de Dominio

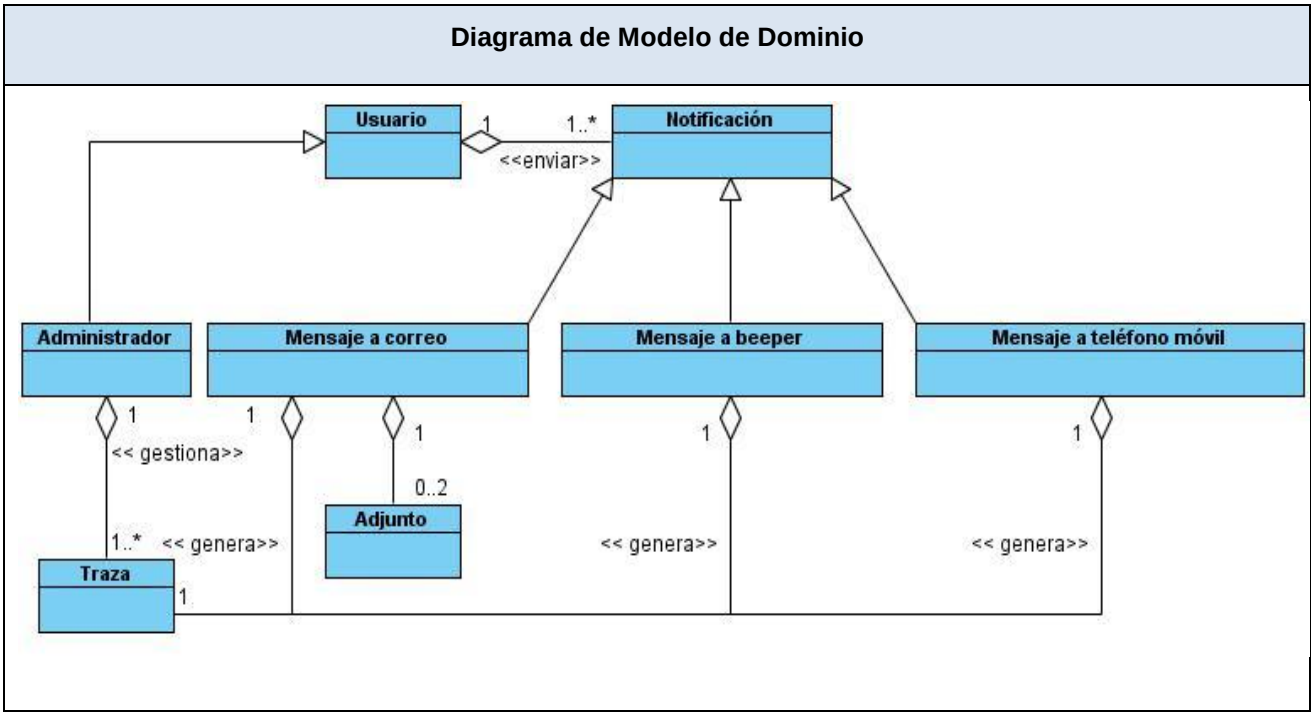


Figura 2.1 Diagrama de Modelo de Dominio

Como se evidencia en el Diagrama de Modelo de Dominio, un usuario del sistema tiene la posibilidad de enviar una o varias notificaciones, que pueden ser de tipo mensaje a beeper, mensaje a teléfono móvil o mensaje a correo y este último brinda la opción de adjuntar hasta dos ficheros. Estas tres vías para notificar generan un registro oficial de eventos, el cual es gestionado por el administrador del sistema, que a su vez es un usuario del mismo.

2.2 Especificación de los requerimientos del software

2.2.1 Requisitos funcionales del sistema

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir y que definen el comportamiento interno del software. Estos describen los servicios que se espera que el sistema cumpla para satisfacer las necesidades del usuario.

El sistema deberá permitir:

RF1. Listar entidades.

RF2. Seleccionar entidad.

RF3. Ver configuración de parámetros de conexión.

RF4. Adicionar configuración para correo electrónico.

RF5. Modificar configuración para correo electrónico.

RF6. Eliminar configuración para correo electrónico.

RF7. Adicionar configuración para mensaje a beeper.

RF8. Modificar configuración para mensaje a beeper.

RF9. Eliminar configuración para mensaje a beeper.

RF10. Adicionar configuración para mensaje a teléfono móvil.

RF11. Modificar configuración para mensaje a teléfono móvil.

RF12. Eliminar configuración para mensaje a teléfono móvil.

RF13. Listar usuarios del sistema.

RF14. Ver números de beeper de usuarios.

RF15. Adicionar número de beeper a usuario.

RF16. Modificar número de beeper asignado a usuario.

RF17. Eliminar número de beeper asignado a usuario.

RF18. Buscar usuario del sistema.

RF19. Seleccionar usuario del sistema.

RF20. Buscar rol del sistema.

RF21. Seleccionar rol del sistema.

RF22. Buscar paciente del sistema.

RF23. Seleccionar paciente del sistema.

RF24. Enviar mensaje a correo electrónico.

RF25. Enviar mensaje a beeper.

RF26. Enviar mensaje a teléfono móvil.

RF27. Buscar módulo del sistema.

RF28. Seleccionar módulo del sistema.

RF29. Generar reporte de trazas de correo electrónico.

RF30. Generar reporte de trazas de mensaje a beeper.

RF31. Generar reporte de trazas de mensaje a teléfono móvil.

2.2.2 Requisitos no funcionales del sistema

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Son características que hacen al producto atractivo, usable, rápido o confiable. Estos requerimientos se agrupan en varias categorías:

2.2.2.1 Requisitos de Usabilidad

El componente debe brindar, a través de una barra de navegación, un acceso fácil y rápido a todas las funcionalidades. La información que se muestra al usuario no debe ser redundante, con dobles sentidos y debe mantener una estructura lógica. El significado de los íconos y los textos debe ser claro para la persona que interactúe con la aplicación. Las opciones que se proveen deben ser comprensibles, sin explicaciones excesivas sobre su uso, y deben admitir flujos alternativos, como cancelar la operación.

2.2.2.2 Requisitos de Fiabilidad

El componente debe estar disponible de forma permanente y funcionar sin necesidad de intervención del usuario.

2.2.2.3 Requisitos de Seguridad

El componente debe mantener 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. Las contraseñas podrán cambiarse únicamente por el propio usuario o por el administrador del sistema. Los objetos introducidos al sistema no serán eliminados físicamente de la base de datos. Toda entrada de información estará validada, de forma que la introducción de datos incorrectos se mostrará al usuario especificándole el tipo de error. Se archivarán trazas con toda la información de las notificaciones enviadas.

2.2.2.4 Requisitos de Rendimiento

El componente debe permitir el envío de múltiples mensajes SMS y correos electrónicos de manera asincrónica, garantizando un mejor aprovechamiento de los recursos. Cada uno de los procesos que se ejecuten serán lo suficientemente óptimos en cuanto al uso de memoria física y necesidad de procesamiento.

2.2.2.5 Requisitos de Hardware

Los requerimientos de hardware estarán dados por la plataforma específica que se utilice para la instalación del componente, en cuanto a sistema operativo, servidor de aplicaciones y gestor de bases de datos.

2.2.2.5.1 Estaciones de trabajo

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

2.2.2.5.2 Servidores

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

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

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

2.2.2.6 Requisitos de Software

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

2.2.2.7 Requisitos de Interfaz de usuario

Las interfaces creadas como parte del componente contendrán los datos legibles y bien estructurados, además de permitir la interpretación correcta de la información. La entrada de datos incorrecta será detectada claramente y se mostrará al usuario.

2.2.2.8 Requisitos de portabilidad

La aplicación podrá ser desplegada sobre Sistemas Operativos Linux (Oracle Linux 5.5+, 6.x, Ubuntu 8.04 LTS (Long Term Support) Desktop Edition, Ubuntu Linux 10.04 y superior, entre otras versiones) y Windows (2000, XP, Server 2003, Server 2008, Server 2012, Vista, 7, 8).

2.3 Modelo de Casos de Uso del Sistema

2.3.1 Definición de los actores del sistema

Los actores de un sistema son agentes externos, roles que las personas (usuarios) o dispositivos juegan cuando interactúan con el software.

Actor	Descripción
Usuario	Usuario global que se autentica en la aplicación, el sistema lo valida y le asigna un rol con la posibilidad de enviar notificaciones.
Administrador	Utiliza el componente de notificaciones, realiza la configuración de los parámetros de conexión para cada tipo de notificación y da seguimiento a las trazas que se derivan de los mensajes enviados.

Tabla 2.1 Definición de los actores del sistema

2.3.2 Vista global de actores del sistema

Vista global de actores del sistema


Figura 2.2 Vista global de actores del sistema

2.3.3 Diagrama de Casos de Uso del Sistema

Los Diagramas de Casos de Uso permiten que los desarrolladores y los clientes lleguen a un entendimiento sobre las condiciones y el alcance del sistema antes de emprender la fase de construcción del mismo.

Diagrama de Casos de Uso del Sistema

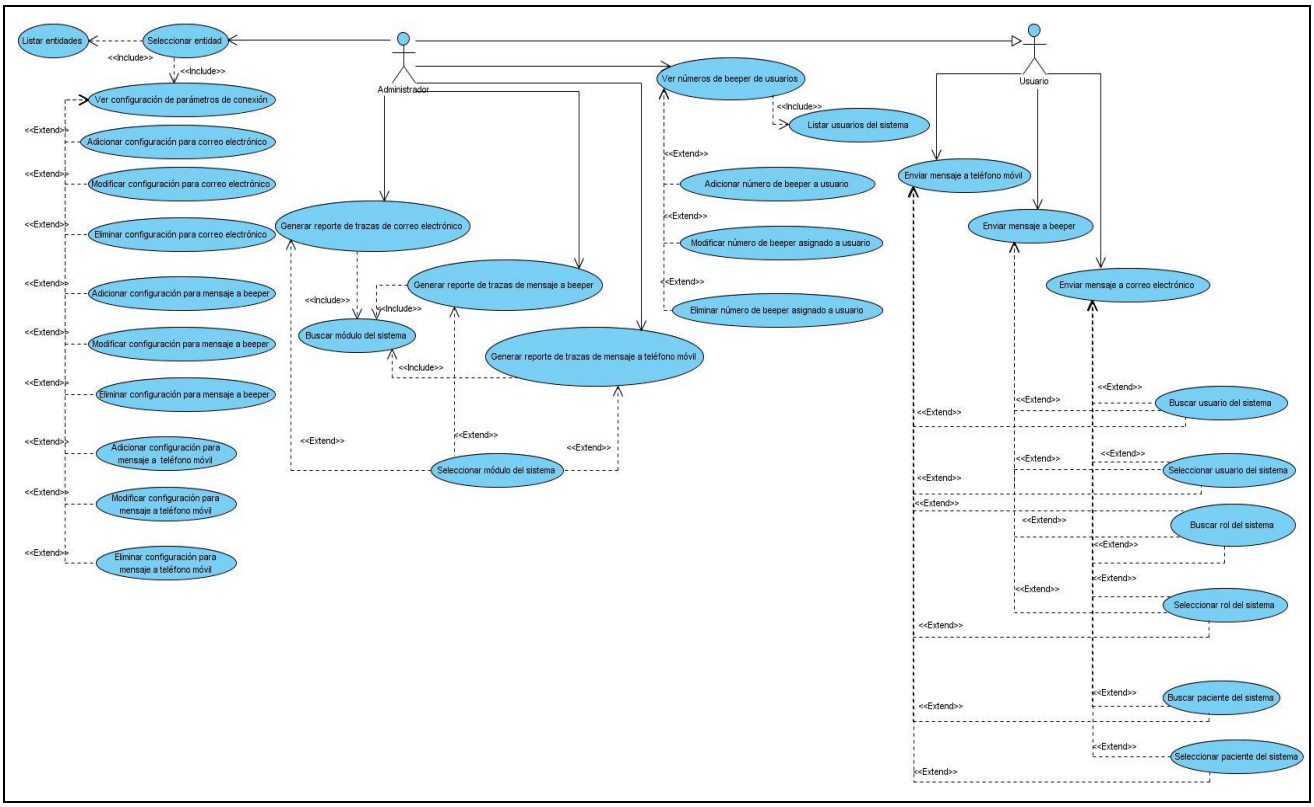


Figura 2.3 Diagrama de Casos de Uso del Sistema

El componente de notificaciones cuenta con tres partes fundamentales: configurar los aspectos necesarios para el funcionamiento del mismo, enviar mensajes y generar reportes de trazas de envíos realizados.

El administrador puede configurar los parámetros de conexión de correo electrónico, beeper y teléfono móvil, para cada entidad del sistema. Esta funcionalidad cuenta con las variantes de adicionar, modificar y eliminar. Además puede adicionar, modificar y eliminar los números de beepers de los usuarios, desde un listado de los usuarios del sistema.

Un usuario tiene la posibilidad de enviar notificaciones hacia correos electrónicos, beepers y teléfonos celulares, las cuales son destinadas a direcciones específicas o a roles y usuarios del sistema, los que pueden ser seleccionados a partir de una búsqueda. Los correos y mensajes SMS también se pueden remitir a los pacientes, que de igual forma se seleccionan después de realizada una búsqueda.

El administrador es el encargado de generar los reportes de trazas de envíos realizados. En dicha acción, selecciona los criterios que desea para la elaboración del reporte.

2.3.4 Descripción textual de los casos de uso

A continuación se describen algunos de los casos de uso que forman parte del diagrama anterior. La descripción del resto se encuentra en el artefacto “Modelo de Casos de Uso del Sistema”.

Caso de uso	
CU-1	**Ver configuración de parámetros de conexión**
Propósito	Permitir al actor ver la configuración de los parámetros de conexión para el envío de cada tipo de notificación.
Actores	Administrador
Resumen:	El caso de uso inicia luego de que el actor selecciona una entidad del sistema (ver **Caso de Uso:** Seleccionar entidad). El sistema muestra los datos de las configuraciones de conexión para el envío de correos electrónicos y mensajes a beepers y a teléfonos móviles, según la entidad seleccionada. El caso de uso termina.
Precondiciones	Se debe haber seleccionado una entidad del sistema.

Referencias	RF2. Seleccionar entidad. RF3. Ver configuración de parámetros de conexión. RF4. Adicionar configuración para correo electrónico. RF5. Modificar configuración para correo electrónico. RF6. Eliminar configuración para correo electrónico. RF7. Adicionar configuración para beeper. RF8. Modificar configuración para beeper. RF9. Eliminar configuración para beeper. RF10. Adicionar configuración para mensaje a teléfono móvil. RF11. Modificar configuración para mensaje a teléfono móvil. RF12. Eliminar configuración para mensaje a teléfono móvil.

Flujo Normal de los Eventos	
Acción del actor	**Respuesta del sistema**
	1. Muestra los datos de las conexiones para correos, mensajes a beeper y mensajes SMS de la entidad seleccionada. • En caso de que existan datos de conexión para correo, el sistema permite realizar las funcionalidades de los casos de uso: **Modificar configuración para correo electrónico, Eliminar configuración para correo electrónico.** De lo contrario, se presenta el mensaje “No existe información para mostrar” y se permite realizar las funcionalidades del caso de uso: **Adicionar configuración para correo electrónico.**

	• En caso de que existan datos de conexión para mensajes a beeper, el sistema permite realizar las funcionalidades de los casos de uso: **Modificar configuración para mensaje a beeper, Eliminar configuración para mensaje a beeper.** De lo contrario, se presenta el mensaje "No existe información para mostrar" y se permite realizar las funcionalidades del caso de uso: **Adicionar configuración para mensaje a beeper.** • En caso de que existan datos de conexión para mensajes a teléfono móvil, el sistema permite realizar las funcionalidades de los casos de uso: **Modificar configuración para mensaje a teléfono móvil, Eliminar configuración para mensaje a teléfono móvil.** De lo contrario, se presenta el mensaje "No existe información para mostrar" y se permite realizar las funcionalidades del caso de uso: **Adicionar configuración para mensaje a teléfono móvil.** El sistema brinda además la opción "Salir". Ver **Alternativa 1**: "Salir".
	2. El caso de uso termina.
Flujos alternos	
Alternativa 1. **"Salir"**	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción "Salir".	2. Regresa a la pantalla de selección de módulos.
	3. El caso de uso termina.

Poscondiciones	Se vieron los datos de conexión para correos electrónicos, mensajes a beeper y mensajes a teléfono celular de la entidad seleccionada.

Tabla 2.2 Descripción textual del caso de uso: *Ver configuración de parámetros de conexión*

Caso de uso	
CU-2	**Adicionar configuración para correo electrónico**
Propósito	Permitir al actor configurar los parámetros de conexión para el envío de correo electrónico.
Actores	Administrador
Resumen:	El caso de uso inicia cuando el actor accede a la opción “Adicionar”. El sistema muestra los campos a llenar. El actor entra los nuevos datos y acepta. El sistema valida la información y si no existen errores crea la nueva configuración de conexión para correo, regresa a la vista anterior y el caso de uso termina.
Precondiciones	No existe una configuración para correo electrónico registrada.
Referencias	RF2. Seleccionar entidad. RF3. Ver configuración de parámetros de conexión. RF4. Adicionar configuración para correo electrónico.
Flujo Normal de los Eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso inicia cuando el actor accede a la opción “Adicionar”.	2. Muestra una ventana con los campos a llenar para crear una nueva configuración: • Usuario • Contraseña • Confirmar contraseña

	• Servidor SMTP • Puerto • Capacidad de adjuntos El sistema brinda las opciones “Aceptar”, “Cancelar” (ver **Alternativa 1**: “Cancelar”) y “Cerrar” (ver **Alternativa 2**: “Cerrar”)
3. Introduce los nuevos datos y selecciona la opción “Aceptar”.	4. Valida los datos introducidos. Si existen datos incompletos o incorrectos, ver **Alternativa 3:** “Datos incorrectos o incompletos”.
	5. Adiciona la configuración de conexión para correo electrónico.
	6. Regresa a la vista anterior.
	7. El caso de uso termina.
Flujos alternos	
Alternativa 1. **“Cancelar”**	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción “Cancelar”.	2. Regresa a la vista anterior.
	3. El caso de uso termina.
Alternativa 2. **“Cerrar”**	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción “Cerrar”.	2. Regresa a la vista anterior.

	3. El caso de uso termina.
Alternativa 3. **"Datos incorrectos o incompletos"**	
Acción del actor	**Respuesta del sistema**
	1. Muestra un indicador sobre los campos incorrectos o incompletos.
	2. Regresa al paso 2 del **Flujo Normal de Eventos.**
Poscondiciones	Se adicionó una configuración de conexión para el envío de correo electrónico.

Tabla 2.3 Descripción textual del caso de uso: *Adicionar configuración para correo electrónico*

Caso de uso	
CU-3	**Enviar mensaje a correo electrónico**
Propósito	Permitir al actor el envío de correo electrónico.
Actores	Usuario
Resumen:	El caso de uso inicia cuando el actor accede a la opción "Enviar mensaje a correo electrónico". El sistema muestra las opciones de envío por direcciones, por roles, por usuarios y por pacientes. Se muestran además los campos necesarios para enviar el correo electrónico. El actor entra los datos y acepta. El sistema valida la información y si no existen errores envía el correo y deja una traza. El caso de uso termina.
Precondiciones	Haber configurado los parámetros de conexión para correo electrónico.

<table>
<tr><td>Referencias</td><td colspan="2">RF18. Buscar usuario del sistema.
RF19. Seleccionar usuario del sistema.
RF20. Buscar rol del sistema.
RF21. Seleccionar rol del sistema.
RF22. Buscar paciente del sistema.
RF23. Seleccionar paciente del sistema.
RF24. Enviar mensaje a correo electrónico.</td></tr>
<tr><td colspan="3">Flujo Normal de los Eventos</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td colspan="2">1. El caso de uso inicia cuando el actor accede a la opción “Enviar mensaje a correo electrónico”.</td><td>2. Muestra cuatro opciones de envío, seleccionando por defecto “Por direcciones”:
• Por direcciones. (Ver Sección 1: “Por direcciones”).
• Por roles. (Ver Sección 2: “Por roles”).
• Por usuarios. (Ver Sección 3: “Por usuarios”).
• Por pacientes. (Ver Sección 4: “Por pacientes”).

El sistema muestra los campos que son comunes para cualquier envío:
• Asunto
• Texto
• Adjunto 1</td></tr>
</table>

	El sistema brinda además las opciones “Aceptar”, “Cancelar” (ver **Alternativa 1**: “Cancelar”) y permite subir un archivo adjunto (ver **Alternativa 2**: “Subir adjunto 1”) y adicionar más adjuntos (ver **Alternativa 3**: “Más adjuntos”).
SECCIONES	
Sección 1: “Por direcciones”	
Acción del actor	**Respuesta del sistema**
	1. El sistema muestra también el campo: • Direcciones de correo
2. Introduce los datos necesarios para enviar el correo, el asunto es opcional.	3. Valida los datos introducidos. Si están incompletos o incorrectos, ver **Alternativa 4:** “Datos incorrectos o incompletos”.
	4. Envía el correo electrónico a la lista de direcciones introducida.
	5. Archiva la traza del envío realizado.
	6. Limpia los campos.
	7. El caso de uso termina.
Sección 2: “Por roles”	
Acción del actor	**Respuesta del sistema**
	1. Busca los roles del sistema y muestra un listado de los mismos. (Ver **Caso de Uso**: Buscar rol del sistema).

2. Selecciona los roles a los cuales se dirige el correo (ver **Caso de Uso:** Seleccionar rol del sistema) e introduce los datos necesarios para enviar el mensaje, el asunto es opcional.	3. Valida los datos introducidos. Si están incompletos o incorrectos, ver **Alternativa 4:** "Datos incorrectos o incompletos".
	4. Envía el correo electrónico a los roles seleccionados.
	5. Archiva la traza del envío realizado.
	6. Limpia los campos.
	7. El caso de uso termina.
Sección 3: "Por usuarios"	
Acción del actor	**Respuesta del sistema**
	1. Permite realizar una búsqueda de los usuarios del sistema y muestra una lista de los mismos. (Ver **Caso de Uso:** Buscar usuarios del sistema).
2. Selecciona los usuarios a los cuales se dirige el correo (ver **Caso de Uso**: Seleccionar usuario del sistema) e introduce los datos necesarios para enviar el mensaje, el asunto es opcional.	3. Valida los datos introducidos. Si están incompletos o incorrectos, ver **Alternativa 4:** "Datos incorrectos o incompletos".
	4. Envía el correo electrónico a los usuarios seleccionados.
	5. Archiva la traza del envío realizado.
	6. Limpia los campos.
	7. El caso de uso termina.
Sección 4: "Por pacientes"	

Acción del actor	**Respuesta del sistema**
	1. Permite realizar una búsqueda de los pacientes del sistema y muestra una lista de los mismos. (Ver **Caso de Uso:** Buscar pacientes del sistema).
2. Selecciona los pacientes a los cuales se dirige el correo (ver **Caso de Uso**: Seleccionar paciente del sistema) e introduce los datos necesarios para enviar el mensaje, el asunto es opcional.	3. Valida los datos introducidos. Si están incompletos o incorrectos, ver **Alternativa 4:** "Datos incorrectos o incompletos".
	4. Envía el correo electrónico a los pacientes seleccionados.
	5. Archiva la traza del envío realizado.
	6. Limpia los campos.
	7. El caso de uso termina.
Flujos alternos	
Alternativa 1. **"Cancelar"**	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción "Cancelar".	2. Regresa a la pantalla de bienvenida del módulo.
	3. El caso de uso termina.
Alternativa 2. **"Subir adjunto 1"**	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción "Browse" (Buscar).	2. Muestra una ventana que permite la selección de un fichero a adjuntar.

3. Selecciona un fichero y acepta.	4. Guarda la dirección y los datos del fichero seleccionado.
Alternativa 3. **"Más adjuntos"**	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción "Adicionar adjunto".	2. El sistema muestra el campo: • Adjunto 2 Da la posibilidad al actor de subir otro fichero adjunto (de forma similar a la **Alternativa 2:** "Subir adjunto 1").
Poscondiciones	Se envió un correo electrónico hacia las direcciones especificadas o hacia usuarios, roles o pacientes registrados en el sistema.

Tabla 2.4 Descripción textual del caso de uso: *Enviar mensaje a correo electrónico*

Caso de uso	
CU-4	**Generar reporte de trazas de correo electrónico**
Propósito	Permitir al actor generar reportes de trazas de correos electrónicos enviados.
Actores	Administrador
Resumen:	El caso de uso inicia cuando el actor accede a la opción "Generar reporte de trazas de correo electrónico". El sistema muestra los parámetros que se necesitan para realizar un reporte de este tipo. El actor entra los datos y selecciona Generar. El sistema valida la información y si no existen errores genera el reporte, dando la posibilidad de exportarlo. El caso de uso termina.
Precondiciones	No existen.

<table>
<tr><td>Referencias</td><td colspan="2">RF27. Buscar módulo del sistema.
RF28. Seleccionar módulo del sistema.
RF29. Generar reporte de trazas de correo electrónico.</td></tr>
<tr><td colspan="3">Flujo Normal de los Eventos</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td colspan="2">1. El caso de uso inicia cuando el actor accede a la opción accede a la opción “Generar reporte de trazas de correo electrónico”.</td><td>2. Muestra los parámetros que se necesitan para realizar un reporte de este tipo:
Desde (fecha inicio)
Hasta (fecha fin)
Archivos adjuntos
El sistema además busca los módulos físicos (ver Caso de Uso: Buscar módulo del sistema) y los presenta en un listado y brinda las opciones “Generar” y “Cancelar” (ver Alternativa 1: “Cancelar”).</td></tr>
<tr><td colspan="2">3. Selecciona la fecha de inicio, la fecha de fin y los módulos para los que se desea que se haga el reporte. Selecciona la opción “Generar”.</td><td>4. Valida los datos introducidos. Si están incompletos o incorrectos, ver Alternativa 2: “Datos incorrectos o incompletos”.</td></tr>
<tr><td colspan="2"></td><td>5. Genera el reporte de trazas de correos enviados, lo muestra en pantalla y permite seleccionar la opción “Exportar” (ver Alternativa 3: “Exportar”).</td></tr>
<tr><td colspan="2"></td><td>6. El caso de uso termina.</td></tr>
<tr><td colspan="3">Flujos alternos</td></tr>
<tr><td colspan="3">Alternativa 1. “Cancelar”</td></tr>
</table>

Acción del actor	**Respuesta del sistema**
1. Selecciona la opción “Cancelar”.	2. Regresa a la pantalla de bienvenida del módulo.
	3. El caso de uso termina.
Alternativa 2. **“Datos incorrectos o incompletos”**	
Acción del actor	**Respuesta del sistema**
	1. Muestra un indicador sobre los campos incorrectos o incompletos.
Alternativa 3. **“Exportar”**	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción “Exportar”.	2. Muestra una ventana con los formatos a exportar. Brinda las opciones “Aceptar”, “Cancelar” (ver **Alternativa 4**: “Cancelar exportar”) y “Cerrar” (ver **Alternativa 5**: “Cerrar”).
3. Selecciona un formato y luego accede a la opción “Aceptar”.	4. Valida que el campo no esté incompleto. Si el campo está incompleto (ver **Alternativa 6**: “Formato incompleto”).
	5. Exporta el reporte en un fichero con el formato seleccionado.
	6. El caso de uso termina.
Alternativa 4. **“Cancelar exportar”**	
Acción del actor	**Respuesta del sistema**

1. Selecciona la opción “Cancelar”.	2. Regresa a la vista anterior.
Alternativa 5. **“Cerrar”**	
Acción del actor	**Respuesta del sistema**
1. Selecciona la opción “Cerrar”.	2. Regresa a la vista anterior.
Alternativa 6. **“Formato incompleto”**	
Acción del actor	**Respuesta del sistema**
	1. Muestra un indicador sobre el campo incompleto.
Poscondiciones	Se generó un reporte de trazas de correos enviados.

Tabla 2.5 Descripción textual del caso de uso: *Generar reporte de trazas de correo electrónico*

Conclusiones

En este capítulo se obtuvo el Modelo de Dominio, donde se abarcaron las definiciones asociadas al sistema y las relaciones definidas entre ellas, lo que posibilita un mejor entendimiento del problema y una rápida identificación de las funcionalidades y propiedades con las que debe contar el componente. Se enunciaron los requisitos funcionales y no funcionales y se definió el Modelo de Casos de Uso del Sistema con su diagrama y especificaciones, con el fin de lograr el desarrollo de una aplicación que responda satisfactoriamente a la situación problémica planteada.

CAPÍTULO 3 ANÁLISIS Y DISEÑO DEL COMPONENTE DE NOTIFICACIONES

En el presente capítulo se realiza una descripción detallada del sistema propuesto y se explica la arquitectura que presenta el mismo. Se presentan los principales patrones de diseño que se ponen en práctica y los Diagramas de Clases del Diseño e Interacción que forman parte del Modelo de Diseño. Se ofrece una breve descripción de las clases u operaciones necesarias para el funcionamiento del componente de notificaciones.

3.1 Descripción de la arquitectura, fundamentación

La arquitectura de software, también denominada arquitectura lógica, es un conjunto de patrones y abstracciones coherentes que proporcionan un marco de referencia necesario para guiar la construcción de un software dentro de un sistema informático. Es considerada el diseño de mayor nivel de la estructura de un sistema. Permite a los programadores, diseñadores, ingenieros y analistas trabajar bajo una línea común que les posibilite la compatibilidad necesaria para lograr el objetivo deseado.

Para el desarrollo del sistema y teniendo en cuenta las herramientas, tecnologías y metodologías propuestas, se define como parte de la línea base de la arquitectura, la implementación del patrón de diseño de arquitectura Modelo Vista Controlador (MVC). Este patrón permite la separación de los datos de una aplicación, la interfaz de usuario y la lógica de control, en tres componentes distintos: el modelo, para la administración de los datos; la vista, que muestra la información del modelo al usuario y el controlador, que gestiona las entradas del usuario.

Con este patrón se logra realizar un diseño que desacople la vista del modelo y permita la reusabilidad de los componentes. De esta manera, las modificaciones en las vistas impactan, en menor medida, en la lógica de negocio o de datos. Brinda mejor organización según la función que realizan, permitiendo que en un momento determinado un elemento de una capa pueda ser modificado o sustituido completamente causando el mínimo de alteraciones en otro elemento que lo utilice.

La capa de presentación está conformada principalmente por páginas XHTML. Estas están compuestas por formularios que mediante controles JSF, Seam UI, Facelets y RichFaces obtienen y validan los datos que el usuario provee en cada una de las operaciones que realiza. El uso de estos componentes enriquece el diseño de la interfaz de usuario. Además, al realizar el envío y la carga de datos mediante los componentes Ajax4sf, se logra un efecto más agradable y natural al interactuar con el sistema.

La capa de negocio está constituida por clases controladoras que se encargan de definir la lógica del negocio del módulo, así como del manejo y validación de los datos capturados en la capa de presentación.

Por último, la capa de acceso a datos tiene la principal función de cargar, modificar, eliminar y persistir la información existente en la base de datos una vez que sea validada. Este proceso se logra gracias al uso de componentes Hibernate por los que se encuentra constituida dicha capa. Estos componentes logran abstraer al desarrollador, del gestor de base de datos utilizado, mediante el mapeo de tablas. Esto da la posibilidad de llevar las consultas a un lenguaje de objetos.

3.2 Estrategias de integración

El componente de notificaciones no responde a un área específica de las instituciones hospitalarias como puede ser hospitalización, consulta externa, anatomía patológica o admisión. Este constituye la vía por la cual se emiten, en el menor tiempo posible, avisos que se pueden generar en cada una de estas áreas. Como fue diseñado para satisfacer las necesidades de notificación en tiempo real existentes en el Sistema alas HIS, puede ser integrado a cualquier módulo de dicha aplicación.

Con el objetivo de lograr uniformidad con respecto a los demás componentes y módulos del Sistema de Información Hospitalaria alas HIS y mejorar la calidad del trabajo, se reutiliza la interfaz IActiveModule con el objetivo de conocer en qué módulo y entidad se encuentra el usuario que está utilizando el sistema, de igual manera se hace uso de la clase Usuario para conocer qué usuario se encuentra trabajando con la aplicación en un determinado momento y obtener los datos del mismo. Además, los reportes de trazas de envíos realizados son generados empleando las funcionalidades de la clase ReportManager.

3.3 Modelo de Diseño

El Modelo de Diseño es un modelo de objetos que describe la realización de los casos de uso, es abarcador y está compuesto por artefactos que engloban todas las clases del diseño, subsistemas, paquetes, colaboraciones, y las relaciones entre ellos. Se centra principalmente en cómo los requisitos funcionales y no funcionales, junto a otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Es usado como una entrada inicial en las actividades de implementación y prueba.

Por lo general, durante la elaboración del diseño se utilizan patrones. Estos expresan esquemas para definir relaciones entre clases y objetos, con las que construir sistemas software. Describen una estructura de diseño que resuelve un problema en caso de presentarse, lo que puede tener un impacto en la forma en la que se aplica y utiliza el patrón. Esto permite al diseñador decidir si es necesario o no emplear un patrón específico.

Para llevar a cabo la definición del diseño de la solución propuesta se tuvo en cuenta una serie de patrones, entre los que se encuentran los Patrones de Software para la Asignación General de Responsabilidad (GRASP, por sus siglas en inglés). Se utilizaron con el objetivo de asignarle a las clases las tareas que podían realizar según la información que poseían, además de crear las instancias de otras clases en correspondencia con la responsabilidad dada, poniéndose de manifiesto los patrones Experto y Creador.

Otro de los utilizados es el de Alta cohesión, que garantiza que la información contenida en las clases sea coherente y que, en la medida de lo posible, esté relacionada con las mismas.

Se hace uso del Bajo acoplamiento con el fin de tener las clases lo menos ligadas posibles, de tal forma que, en caso de producirse una modificación en alguna de ellas, tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización, y disminuyendo la dependencia entre estas.

Por otra parte, en esta fase se elaboran los Diagramas de Clases del Diseño, donde se presenta un conjunto de interfaces, controladores y sus relaciones. Estos son de gran importancia, puesto que permiten visualizar, especificar y documentar modelos estructurales.

Se tuvieron en cuenta además los Diagramas de Interacción, los cuales son considerados artefactos que gráficamente muestran las relaciones entre objetos, incluyendo los mensajes que se pueden enviar entre ellos. Se utilizan para visualizar, especificar, construir y documentar la dinámica de una sociedad particular de objetos, o para modelar un flujo de control particular de un caso de uso.

Dentro de estos diagramas se definen los Diagramas de Secuencia y los Diagramas de Colaboración, donde generalmente se recomienda utilizar los de colaboración en el análisis y los de secuencia en el diseño. Los últimos destacan la ordenación temporal de los mensajes.

En la confección del diseño se define una estructura de paquetes que permite dividir el sistema en fragmentos manejables para su implementación. Se emplea además el criterio de empaquetamiento por

procesos y por clases, siguiendo la estructura de procesos definidos en el sistema. Los paquetes que hacen referencia a procesos se componen por subcarpetas que responden a las realizaciones de los casos de uso y los cuales contienen un Diagrama de Clases y los respectivos Diagramas de Secuencia.

Se crea un paquete Repositorio de clases que contiene tres subpaquetes: el de las Vistas, el de las Sesiones y el paquete de las Entidades.

El paquete de las Vistas incluye los contenidos web referentes a las páginas clientes y los formularios que la componen.

En el de Sesiones se agrupan las clases controladoras autogeneradas por el entorno de desarrollo y las clases controladoras personalizadas.

El subpaquete de las Entidades contiene las Entidades Autogeneradas y las Personalizadas. Las Entidades Autogeneradas son obtenidas mediante el mapeo de la base de datos. Las Entidades Personalizadas son las modificadas, o entidades que heredan de las autogeneradas y son cambiadas para lograr un mejor funcionamiento.

3.3.1 Diagrama de Paquetes

Diagrama de Paquetes

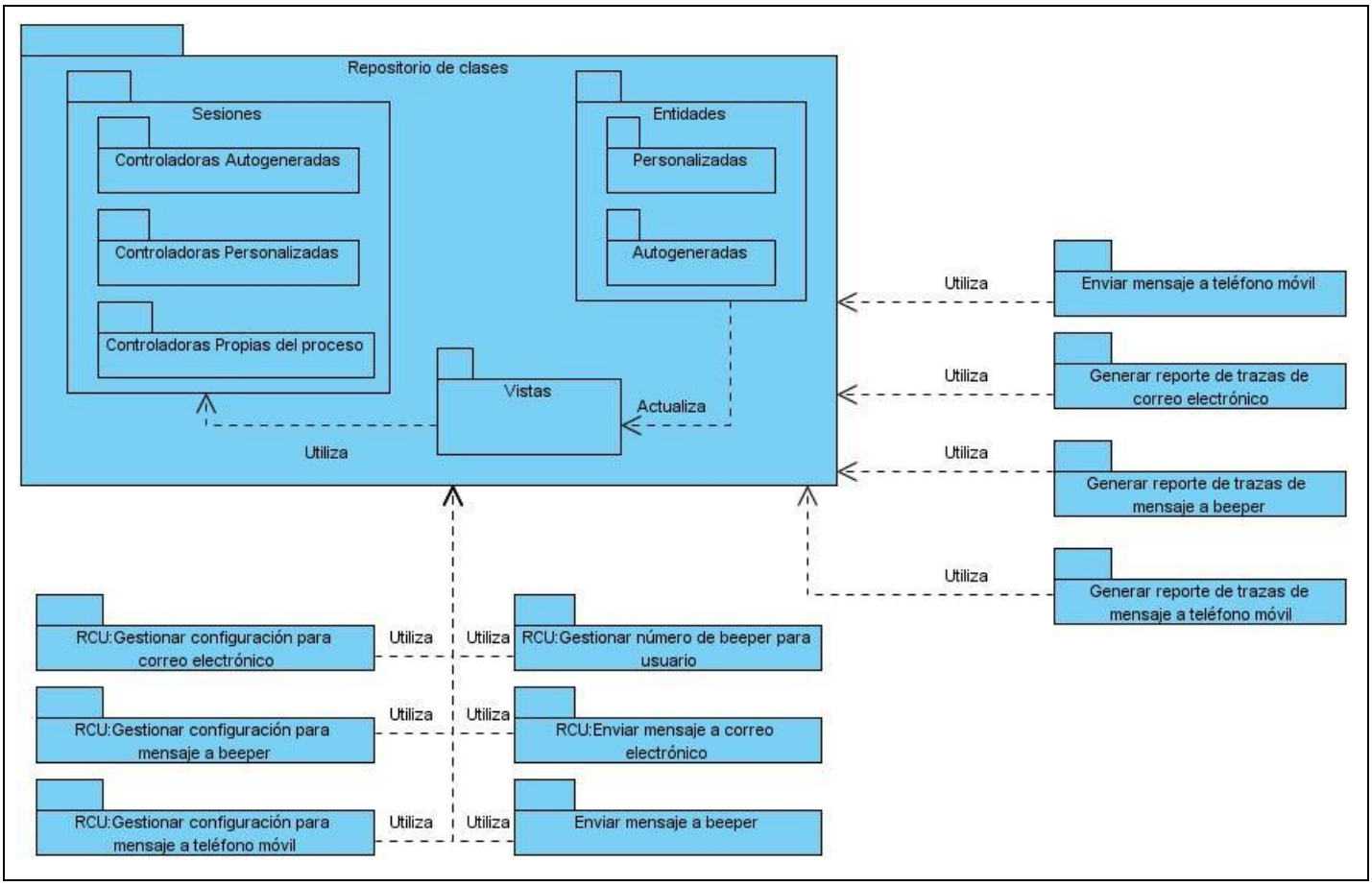


Figura 3.1 Diagrama de Paquetes

3.3.2 Diagramas de Clases del Diseño

Los Diagramas de Clases del Diseño se realizan mediante estereotipos web que son una representación gráfica de los componentes a los cuales hacen referencia. En estos diagramas se reflejan las relaciones entre páginas y clases. En general, muestran el flujo de cómo una página servidora construye una página cliente, la cual contiene un formulario que puede actualizar directamente a las entidades o enviar las peticiones a la página servidora, estás últimas invocan métodos y responsabilidades de las clases controladoras, las cuales pueden consultar o modificar las entidades.

DCD_ Ver configuración de parámetros de conexión

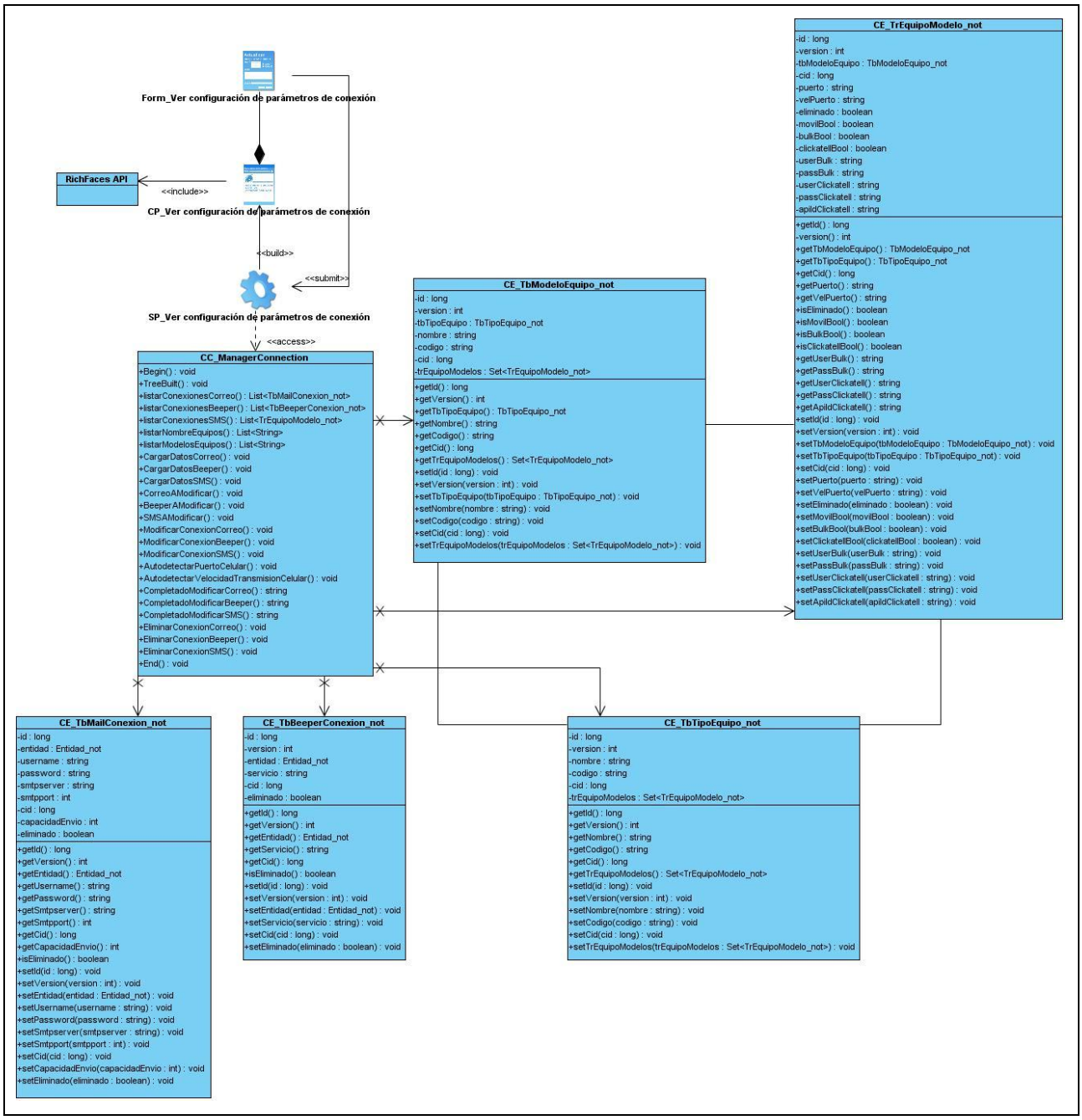


Figura 3.2 Diagrama de Clases del Diseño: *Ver configuración de parámetros de conexión*

DCD_Enviar mensaje a correo electrónico

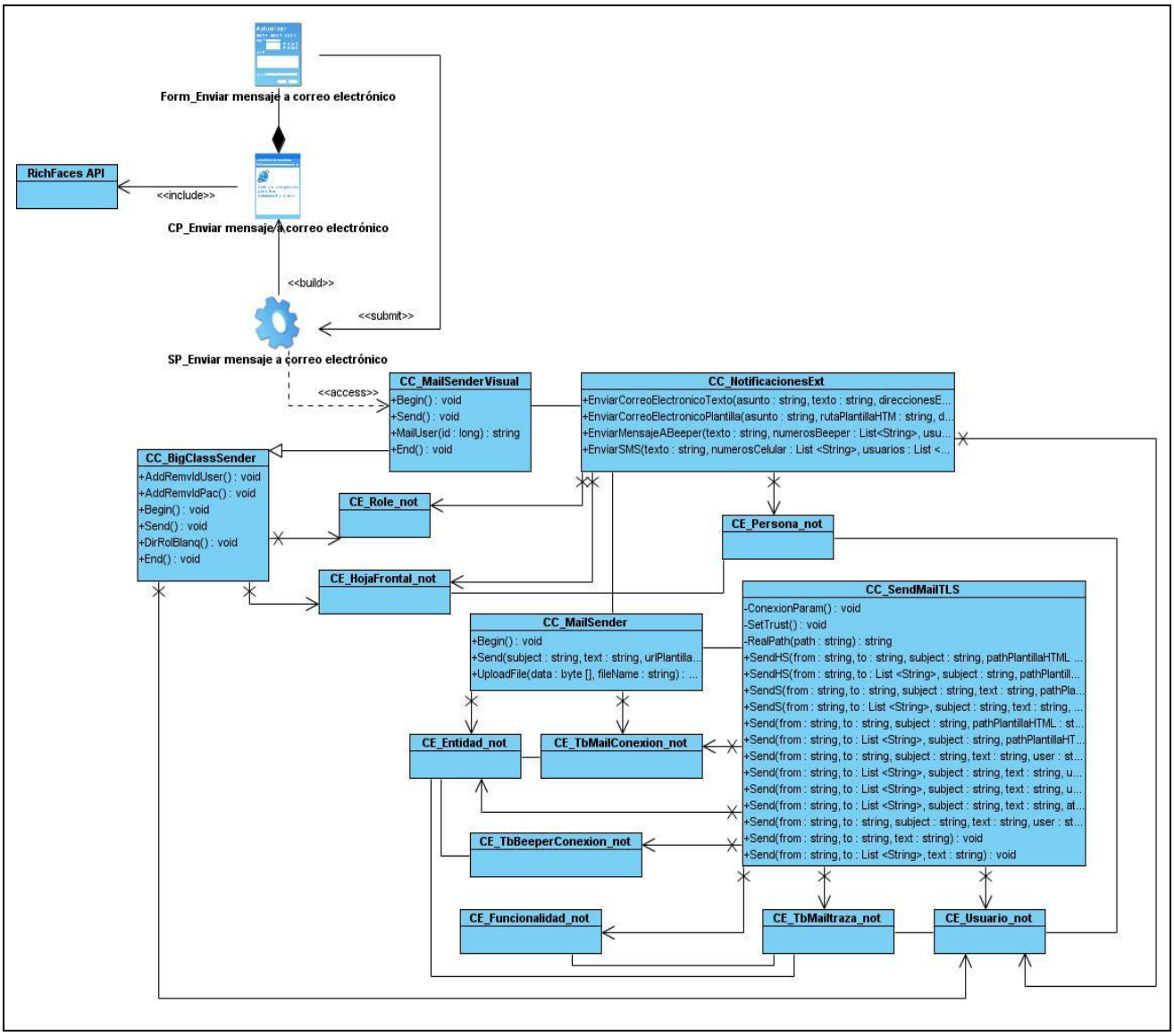


Figura 3.3 Diagrama de Clases del Diseño: *Enviar mensaje a correo electrónico*

DCD_Generar reporte de trazas de correo electrónico

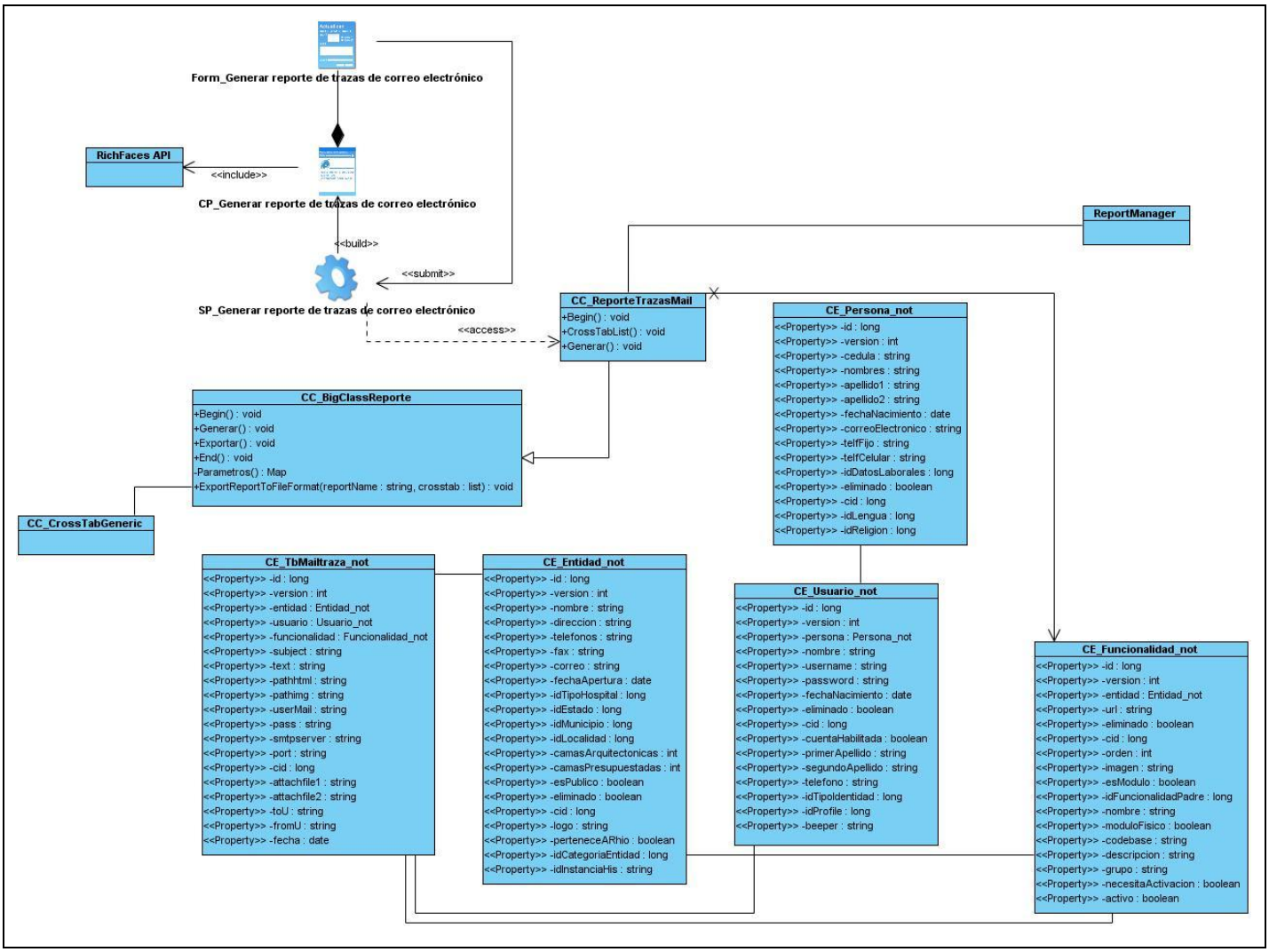


Figura 3.4 Diagrama de Clases del Diseño: *Generar reporte de trazas de correo electrónico*

DS_Generar reporte de trazas de correo electrónico

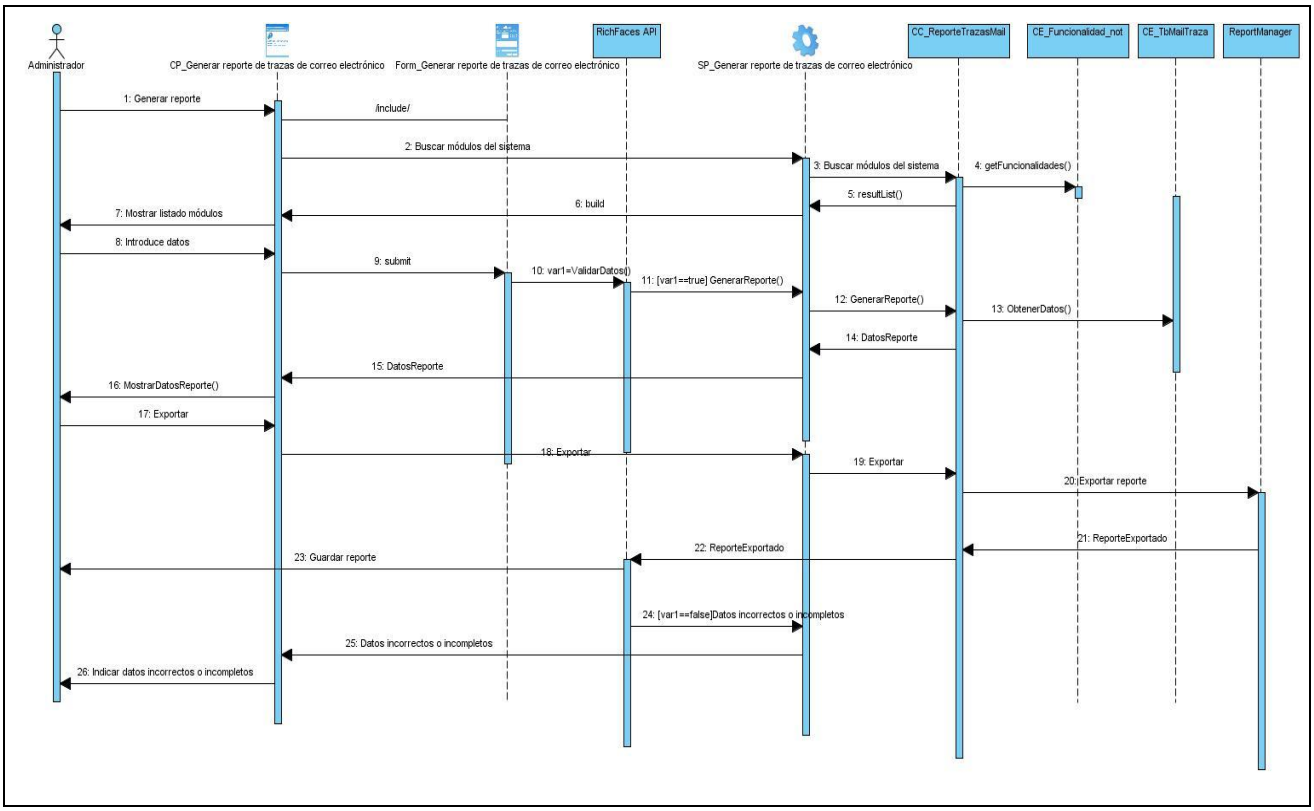


Figura 3.5 Diagrama de Secuencia: *Generar reporte de trazas de correo electrónico*

Las clases del diseño están agrupadas en:

Páginas Servidoras: Están compuestas por componentes Facelets, RichFaces, JSF, Seam UI, así como código HTML. Todo este código será ejecutado en el servidor web, generando páginas clientes que pueden ser representadas por los navegadores web.

Páginas Clientes: Están compuestas por código HTML, CSS, JavaScript. Son interpretadas por los navegadores web, presentándole al usuario la interfaz con la que puede interactuar con el sistema.

Formularios: Un formulario HTML es una sección de un documento enmarcado entre etiquetas <form> y que puede contener elementos especiales llamados controles. Los usuarios normalmente "completan" un formulario modificando sus controles (introduciendo texto, seleccionando objetos de un menú, entre otras opciones), envían la información al servidor donde estos son procesados, lo que constituye una manera de obtener en el servidor información entrada por el usuario en el cliente.

Controladoras: Las clases controladoras se encargan de la implementación de un caso de uso o un proceso en dependencia de la complejidad de los mismos.

3.3.3 Descripción de las clases y sus atributos

A continuación se realiza la descripción de dos de las principales clases que han sido identificadas en el diseño para su futura implementación, con el objetivo de lograr una comprensión más amplia del sistema en cuestión.

Nombre: NotificacionesExt	
Tipo de clase: Controladora.	
Atributo	**Tipo**
expRMail	String
expRBeeperSMS	String
smsSender	SMSSender
beeperSender	BeeperSender
mailSender	MailSender
activeModule	IActiveModule
Para cada responsabilidad:	
Nombre:	EnviarCorreoElectronicoTexto(asunto: String, texto: String, direccionesEspecificas : List<String>, usuarios: List<Usuario_not>, roles: List <Role_not>, pacientes: List <HojaFrontal_not>, nombreAdjunto1 : String, nombreAdjunto2 : String, adjunto1 : [] byte, adjunto2 : [] byte): int
Descripción:	Envía un correo electrónico a una lista de destinatarios (incluidos usuarios, roles y pacientes del sistema), con el texto del mensaje especificado y los demás datos pasadas por parámetro. Retorna el número de direcciones de correo diferentes a las que se envió con éxito la notificación.
Nombre:	EnviarCorreoElectronicoPlantilla(asunto: String, rutaPlantillaHTM: String, direccionesEspecificas: List<String>, usuarios: List<Usuario_not>, roles: List<Role_not>, pacientes: List<HojaFrontal_not>): int.
Descripción:	Envía un correo electrónico a una lista de destinatarios (incluidos usuarios, roles y pacientes del sistema), a partir de un texto que está predefinido en una plantilla. Retorna el número de direcciones de correo diferentes a las que se envió con éxito la notificación.
Nombre:	EnviarMensajeABeeper(texto: String, numerosBeeper: List<String>, usuarios: List<Usuario_not>,

	roles: List<Role_not>) : int
Descripción:	Envía mensajes a los beepers de una lista de destinatarios (incluidos usuarios y roles del sistema). Retorna la cantidad de números de beeper diferentes a los que se envió con éxito la notificación.
Nombre:	EnviarSMS(texto: String, numerosCelular: List<String>, usuarios: List<Usuario_not>, roles: List<Role_not>, pacientes: List<HojaFrontal_not>): int
Descripción:	Envía mensajes a los teléfonos móviles de una lista de destinatarios (incluidos usuarios, roles y pacientes del sistema). Retorna la cantidad de números de celular diferentes a los que se envió con éxito la notificación.

Tabla 3.2 Descripción de la clase controladora: *NotificacionesExt*

Nombre: MailSender	
Tipo de clase: Controladora.	
Atributo	**Tipo**
folderServer	String
ruta	String
from	String
sign	String
serverFileCap	Integer
sendMailTLS	SendMailTLS
activeModule	IActiveModule
Para cada responsabilidad:	
Nombre:	Begin(): void
Descripción:	Obtiene el nombre del usuario por el cual se enviará la notificación, la capacidad máxima de adjuntos del servidor, la ruta de la imagen de la entidad seleccionada para colocarla como pie de firma en el mensaje y la ruta de la carpeta del servidor donde se subirán los adjuntos.
Nombre:	Send(subject: String, text: String, urlPlantillaHTML: String, dest: List<String>, filename: String, filename1: String, data: [] byte, data1: [] byte): int
Descripción:	Verifica que los ficheros a adjuntar no excedan la capacidad máxima y los sube al servidor. Ejecuta diferentes métodos de envío de la clase SendMailTLS, según la información que contendrá el correo electrónico.
Nombre:	UploadFile(data: [] byte, fileName: String): void

Descripción:	Sube al servidor JBoss el fichero que se pasa por parámetro.

Tabla 3.1 Descripción de la clase controladora: *MailSender*

Conclusiones

Con el desarrollo de este capítulo, se definieron las clases necesarias para el correcto funcionamiento del componente de notificaciones. De igual manera fueron descritos los atributos y métodos que deben tener las clases, los cuales servirán de apoyo al desarrollador durante la fase de implementación. Se elaboraron los Diagramas de Clases y los Diagramas de Secuencia correspondientes a cada funcionalidad.

CAPÍTULO 4 IMPLEMENTACIÓN DEL COMPONENTE DE NOTIFICACIONES

En este capítulo se introduce el flujo de trabajo de implementación, partiendo de los resultados obtenidos en el diseño. Primeramente, se detalla el Modelo de Datos, en el que se ve la estructura donde se almacena toda la información requerida en el sistema. Luego se muestra el Modelo de Implementación, que está compuesto por el Diagrama de Despliegue y el Diagrama de Componentes. Estos diagramas, describen los componentes a construir, su organización y dependencias entre los nodos físicos en los que funcionará la aplicación. También se exponen aspectos referentes a la seguridad del sistema, las estrategias de codificación, así como la forma en que se tratarán los errores.

4.1 Modelo de Datos

El Modelo de Datos es la traducción del análisis de requisitos al esquema conceptual, mediante una representación gráfica de las entidades y sus relaciones. Es usado para definir el mapeo entre las clases del diseño y las estructuras de datos. Está compuesto por entidades, atributos y sus relaciones.

Las entidades son objetos de los que el sistema necesita guardar información.

Los atributos son las características asociadas a una entidad. Estos pueden ser clasificados en obligatorios, opcionales, claves foráneas y claves primarias (estas se dividen en simples y compuestas).

Las relaciones, por su parte, muestran la forma en que dos entidades se asocian. Se representan mediante una línea que une a las dos entidades implicadas y manifiestan dos características principales: la cardinalidad y la obligatoriedad.

La cardinalidad se refiere al número de ocurrencias de una entidad con respecto a la otra. La entidad de la que sale la relación tendrá tantas ocurrencias como indique el número asociado a la entidad a donde llega la relación señalando con una flecha el sentido de entrada, de no mostrarse el número de la cardinalidad se asume que la ocurrencia es de solamente una vez.

La obligatoriedad determina que ante la existencia de una entidad puede haber ocurrencias de otras relacionadas con esta. Si la ocurrencia es obligatoria se representa mediante una línea continua, en caso contrario, se realiza a través de una línea discontinua.

La anterior descripción de los componentes del Modelo de Datos, proporcionará un mejor entendimiento del diagrama que se presenta a continuación:

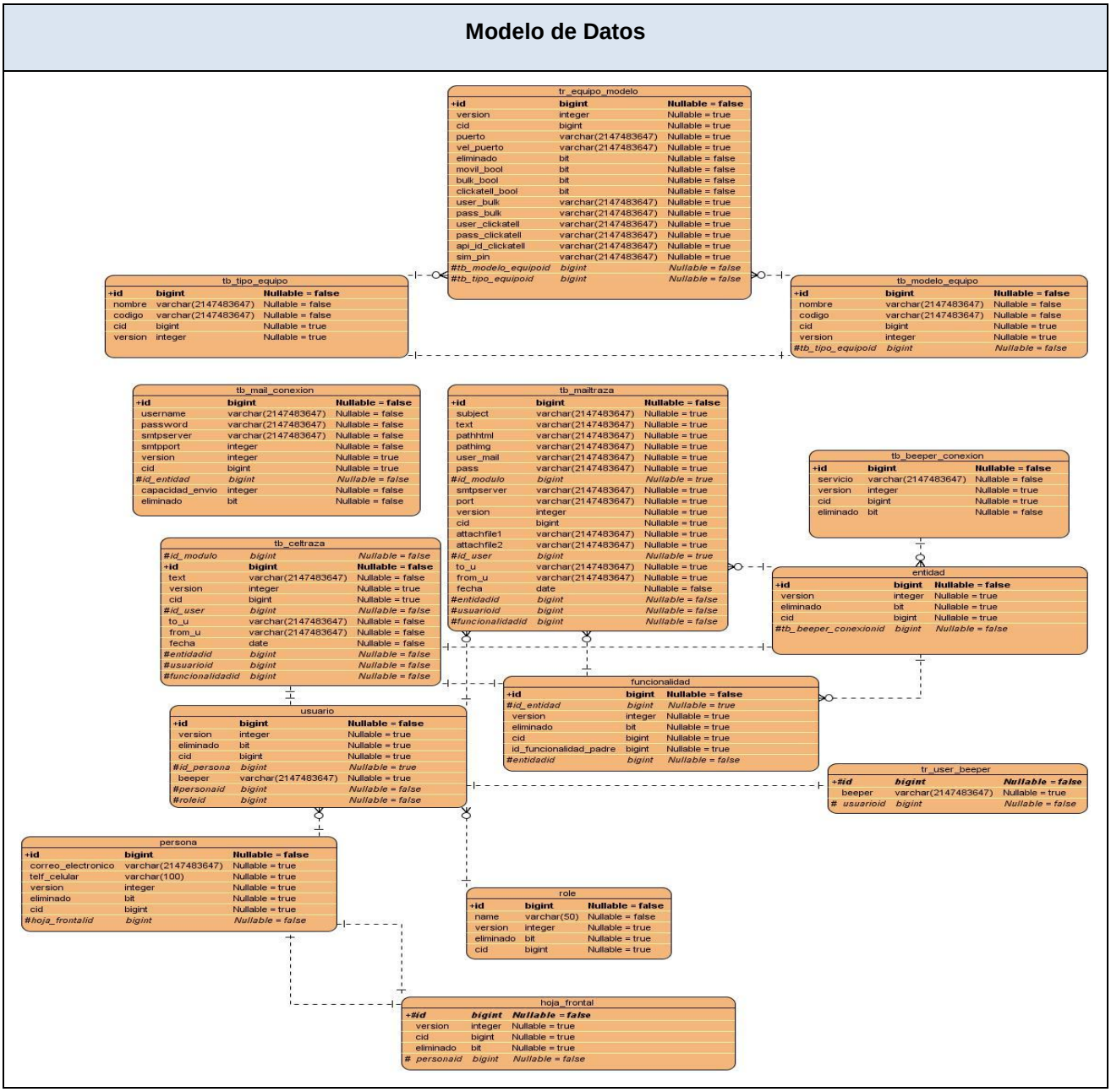


Figura 4.1 Modelo de Datos

4.1.1 Descripción de las tablas de la base de datos

A continuación se describen las entidades más representativas del modelo anterior.

Los siguientes atributos son comunes a todas las entidades ya que fueron agregados con el objetivo de facilitar la implementación de algunas funcionalidades del componente.

Atributo	Tipo	Descripción
Id	long	Identificador necesario en cada entidad para las referencias en las relaciones entre tablas.
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.
cid	long	Permite identificar quién realiza alguna acción sobre la entidad.

Tabla 4.1 Descripción de los atributos comunes entre todas las entidades

tb_mail_conexion		
La tabla contiene los datos necesarios para la configuración de la conexión que permitirá el envío de correos electrónicos.		
Atributo	**Tipo**	**Descripción**
username	varchar	Nombre de un usuario válido para acceder al servidor de correo SMTP. Ejemplo: adarias@estudiantes.uci.cu.
password	varchar	Contraseña para acceder a un servidor de correo SMTP.

smtpserver	varchar	Dirección del servidor de correo SMTP. Ejemplo: smtp.uci.cu.
smtpport	integer	Puerto por el que trabaja el servidor de correo. Ejemplo: 25.
capacidad_envio	integer	Tamaño máximo que pueden tener los adjuntos en conjunto (se registra en Kbyte). Ejemplo: 1024.
eliminado	boolean	Campo que se pone en verdadero (true) si la información de la conexión para el envío de correos electrónicos se elimina. En caso contrario se mantiene en falso (false).

Tabla 4.2 Descripción de la tabla: *tb_mail_conexion*

tb_mailtraza		
La tabla contiene información de la traza que se registrará luego del envío satisfactorio de un correo electrónico.		
Atributo	**Tipo**	**Descripción**
subject	varchar	Asunto del correo electrónico enviado. Puede ser nulo (null).
text	varchar	Texto del correo electrónico enviado.
pathhtml	varchar	Ruta de la plantilla HTML que se utilizó para el envío del correo electrónico.
pathimg	varchar	Ruta de la imagen que se utilizó para el pie de firma.
user_mail	varchar	Usuario de la cuenta en el servidor SMTP desde la cual se envió el correo.

pass	varchar	Contraseña de la cuenta en el servidor SMTP desde la cual se envió el correo.
smtpserver	varchar	Servidor SMTP desde el cual se envió el correo.
port	varchar	Puerto del servidor SMTP por el cual se envió el correo.
attachfile1	varchar	Ruta en el servidor JBoss del primer fichero adjunto enviado.
attachfile2	varchar	Ruta en el servidor JBoss del segundo fichero adjunto enviado.
to_u	varchar	Dirección del destinatario, o una lista de direcciones en caso de que el correo fuera enviado a más de una persona.
from_u	varchar	Remitente del correo electrónico.

Tabla 4.3 Descripción de la tabla: *tb_mailtraza*

tb_celtraza		
La tabla contiene información de la traza que se registrará luego del envío satisfactorio de un mensaje SMS.		
Atributo	**Tipo**	**Descripción**
text	varchar	Texto del mensaje SMS enviado.
to_u	varchar	Número de celular del destinatario, o una lista de números en caso de que el mensaje SMS fuera enviado a más de una persona.
from_u	varchar	Remitente del mensaje SMS.

Tabla 4.4 Descripción de la tabla: *tb_celtraza*

tb_beeper_conexion		
La tabla contiene los datos necesarios para la configuración de la conexión que permitirá el envío de mensajes de texto a beepers.		
Atributo	**Tipo**	**Descripción**
servicio	varchar	Servicio que permite el envío de mensajes de texto a beeper a través del correo electrónico. Ejemplo: paging.movitel.co.cu
eliminado	boolean	Campo que se pone en verdadero (true) si la información de la conexión para el envío de mensajes a beeper se elimina. En caso contrario se mantiene en falso (false).

Tabla 4.5 Descripción de la tabla: *tb_beeper_conexion*

4.2 Modelo de Implementación

El Modelo de Implementación consiste en obtener una visión general de lo que tiene que ser implementado, y una estructura para cada iteración con los componentes y subsistemas a implementar durante esa iteración, así como el testeo que se ha de realizar sobre ellos. Esta vista presenta como objetivo determinar la organización del código, planificar las integraciones de sistemas necesarias en cada iteración e implementar las clases y subsistemas definidos durante el diseño.

4.2.1 Diagrama de Componentes

Un Diagrama de Componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes. Desde el punto de vista del Diagrama de Componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión de software, la reutilización, y las restricciones impuestas por los lenguajes de programación.

A continuación se presenta el Diagrama de Componentes implementado con tres componentes esenciales: Modelo, Vista y Controlador, donde se describen de forma detallada los componentes que usan, sean éstos de código fuente, librerías, binarios o ejecutables, realizado en el Lenguaje Unificado de Modelado.

Diagrama de Componentes

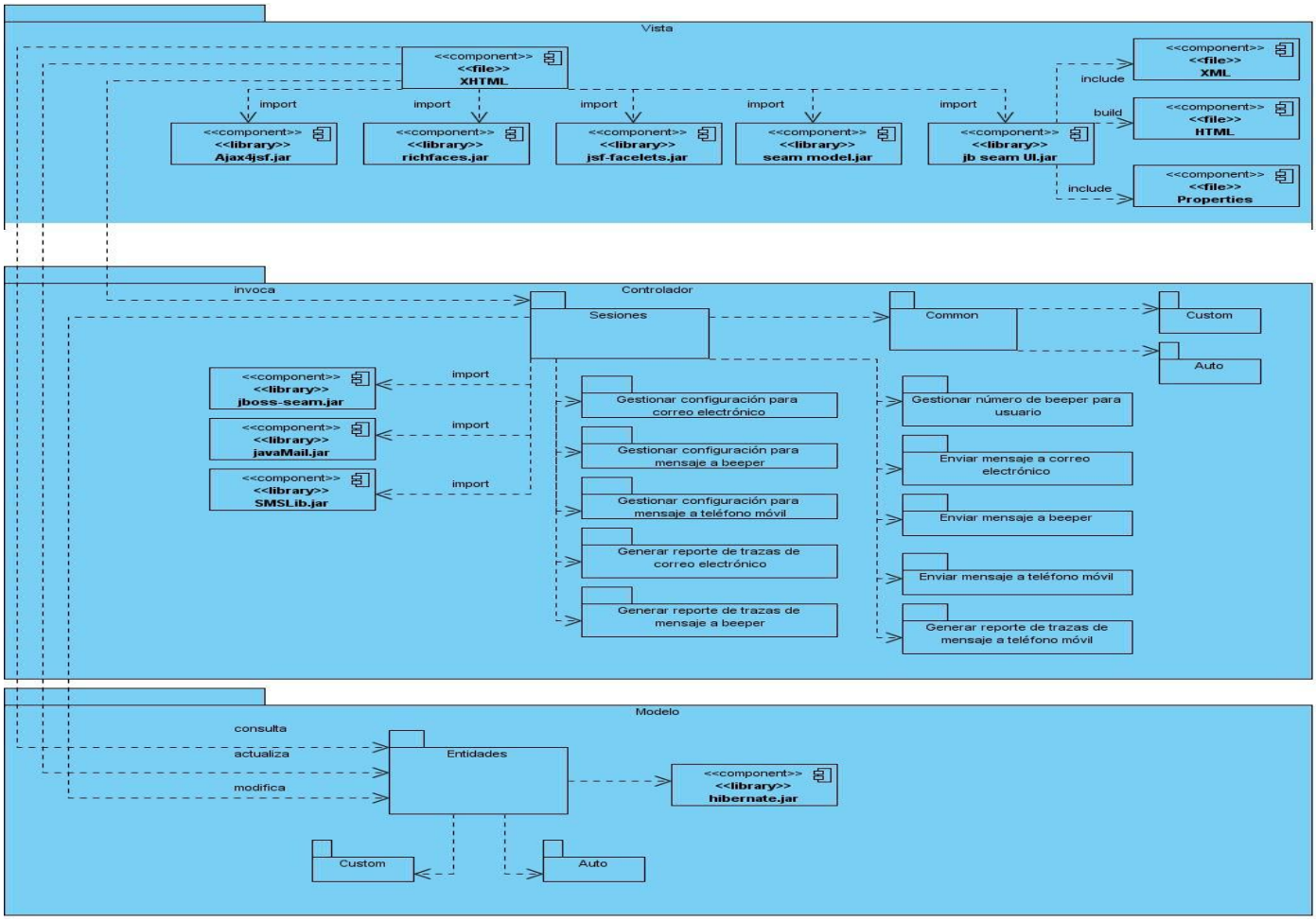


Figura 4.2 Diagrama de Componentes

El diagrama anterior muestra cómo se encuentra estructurado el sistema en componentes. La Vista está formada por las páginas XHTML que importan diversas librerías para su construcción. El paquete Sesiones hace uso de varias librerías y contiene las controladoras autogeneradas, las personalizadas y

las controladoras propias del proceso, que son aquellas que permiten llevar a cabo los requisitos funcionales del sistema. En el Modelo están presentes las entidades autogeneradas y las personalizadas, contenidas todas en el paquete Entidades, el cual utiliza la librería hibernate.jar. Estos paquetes se relacionan entre sí, las vistas consultan y actualizan las entidades e invocan a las controladoras y estas a su vez modifican las entidades.

4.2.2 Diagrama de Despliegue

El Diagrama de Despliegue muestra las relaciones físicas entre los componentes de hardware y software en el sistema final. Es un grafo de nodos unidos por asociaciones de comunicación. Estas asociaciones indican algún tipo de ruta de comunicación entre los nodos, donde estos últimos intercambian objetos o envían mensajes a través de esta ruta.

Asimismo, el tipo de comunicación se identifica con un estereotipo que indica el protocolo de comunicación o la red.

Teniendo en cuenta las características del sistema, el Diagrama de Despliegue quedó estructurado de la siguiente manera:

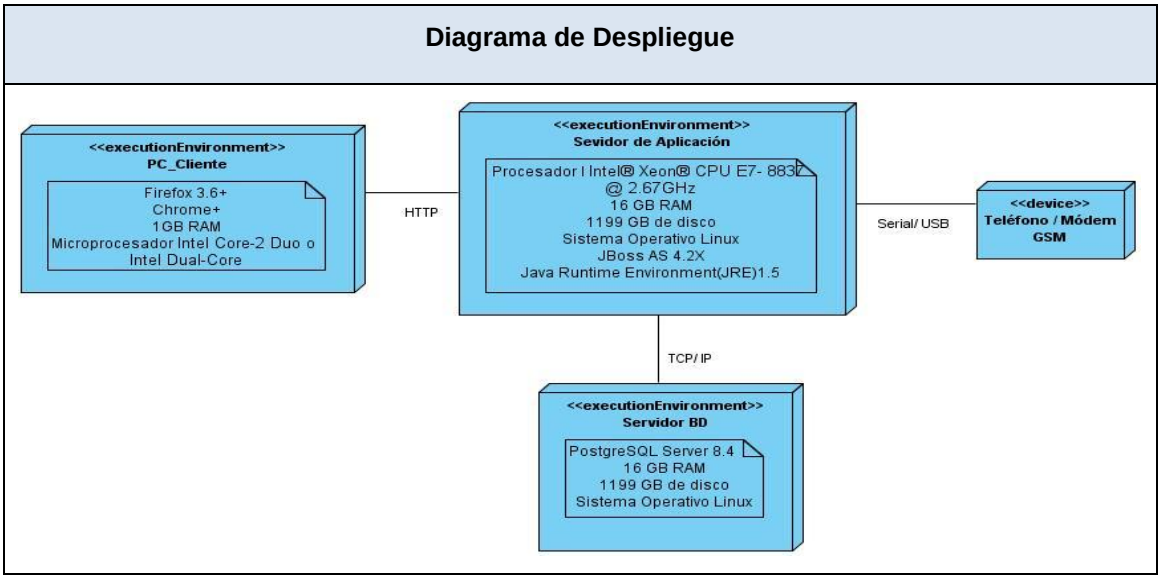


Figura 4.3 Diagrama de Despliegue

Para la implantación y utilización del sistema en un hospital, el usuario debe conectarse al mismo mediante una computadora cliente, haciendo uso de un navegador web. Las peticiones por el protocolo

HTTP serán procesadas por el servidor de aplicaciones, el cual enviará respuesta al cliente. El servidor de aplicaciones emitirá peticiones por el protocolo TCP / IP (Transfer Control Protocol / Internet Protocol) hacia el servidor de base de datos y se encontrará conectado a un teléfono o módem GSM mediante el puerto Serial / USB.

4.3 Tratamiento de errores

El tratamiento de errores es uno de los aspectos más importantes a tener en cuenta durante el desarrollo del sistema. La validación de la información garantiza la corrección y precisión de todos los valores introducidos en la aplicación, además de lograr elevar la calidad de la misma.

A toda porción de código donde pueda surgir cierta situación inesperada se lleva a cabo el control de las excepciones, principalmente donde se ejecutan sentencias que manipulan datos que viajan desde y hacia la base de datos. Por otra parte, se hace uso de la clase Validator con el objetivo de validar los datos entrados por el usuario.

Para el control de las demás excepciones es utilizado el componente FacesMessages del framework Seam, el cual se encarga de mostrar los mensajes que se manejan a través del objeto facesMessages inyectado en las clases controladoras.

4.4 Seguridad

Las notificaciones que se envíen a los pacientes desde el componente, deben respetar la confidencialidad de la información que se comunica, por ejemplo, no se debe emitir el resultado de una prueba sanitaria al teléfono móvil o el correo de un paciente.

En aras de garantizar una correcta protección de los datos manejados, se propone un control de acceso a nivel de usuario y contraseña, con el objetivo de obtener el principio de mínimo privilegio, lo que garantiza el acceso de cada usuario únicamente a los lugares donde tiene permisos.

El registro de trazas de los envíos realizados es vital para mantener un control sobre los mismos. Estos datos sólo podrán ser vistos por un administrador del sistema.

4.5 Estrategias de codificación. Estándares y estilos a utilizar

Un estándar de codificación comprende todos los aspectos de la generación de código, de tal manera que sea práctico y entendible para todos los programadores. Por lo general, incluye pautas sobre cómo

nombrar variables y constantes, dónde ubicar comentarios, cómo poner entre paréntesis, forma de guión, entre otras. No detecta los errores existentes, más bien evita la ocurrencia de estos, lo que permite obtener un código de alta calidad.

4.5.1 Identación

El identado debe ser de dos espacios por bloque de código. No se debe usar el tabulador; ya que este puede variar según la computadora o la configuración de dicha tecla. Los inicios ({) y cierre (}) de ámbito deben estar alineados debajo de la declaración a la que pertenecen y deben evitarse si hay sólo una instrucción.

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

4.5.2 Variables y constantes

El nombre empleado para las variables y constantes, debe permitir que con sólo leerlo se conozca el propósito de la misma.

El nombre que se le da a las variables debe comenzar con la primera letra en minúscula e identificará el tipo de datos al que se refiere. En caso de que sea un nombre compuesto, la segunda palabra, comenzará con letra inicial mayúscula.

Ejemplo: envioCorreo.

Las constantes deben declararse con todas sus letras en mayúsculas.

4.5.3 Comentarios, separadores, líneas, espacios en blanco y márgenes

Ubicación de comentarios: Se recomienda comentar al inicio de cada clase o función de forma que se especifique el objetivo de la misma así como los parámetros que usa (declarar tipos de datos, y objetivo del parámetro) entre otras cosas.

Líneas en blanco: Se recomienda dejar una línea en blanco antes y después de la declaración de una clase o de una estructura y de la implementación de una función.

Espacios en blanco: Se recomienda usar espacios en blanco entre operadores lógicos y aritméticos para lograr una mayor legibilidad del código. Ejemplo: usuario = nombreUsuario. No se debe usar espacio en blanco después del corchete abierto y antes del cerrado de un arreglo, luego del paréntesis abierto y antes del cerrado o antes de un punto y coma.

4.5.4 Clases y objetos

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

Los nombres de las clases deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing. Ejemplo: SmsSender(). Para el caso de las instancias se comenzará con un prefijo que identificará el tipo de dato, este se escribirá en minúscula.

El nombre que se le da a los atributos de las clases debe comenzar con la primera letra en minúscula y estará en correspondencia al tipo de dato al que se refiere, en caso de que sea un nombre compuesto, la segunda palabra comenzará con mayúscula.

Para nombrar las funciones se debe tratar de utilizar verbos que denoten la acción que hacen las mismas. Ejemplo: EnviarCorreo(). Si son funciones que obtienen un dato se emplea el prefijo “get” y si fijan algún valor se emplea el prefijo “set”.

Conclusiones

Una vez concluido este capítulo se pudo dar solución a los requisitos funcionales y no funcionales especificados anteriormente. Para ello se estableció el Modelo de Datos del sistema, el Diagrama de Despliegue, el Diagrama de Componentes y se realizó una descripción de las tablas de la base de datos. Durante el proceso de codificación se cumplió con los estándares y estilos definidos, lo que permitió obtener una aplicación entendible para todos los programadores y fácil de mantener en el transcurso del tiempo.

CONCLUSIONES

Con el desarrollo del componente de notificaciones del Sistema de Información Hospitalaria alas HIS, se concluye lo siguiente:

1. El análisis de los sistemas relacionados con el campo de acción evidenció que los mismos no cumplen con todos las funcionalidades deseadas ni permiten su integración al sistema alas HIS.
2. Se desarrolló un componente de notificaciones para el Sistema de Información Hospitalaria alas HIS, basado en el uso del correo electrónico, el beeper y la telefonía celular.
3. Para el desarrollo del componente se asimiló la arquitectura propuesta por el departamento Sistemas de Gestión Hospitalaria. El diseño propuesto y las tecnologías empleadas se basaron en dicha arquitectura.
4. Se utilizaron patrones de arquitectura que permiten el desarrollo independiente de las capas. Las tecnologías empleadas aportan a la solución web las ventajas de ser multiplataforma, multiusuario y las facilidades de despliegue y mantenimiento.
5. La implementación se basó en tecnologías de desarrollo disponibles sin costo y que aseguran el cumplimiento de los requisitos y la construcción de un componente con atributos de seguridad.
6. Con el componente se espera que los usuarios cuenten con una herramienta atractiva, organizada y eficiente, que permita notificar a pacientes, y al personal médico que no se encuentre interactuando con la aplicación.

RECOMENDACIONES

Para una utilización eficiente del componente de notificaciones y la comprobación de la totalidad de sus funcionalidades, se recomienda tener en cuenta:

- Configurar el número de beeper de cada usuario durante la gestión inicial del usuario en el sistema.
- Implementar mecanismos que permitan el envío automático de notificaciones por parte del sistema.
- Extender el uso del componente de notificaciones a todos los módulos del sistema alas HIS que lo requieran.
- Probar desde el exterior el envío masivo de SMS a través de los operadores internacionales BulkSMS y Clickatell.

REFERENCIAS BLIBLIOGRÁFICAS

1. Empresas especializadas en servicios de Tecnologías de la Información y la Comunicación BEIT y PYMES. ServiciosTIC. *Definición de TIC.* [En línea] Latitud Web, 2006. [Citado el: 22 de noviembre de 2012.] http://www.serviciostic.com/las-tic/definicion-de-tic.html.

2. Gatica Lara, Florina , Fernández Puerto, Fernando J y Hernández López, Ana María. Manual de Introducción a la Informática Médica. *Sistemas de Información Hospitalaria.* [En línea] 2003. [Citado el: 22 de noviembre de 2012.] http://www.facmed.unam.mx/emc/computo/ssa/HIS/his.pdf.

3. Definición.DE. *Definición de notificación.* [En línea] 2008. [Citado el: 27 de noviembre de 2012.] http://definicion.de/notificacion/.

4. Diccionarist. [En línea] 2011. [Citado el: 5 de diciembre de 2012.] http://definicion.dictionarist.com/beeper.

5. Kioskea.net. Kioskea.net. *Estándar GSM (Sistema global de comunicaciones móviles).* [En línea] mayo de 2013. http://es.kioskea.net/contents/681-estandar-gsm-sistema-global-de-comunicaciones-moviles.

6. Definición.DE. *Definición de móvil.* [En línea] 2008. [Citado el: 6 de diciembre de 2012.]

7. Punto Flotante, S.A. *Tutorial: los módems GSM y GPRS.* [En línea] [Citado el: 29 de enero de 2013.] http://www.puntoflotante.net/TUTORIAL-MODEM-GSM-GPRS.htm.

8. MICHIGAN INGENIERIA INFORMATICA S.A. *Hospitales y Centros de Salud.* Córdoba : s.n. Documento en formato PDF.

9. QSOFT. *Software Salus.* 2009. Dossier de Descripción. Documento en formato PDF.

10. Equipo de desarrollo. NetClinicas. NetClinicas. Software gestión de clínicas y centros médicos. *Características de NetClinicas.* [En línea] 1999. [Citado el: 12 de diciembre de 2012.] http://www.netclinicas.com/software-clinicas.html.

11. Bird, Adam y Hucker, Julian. esendex. *Email SMS.* [En línea] 2001. [Citado el: 12 de diciembre de 2012.] http://www.esendex.es/blog/post/los-sms-reducen-un-20-el-%20%20absentismo-en-los-hospitales/.

12. Hospital Bouquet Roldan. *Turnos por mensaje de texto, una buena experiencia que se extenderá a otros centros de salud y a nuestro vecino país de Chile.* [En línea] 2010. [Citado el: 12 de diciembre de 2012.] http://hospitalbouquetroldan.blogspot.com/2011_04_01_archive.html.

13. Piva, Elisa, y otros, y otros. ASCP. *Evaluation of Effectiveness of a Computerized Notification System for Reporting Critical Values.* [En línea] 2013. [Citado el: 10 de enero de 2013.] http://ajcp.ascpjournals.org/content/131/3/432.long.

14. *Capítulo 5.Cliente - Servidor.* pág. 9, PDF.

15. Kioskea.net. *Entorno cliente/servidor.* [En línea] [Citado el: 24 de enero de 2013.] http://es.kioskea.net/contents/148-entorno-cliente-servidor.

16. Sánchez González, Carlos. Proyecto de fin de carrera. *Aplicaciones en capas.* [En línea] 28 de septiembre de 2004. [Citado el: 17 de enero de 2013.] http://oness.sourceforge.net/proyecto/html/ch03s02.html.

17. Díaz González, Yanette y Fernández Romero, Yenisleidy. Revista Telem@tica. *Patrón Modelo-Vista-Controlador.* [En línea] 2012. [Citado el: 10 de febrero de 2013.] http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/15.

18. Instituto de Física Aplicada del CSIC. ¿Qué es Java? *Características del lenguaje Java.* [En línea] 1997. [Citado el: 20 de enero de 2013.] http://www.iec.csic.es/criptonomicon/java/quesjava.html.

19. Universidad Interamericana para el Desarrollo. *Posgrado.* [En línea] [Citado el: 10 de febrero de 2013.] http://moodle.unid.edu.mx/dts_cursos_mdl/maestria_en_tecnologias_de_informacion/aplicaciones_web/sesion2/actividades/Programacion_por_capas.pdf.

20. Ferguson, Jhon. [En línea] 2007. [Citado el: 2 de febrero de 2013.] http://www.wakaleo.com/public_resources/jsf-jumpstarter.pdf.

21. Ghia, Dustin. Programación Práctica. *JEE5 - Fundamentos.* [En línea] 31 de marzo de 2011. [Citado el: 15 de diciembre de 2012.] http://programmabilis.blogspot.com/2011/03/i1-fundamentos-de-jee5.html.

22. Newton, Mark, y otros, y otros. WildFly. *RichFaces.* [En línea] 2008. [Citado el: 3 de febrero de 2013.] http://www.jboss.org/richfaces.

23. JBoss Community. *Jboss ajax4jsf.* [En línea] 2007. [Citado el: 5 de febrero de 2013.] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.

24. *9 Tecnologías.* pág. 23, PDF.

25. LIBROSWEB. *HTML y XHTML.* [En línea] 2010. [Citado el: 6 de febrero de 2013.] http://www.librosweb.es/xhtml/capitulo_1/html_y_xhtml.html.

26. Java.net. The Source for Java Technology Collaboration. *JavaMail API documentation.* [En línea] 2013. [Citado el: 15 de febrero de 2013.] https://javamail.java.net/nonav/docs/api/.

27. SMSLib. A universal API for sms messaging. *About SMSLib.* [En línea] [Citado el: 15 de febrero de 2013.] http://smslib.org/doc/about/.

28. JBossCommunity. HIBERNATETools. *Hibernate Tools for Eclipse and Ant.* [En línea] [Citado el: 15 de febrero de 2013.] http://www.hibernate.org/subprojects/tools.html.

29. Universidad de los Andes, Departamento de Sistemas. Enterprise Java Bean 3. *Características.* [En línea] [Citado el: 15 de febrero de 2013.] http://sistemas.uniandes.edu.co/~isis3702/dokuwiki/lib/exe/fetch.php?media=principal:isis3702-ejb3.pdf.

30. Rondon Grados, Luis . JAVA J2EE. *JPA - Java Persistence API .* [En línea] 28 de agosto de 2009. [Citado el: 15 de febrero de 2013.] http://luchorondon.blogspot.com/2009/04/jpa-java-persistence-api.html.

31. Jaramillo M, Wilmer. Fedora People. *JBoss Application Server.* [En línea] 2006. [Citado el: 10 de febrero de 2013.] http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf.

32. Osmosis Latina. *JVM ("Java Virtual Machine").* [En línea] 2000. [Citado el: 16 de febrero de 2013.] http://www.osmosislatina.com/java/basico.htm.

33. Popkin Software and Systems. *Modelado de Sistemas con UML.* pág. 20, PDF.

34. Eclipse Foundation. Eclipsepedia. *FAQ What is Eclipse?* [En línea] 2004. [Citado el: 16 de febrero de 2013.] http://wiki.eclipse.org/FAQ_What_is_Eclipse%3f.

35. PostgreSQL. *What is PostgreSQL?* [En línea] 20 de mayo de 2009. [Citado el: 16 de febrero de 2013.] http://wiki.postgresql.org/wiki/FAQ#What_is_PostgreSQL.3F_How_is_it_pronounced.3F_What_is_Postgres.3F.

36. Free Download Manager. *Visual Paradigm para UML.* [En línea] marzo de 2007. [Citado el: 17 de febrero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

37. Canales Mora, Roberto . AdisctosAlTrabajo. *Introducción a iReport.* [En línea] Autentia, 2003. [Citado el: 18 de febrero de 2013.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=ireport.

BIBLIOGRAFÍA

- *9 Tecnologías.* p. 23, PDF.
- Bird, Adam and Hucker, Julian. esendex. *Email SMS.* [Online] 2001. [Cited: diciembre 12, 2012.] http://www.esendex.es/blog/post/los-sms-reducen-un-20-el-%20%20absentismo-en-los-hospitales/.
- Bird, Adam and Hucker, Julian. esendex. *Email SMS.* [Online] 2001. [Cited: diciembre 12, 2012.] http://www.esendex.es/blog/post/los-sms-reducen-un-20-el-absentismo-en-los-hospitales/.
- Canales Mora, Roberto . AdisctosAlTrabajo. *Introducción a iReport.* [Online] Autentia, 2003. [Cited: febrero 18, 2013.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=ireport.
- *Capítulo 5.Cliente - Servidor.* p. 9, PDF.
- Definición.DE. *Definición de móvil.* [Online] 2008. [Cited: diciembre 6, 2012.]
- Definición.DE. *Definición de notificación.* [En línea] 2008. [Citado el: 27 de noviembre de 2012.] http://definicion.de/notificacion/.
- Díaz González, Yanette and Fernández Romero, Yenisleidy. Revista Telem@tica. *Patrón Modelo-Vista-Controlador.* [Online] 2012. [Cited: febrero 10, 2013.] http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/15.
- Diccionarist. [Online] 2011. [Cited: diciembre 5, 2012.] http://definicion.dictionarist.com/beeper.
- Eclipse Foundation. Eclipsepedia. *FAQ What is Eclipse?* [Online] 2004. [Cited: febrero 16, 2013.] http://wiki.eclipse.org/FAQ_What_is_Eclipse%3f.
- Empresas especializadas en servicios de Tecnologías de la Información y la Comunicación BEIT y PYMES. ServiciosTIC. *Definición de TIC.* [En línea] Latitud Web, 2006. [Citado el: 22 de noviembre de 2012.] http://www.serviciostic.com/las-tic/definicion-de-tic.html.
- Equipo de desarrollo. NetClinicas. NetClinicas. Software gestión de clínicas y centros médicos. *Características de NetClinicas.* [Online] 1999. [Cited: diciembre 12, 2012.] http://www.netclinicas.com/software-clinicas.html.

- Ferguson, Jhon. [Online] 2007. [Cited: febrero 2, 2013.] http://www.wakaleo.com/public_resources/jsf-jumpstarter.pdf.
- Free Download Manager. *Visual Paradigm para UML.* [Online] marzo 2007. [Cited: febrero 17, 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.
- García Carmona, Juan. Patrones. *GRASP: Alta cohesión y bajo acoplamiento.* [Online] [Cited: marzo 12, 2013.] http://www.grasp-alta-cohesion-y-bajo-acoplamiento.html.
- Gatica Lara, Florina , Fernández Puerto, Fernando J y Hernández López, Ana María. Manual de Introducción a la Informática Médica. *Sistemas de Información Hospitalaria.* [En línea] 2003. [Citado el: 22 de noviembre de 2012.] http://www.facmed.unam.mx/emc/computo/ssa/HIS/his.pdf.
- Ghia, Dustin. Programación Práctica. *JEE5 - Fundamentos.* [Online] marzo 31, 2011. [Cited: diciembre 15, 2012.] http://programmabilis.blogspot.com/2011/03/i1-fundamentos-de-jee5.html.
- Gómez Lara, Cipriano. *Formas o medios de hacer las notificaciones.* p. 7.
- Hospital Bouquet Roldan. *Turnos por mensaje de texto, una buena experiencia que se extenderá a otros centros de salud y a nuestro vecino país de Chile.* [Online] 2010. [Cited: diciembre 12, 2012.] http://hospitalbouquetroldan.blogspot.com/2011_04_01_archive.html.
- Instituto de Física Aplicada del CSIC. ¿Qué es Java? *Características del lenguaje Java.* [Online] 1997. [Cited: enero 20, 2013.] http://www.iec.csic.es/criptonomicon/java/quesjava.html.
- Jamae, David and Johnson, Peter. *JBoss in Action.* 2009. p. 10, PDF.
- Jaramillo M, Wilmer. Fedora People. *JBoss Application Server.* [Online] 2006. [Cited: febrero 10, 2013.] http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf.
- Java.net. The Source for Java Technology Collaboration. *JavaMail API documentation.* [Online] 2013. [Cited: febrero 15, 2013.] https://javamail.java.net/nonav/docs/api/.
- JBoss Community. *Jboss ajax4jsf.* [Online] 2007. [Cited: febrero 5, 2013.] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.

- JBossCommunity. HIBERNATETools. *Hibernate Tools for Eclipse and Ant.* [Online] [Cited: febrero 15, 2013.] http://www.hibernate.org/subprojects/tools.html.
- Kioskea.net. *Entorno cliente/servidor.* [Online] [Cited: enero 24, 2013.] http://es.kioskea.net/contents/148-entorno-cliente-servidor.
- Kioskea.net. Kioskea.net. *Estándar GSM (Sistema global de comunicaciones móviles).* [Online] mayo 2013. http://es.kioskea.net/contents/681-estandar-gsm-sistema-global-de-comunicaciones-moviles.
- Lewis, jackson . eHow. *Cómo enviar SMS con Java.* [Online] 1999. [Cited: marzo 10, 2013.] http://www.librerias java para envio de sms/Cómo enviar SMS con Java _ eHow en Español.htm.
- LIBROSWEB. *HTML y XHTML.* [Online] 2010. [Cited: febrero 6, 2013.] http://www.librosweb.es/xhtml/capitulo_1/html_y_xhtml.html.
- Maldonado, Daniel M. El CoDiGo K. *Arquitectura de programación en 3 capas.* [Online] 2007. [Cited: diciembre 10, 2012.] http://www.elcodigok.com.ar/2007/09/arquitectura-de-programacion-en-3-capas/.
- MASTERMAGAZINE. *Definición de Arquitectura de Software.* [Online] 2007. [Cited: enero 5, 2013.] http://www.Definición de Arquitectura Software - Significado y definición de Arquitectura Software.html.
- Meneses Snáchez, Jesús David and Marín Martínez, Juan David. *Java y USB.* 2008. p. 45.
- MICHIGAN INGENIERIA INFORMATICA S.A. *Hospitales y Centros de Salud.* Córdoba : s.n. Documento en formato PDF.
- Newton, Mark, et al., et al. WildFly. *RichFaces.* [Online] 2008. [Cited: febrero 3, 2013.] http://www.jboss.org/richfaces.
- Orshalick , Jacob . Refcardz. *SeamUI.* [Online] [Cited: noviembre 30, 2013.] http://refcardz.dzone.com/refcardz/seam-ui.
- Osmosis Latina. *JVM ("Java Virtual Machine").* [Online] 2000. [Cited: febrero 16, 2013.] http://www.osmosislatina.com/java/basico.htm.

- Palos, Juan Antonio. *API JavaMail.* p. 21, Tutorial.
- Piva, Elisa, et al., et al. ASCP. *Evaluation of Effectiveness of a Computerized Notification System for Reporting Critical Values.* [Online] 2013. [Cited: enero 10, 2013.] http://ajcp.ascpjournals.org/content/131/3/432.long.
- Popkin Software and Systems. *Modelado de Sistemas con UML.* p. 20, PDF.
- PostgreSQL. *What is PostgreSQL?* [Online] mayo 20, 2009. [Cited: febrero 16, 2013.] http://wiki.postgresql.org/wiki/FAQ#What_is_PostgreSQL.3F_How_is_it_pronounced.3F_What_is_Postgres.3F.
- Puelles , Lizana Esther . El Rinconcito Informático. *Modelado de Sistemas con UML.* [Online] 2000. [Cited: enero 15, 2013.] http://www.elrinconcito.com/articulos/modeladoUML/modeladoUML.html.
- Punto Flotante, S.A. *Tutorial: estándares de comunicaciones RS232, RS422/485.* [Online] [Cited: enero 29, 2013.] http://www.puntoflotante.net/RS485.htm.
- Punto Flotante, S.A. *Tutorial: los módems GSM y GPRS.* [Online] [Cited: enero 29, 2013.] http://www.puntoflotante.net/TUTORIAL-MODEM-GSM-GPRS.htm.
- QSOFT. *Software Salus.* 2009. Dossier de Descripción. Documento en formato PDF.
- Ramos González, Victoria. *Las TIC en el sector de la salud.* 2007. p. 7.
- Rondon Grados, Luis . JAVA J2EE. *JPA - Java Persistence API .* [Online] agosto 28, 2009. [Cited: febrero 15, 2013.] http://luchorondon.blogspot.com/2009/04/jpa-java-persistence-api.html.
- Sánchez González, Carlos. Proyecto de fin de carrera. *Aplicaciones en capas.* [Online] septiembre 28, 2004. [Cited: enero 17, 2013.] http://oness.sourceforge.net/proyecto/html/ch03s02.html.
- Santos Jaime, Luz Marina. *GestaSM: Librería Java SMS usando SMPP3.4.* Grupo de Investigación Ciencias Computacionales, Universidad de Pamplona. Colombia : s.n. p. 7.
- SMSLib. A universal API for sms messaging. *About SMSLib.* [Online] [Cited: febrero 15, 2013.] http://smslib.org/doc/about/.

- Universidad de los Andes, Departamento de Sistemas. Enterprise Java Bean 3. *Características.* [Online] [Cited: febrero 15, 2013.] http://sistemas.uniandes.edu.co/~isis3702/dokuwiki/lib/exe/fetch.php?media=principal:isis3702-ejb3.pdf.
- Universidad Interamericana para el Desarrollo. *Posgrado.* [Online] [Cited: febrero 10, 2013.] http://moodle.unid.edu.mx/dts_cursos_mdl/maestria_en_tecnologias_de_informacion/aplicaciones_web/sesion2/actividades/Programacion_por_capas.pdf.

ANEXOS

Anexo 1: Interfaces de usuario para la configuración del componente.

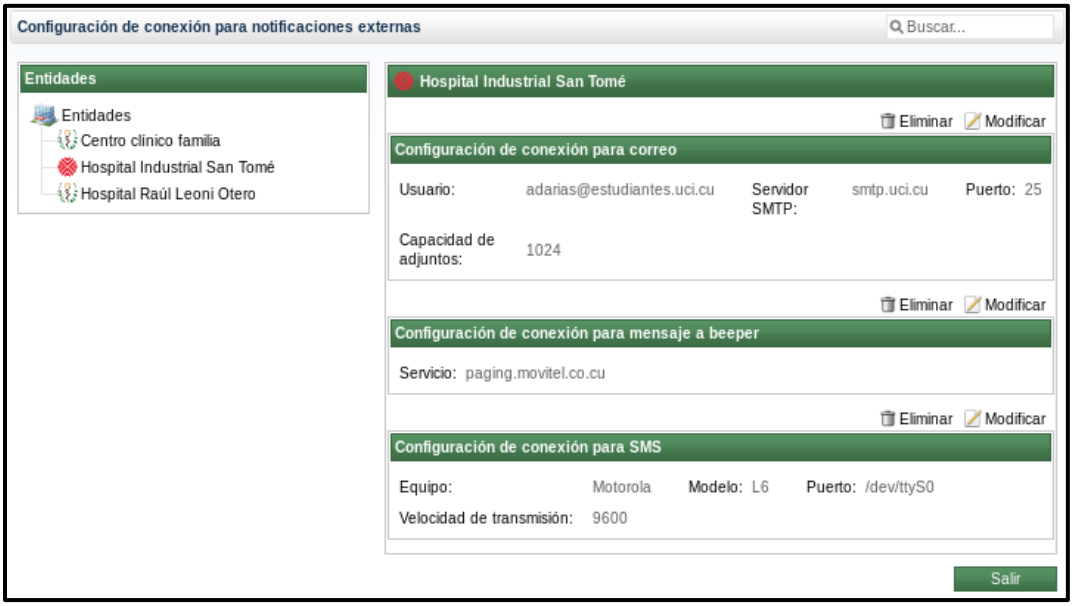


Figura A1: Configurar parámetros de conexión para notificaciones

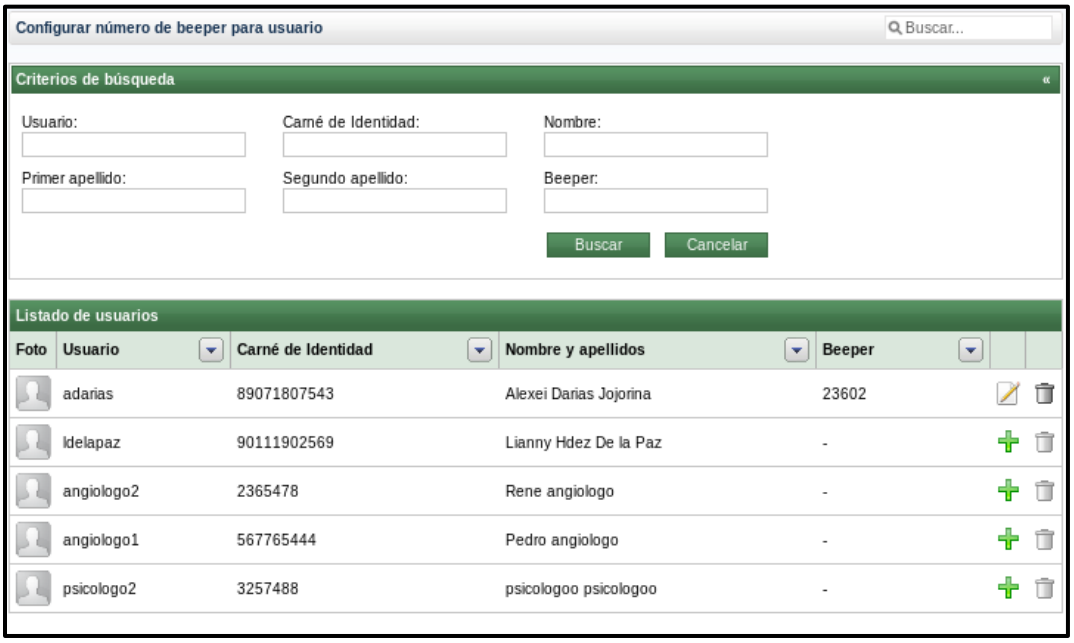


Figura A2: Configurar número de beeper para usuario

Anexo 2: Interfaces de usuario para el envío de notificaciones.

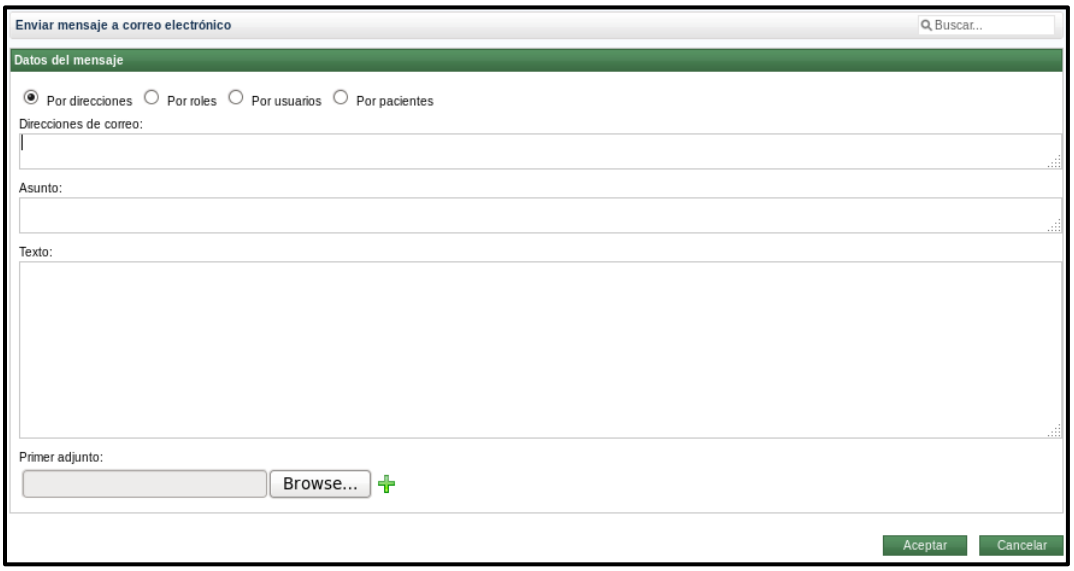


Figura A3: Enviar mensaje a correo electrónico (por direcciones)

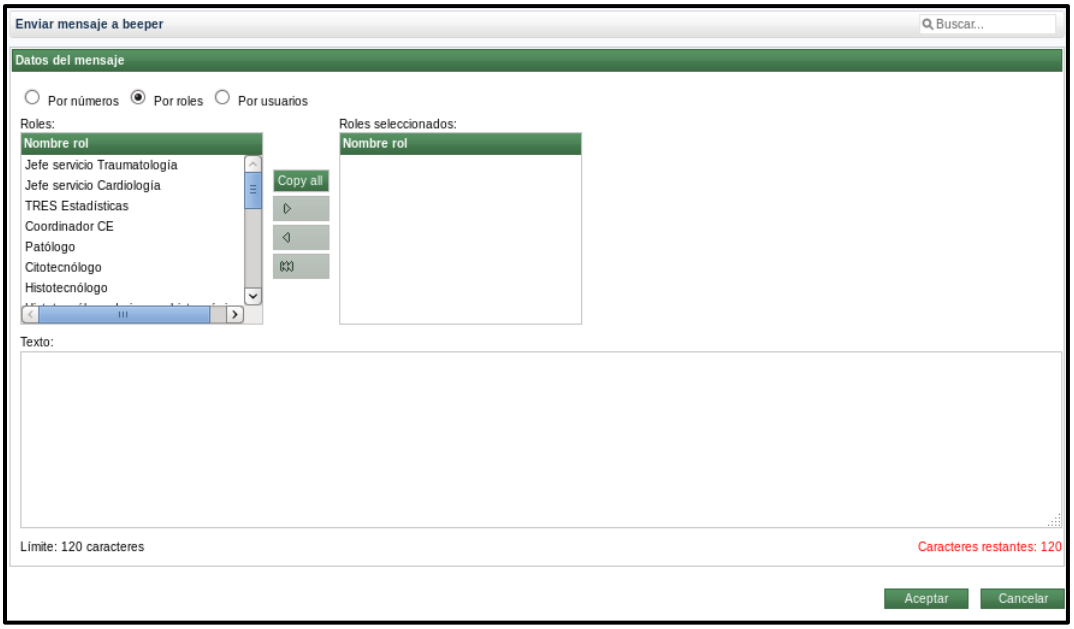


Figura A4: Enviar mensaje a beeper (por roles)

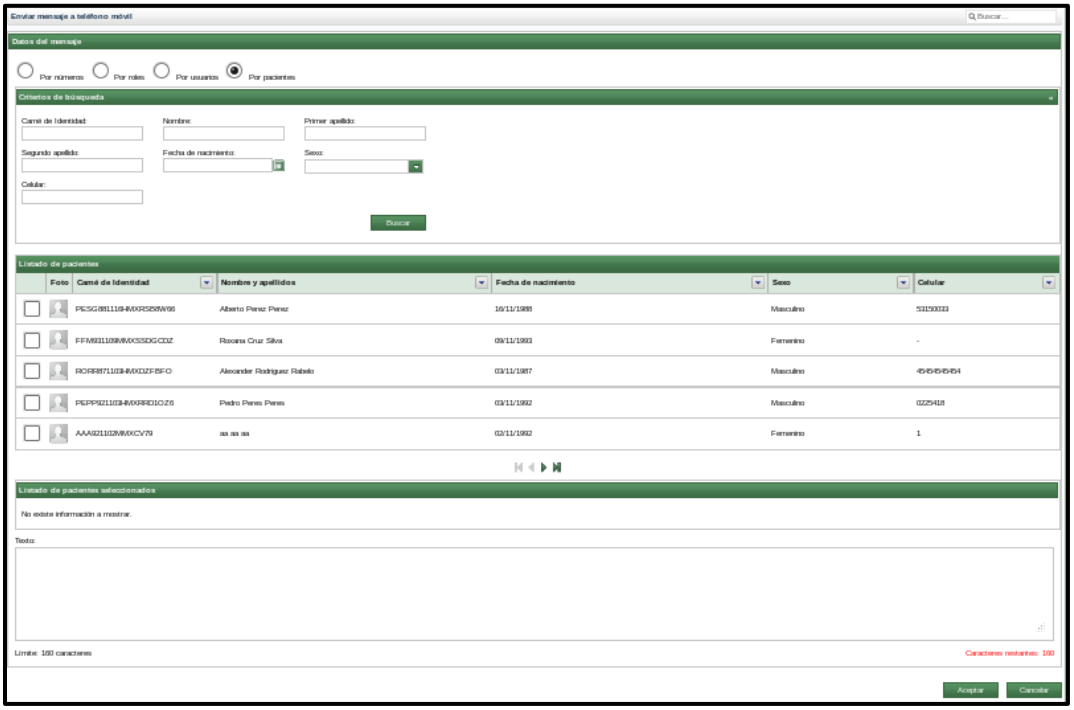
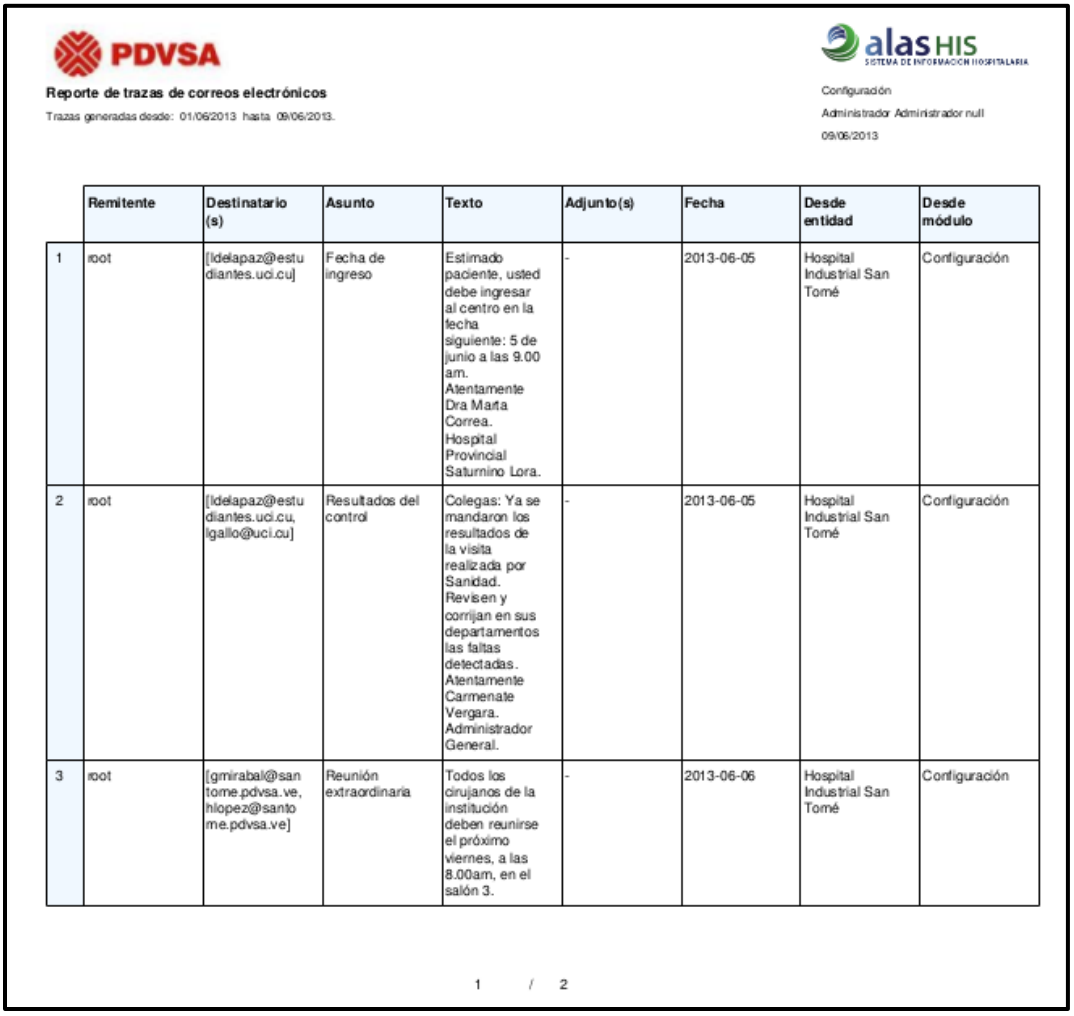


Figura A5: Enviar mensaje a teléfono móvil (por pacientes)

Anexo 3: Interfaces de usuario para la generación de trazas de envíos realizados.

Figura A6: Seleccionar parámetros para generar reporte de trazas de correos electrónicos



PDVSA

alas HIS
SISTEMA DE INFORMACIÓN HOSPITALARIA

Reporte de trazas de correos electrónicos

Trazas generadas desde: 01/06/2013 hasta 09/06/2013.

Configuración
Administrador Administrador null
09/06/2013

	Remitente	Destinatario (s)	Asunto	Texto	Adjunto(s)	Fecha	Desde entidad	Desde módulo
1	root	[idelapaz@estudiantes.uci.cu]	Fecha de ingreso	Estimado paciente, usted debe ingresar al centro en la fecha siguiente: 5 de junio a las 9.00 am. Atentamente Dra Marta Correa. Hospital Provincial Saturnino Lora.	-	2013-06-05	Hospital Industrial San Tomé	Configuración
2	root	[idelapaz@estudiantes.uci.cu, lgallo@uci.cu]	Resultados del control	Colegas: Ya se mandaron los resultados de la visita realizada por Sanidad. Revisen y corrijan en sus departamentos las faltas detectadas. Atentamente Carmenate Vergara. Administrador General.	-	2013-06-05	Hospital Industrial San Tomé	Configuración
3	root	[gmirabal@santome.pdvsa.ve, hlopez@santome.pdvsa.ve]	Reunión extraordinaria	Todos los cirujanos de la institución deben reunirse el próximo viernes, a las 8.00am, en el salón 3.	-	2013-06-06	Hospital Industrial San Tomé	Configuración

1 / 2

Figura A7: Reporte de trazas de correos electrónicos

Anexo 4: Posibles escenarios en los que se puede hacer uso del componente de notificaciones.

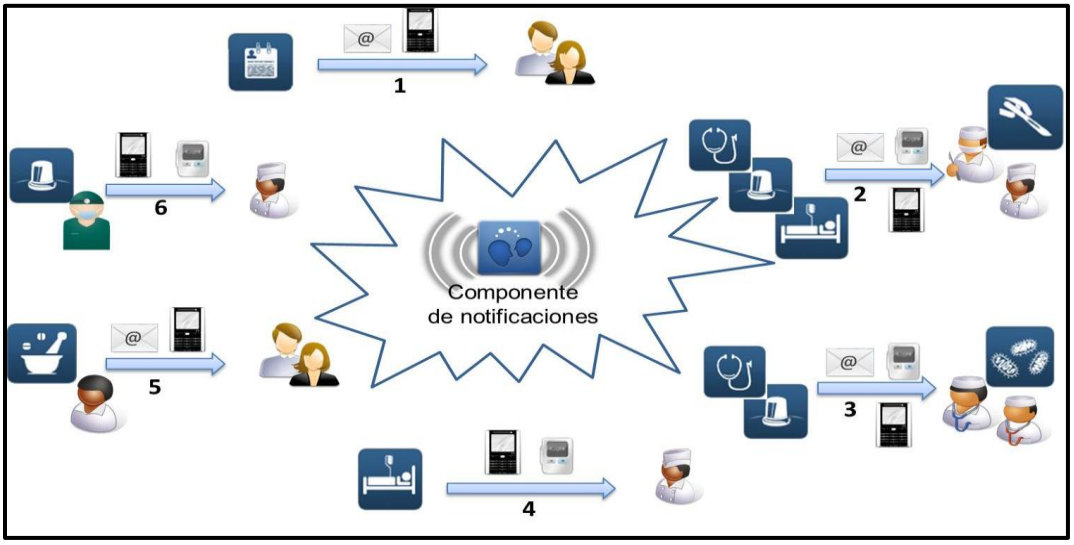


Figura A8: Posibles escenarios en los que se puede hacer uso del componente de notificaciones

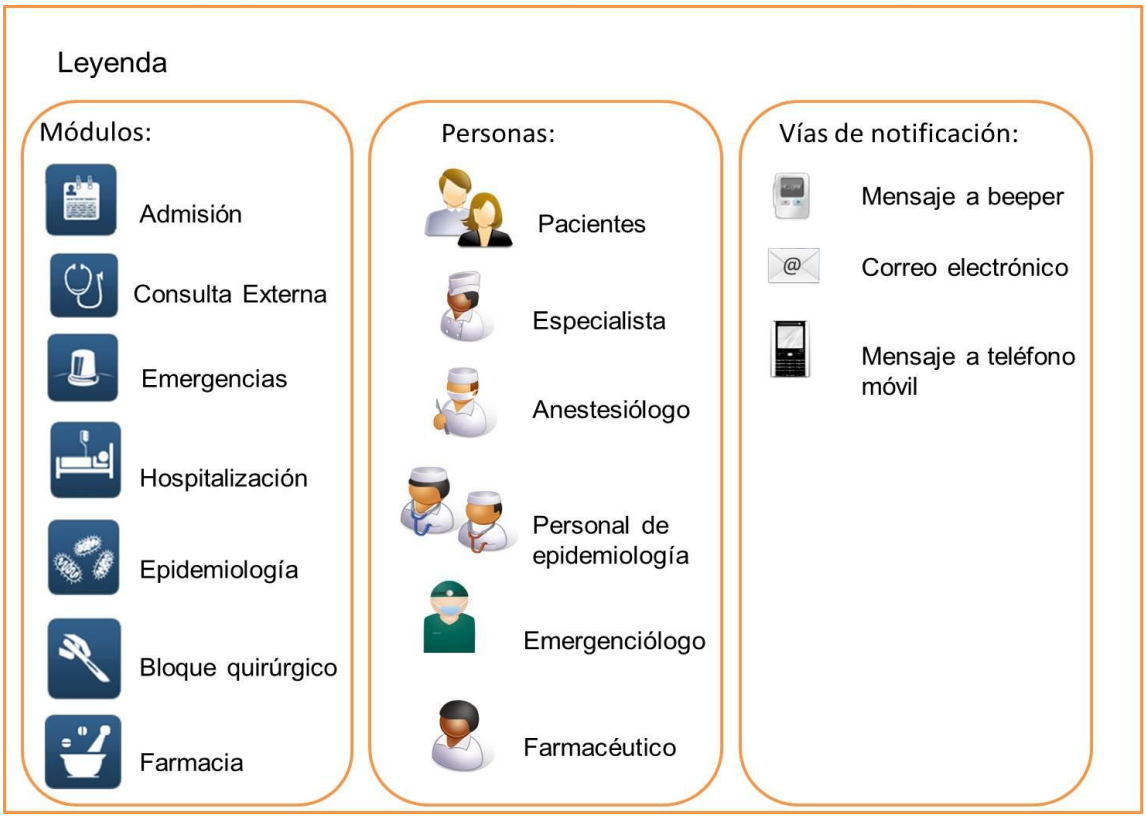


Figura A9: Leyenda

1. Desde el módulo Admisión se les puede notificar a los pacientes, mediante correo electrónico o mensaje a teléfono móvil, la fecha próxima de ingreso.

2. Desde Consulta Externa, Emergencias u Hospitalización se les puede comunicar, enviando un correo electrónico, un mensaje a beeper o un SMS, al anestesiólogo y al cirujano de guardia de la especialidad requerida, sobre una cirugía de urgencia.
3. Al ser detectada una Enfermedad de Notificación Obligatoria (ENO) en Consulta Externa o Emergencias, se le puede notificar su descubrimiento, mediante el correo electrónico, o el envío de mensajes a beeper o teléfono celular, al personal de Epidemiología para que tome las medidas pertinentes. Algunas de las enfermedades consideradas como ENO son: la malaria, el cólera, la leptospirosis, el dengue, la fiebre amarilla, el VIH, entre otras.
4. El personal de Hospitalización puede avisar el empeoramiento del estado de salud de un paciente, remitiendo un mensaje al beeper o al teléfono celular del especialista que lo atiende.
5. El farmacéutico puede enviar un correo o un SMS a aquellos pacientes que requieran de un medicamento que se encontraba en déficit.
6. Una vez estabilizado un paciente, el emergenciólogo puede mandar un mensaje al beeper o al celular del especialista que esté de guardia, para que establezca un nuevo tratamiento al doliente.

Anexo 5: Configuraciones necesarias para el funcionamiento de la librería SMSLib, utilizando Eclipse.

Instalación de la librería SMSLib.

1. Para el correcto funcionamiento de la librería, se debe tener instalado Java SUN JDK 1.6 o superior.
2. Se debe instalar además alguna librería de comunicación para Java, se recomienda SUN Java Comm v2 para sistemas de Win32 y Java Comm v3 o RxTx para sistemas Linux, Unix, BSD y otros. Para el desarrollo del componente de notificaciones del sistema alas HIS se seleccionó RxTx.

   Instalación de RxTx:

   El paquete binario oficial de RxTx está disponible en:

   http://rxtx.qbang.org/pub/rxtx/rxtx-2.1-7-bins-r2.zip

Para instalarlo, se descomprime el compactado en una carpeta temporal. El fichero *RXTXcomm.jar* se debe copiar en el directorio: *JDKDIR/jre/lib/ext/*. Las librerías necesarias (por ejemplo, *librxtxSerial.so* para Linux de 32 bits) deben copiarse en *JDKDIR/jre/bin/*. Si se tiene un directorio JRE aparte, se deben realizar las mismas copias para el directorio *JREDIR.*

3. Se deben copiar las siguientes librerías para el directorio *JBossDIR/server/default/lib*:
   commons-net-3.0.1.jar
   jsmpp-2.1.0.jar
   log4j-1.2.16.jar
   slf4j-api-1.6.3.jar
   slf4j-log4j12-1.6.3.jar
   smslib-3.5.2.jar

   Cada una de estas librerías se encuentran dentro del paquete de SMSLib, en la carpeta *lib*.

Detección del teléfono / módem GSM

La librería SMSLib detecta teléfonos o módems GSM conectados a la computadora mediante el puerto serial. Estos dispositivos también pueden ser conectados mediante puerto USB. Para el envío de SMS desde el Sistema Operativo Windows, se necesitan instalar los drivers correspondientes al teléfono o módem que se va a utilizar. Los drivers permiten simular un puerto serial a partir de la conexión por vía USB. La instalación de drivers no es necesaria en sistemas Linux.

Las pruebas de envío de SMS se realizaron mediante un móvil Motorola L6. Al conectar dicho dispositivo en una computadora con el Sistema Operativo Ubuntu 10.04, se reconoce como un módem PPP (módem que utiliza el Protocolo Punto a Punto para conectarse a Internet), y lo monta en */dev/ACM0*. La librería SMSLib trabaja con los dispositivos conectados a puertos serial, los que en Ubuntu 10.04 se montan en */dev/ttyS0*, */dev/ttyS1*, */dev/ttyS2/*, etc. Por esta razón, para el correcto funcionamiento de SMSLib, se necesita vincular el dispositivo ACM0 con el S0, lo cual se logra con la siguiente línea de comando:

sudo ln -sf /dev/ttyACM0 /dev/ttyS0

GLOSARIO DE TÉRMINOS

RS232: Norma para comunicación punto a punto, en donde se tiene una computadora que se encuentra trasmitiendo hacia un equipo esclavo ubicado a distancias no mayores a 50 metros y a una velocidad máxima de 19.200 bps. Este tipo de trasmisión se le conoce como "single ended" porque usa en el cable un solo retorno. Es un modo de trasmisión muy simple, pero también vulnerable al ruido auditivo en la línea y únicamente es empleada para comunicación punto a punto.

Ventanilla virtual: Suministra al ciudadano toda la información que necesita y le evita parcial o totalmente la presencia en el establecimiento oficial.

Transaccionalidad: Se refiere a la interacción con una estructura de datos compleja, donde los procesos deben aplicarse uno después del otro de manera similar a una interacción atómica.

MIME: Extensiones multipropósitos de correo que permite codificar en el Código Estándar Estadounidense para el Intercambio de Información (ASCII, pos sus siglas en inglés) todo tipo de archivos (texto, audio, vídeo, imágenes, etc.) de forma transparente al usuario, con el objetivo de poder trasportar los mensajes que contengan cualquiera de estos adjuntos mediante el protocolo SMTP.

POP3: Protocolo diseñado para recibir correo, no para enviarlo, permite a los usuarios con conexiones intermitentes o muy lentas (como las conexiones por módem), descargar su correo electrónico mientras tienen conexión y revisarlo posteriormente incluso estando desconectados.

IMAP: Protocolo de acceso a mensajes electrónicos almacenados en un servidor. Mediante dicho protocolo, el usuario puede acceder al correo electrónico desde cualquier equipo que tenga una conexión a Internet.

RFC822: Norma para el formato de correo electrónico que solo soporta texto en el cuerpo del mensaje.

PDU: Protocolo que trata el SMS como una cadena de caracteres en octetos hexadecimales o semioctetos decimales, de cuya codificación resulta el SMS en modo texto. La ventaja del modo PDU respecto al modo texto es que en modo texto la

aplicación queda limitada a la opción de codificación que se haya preestablecido, mientras que en la forma PDU se puede implementar cualquier codificación.

SMPP: Protocolo estándar de telecomunicaciones pensado para el intercambio de SMS entre equipos que gestionan este tipo de mensajes.

Licencia GNU LGPL: También conocida como GPL Reducida de GNU. Permite el uso de librerías en programas privativos.