Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

TÍTULO: *MÓDULO EMERGENCIAS DEL SISTEMA DE INFORMACIÓN HOSPITALARIA ALAS HIS*

Autores: Lorena Alemán Antelo

Juan Manuel Garcia Orduñez

Tutores: Ing. Alexander Rodríguez Rabelo

Ing. Joselín Miló Pérez

Ciudad de La Habana

Junio de 2009

"Año del 50 aniversario del triunfo de la Revolución"

La mayoría de las ideas fundamentales de la ciencia son esencialmente sencillas y por regla general pueden ser expresadas en un lenguaje comprensible para todos.

Albert Einstein

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 26 días del mes de junio del año 2009.

Juan Manuel Garcia Orduñez

_____

Firma del Autor

Lorena Alemán Antelo

_____

Firma del Autor

Alexander Rodríguez Rabelo

_____

Firma del Tutor

Joselín Miló Pérez

_____

Firma del Tutor

DATOS DE CONTACTO.

Síntesis del Tutor: Ing. Alexander Rodríguez Rabelo, Instructor recién graduado en el año 2007 de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Profesor de la Facultad # 7. Ha impartido las asignaturas Matemática 3 y Matemática Numérica. Se ha desempeñado como jefe de proyecto en el Área Temática de Gestión Hospitalaria.

e-mail: arodriguezra@uci.cu

Síntesis del Tutor: Ing. Joselín Miló Pérez, Instructor recién graduado en el año 2008 de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas. Profesor de la Facultad # 7. Ha impartido las asignaturas Física I y Física II. Se ha desempeñado como desarrollador y jefe de proyecto en el Área Temática de Gestión Hospitalaria.

e-mail: jmilo@uci.cu

AGRADECIMIENTOS.

Gracias a nuestros profesores por su entrega, en especial a Maykell Sánchez por el ejemplo a seguir que ha sido siempre para nosotros y su amistad incondicional. Fue quien nos enseñó que hay un mundo de información valiosa por aprender. A Keila, Kenia, Dayneris y Nadiezka mil gracias por la amistad tan linda que han compartido con nosotros y la ayuda que nos han brindado en todo momento.

A nuestras familias les agradecemos el esfuerzo que han realizado siempre para darnos lo mejor y educarnos como hombres de bien.

Especial agradecimiento al Doctor Carlos Guilbeaux por compartir sus conocimientos médicos en apoyo de este trabajo y brindar su bella amistad.

A nuestros compañeros de estos 5 años por ayudarnos en la travesía de esta carrera llena de retos.

A Osmany, Carlos y Estela le agradecemos su colaboración en muchas de las tareas de este trabajo.

A nuestro tutores por su apoyo en la revisión y perfeccionamiento de este trabajo.

DEDICATORIA.

Lorena:

Dedico este trabajo a mi familia, quienes me ayudaron con su apoyo incondicional a estar más cerca de mis metas profesionales y han apoyado siempre mis locos proyectos e ideas que no terminan nunca. En especial a mi abuela querida por su infinito amor y comprensión. A mis padres que juntos estuvieron siempre a mi lado. A mi hermano y mi tía quienes sé que me quieren mucho. A Gilberto por compartir mis estados de ánimo y apoyarme… por aconsejarme en todo momento y quererme con el corazón.

Juan Manuel:

A toda mi familia y de forma especial a mis padres, a mi hermano y a mi abuela, por entregarme de forma incondicional su amor y dedicación, su apoyo en todo momento y haberme encaminado de forma correcta en la vida.

RESUMEN.

El presente trabajo se centra en el desarrollo de un sistema que facilite la gestión de la información clínica de los pacientes en el área de Emergencias de las instituciones hospitalarias. Este, forma parte de un Sistema de Información Hospitalaria que tiene como objetivo fundamental registrar, organizar, actualizar, conservar y mostrar la información de forma dinámica y segura.

Su desarrollo está basado en tecnologías libres o de código abierto, multiplataformas y sobre una arquitectura en capas, empleándose el patrón Modelo Vista Controlador. Se utilizó Java como lenguaje de programación, PostgreSQL 8.3 como servidor de aplicaciones, el framework Hibernate para el acceso a datos, el framework Seam para la unión entre la capa de presentación y acceso a datos y el motor de reglas de negocio Drools para lograr gran flexibilidad y adaptación a las reglas del negocio.

El uso de la aplicación trae beneficios para el paciente ya que tendrá una Historia Clínica única y centralizada que garantiza el seguimiento y la seguridad de la información médica. Por lo que posibilitará mayor rapidez y calidad en el servicio recibido. En cuanto al médico este podrá acceder a la información clínica del paciente para facilitar los procesos de diagnóstico, tratamiento y seguimiento y podrá realizar investigaciones médicas mediante la revisión de los diagnósticos.

Palabras Claves

Emergencias, Sistema de Información Hospitalaria.

TABLA DE CONTENIDOS.

ÍNDICE DE TABLAS.

ÍNDICE DE FIGURAS.

INTRODUCCIÓN.

Según expertos la humanidad se encuentra actualmente en la Era de la información. Caracterizada por un movimiento de la información más rápido que el clásico movimiento físico. Esta se convierte en un recurso de incalculable valor y el poder radica en la capacidad de gestionarla de forma eficiente.

La informática es la ciencia que estudia los métodos, procesos y técnicas de manejo de la información. Esta ha tenido un desarrollo extraordinario expandiéndose a las disímiles ramas de la sociedad por lo que puede afirmarse que el futuro estará estrechamente vinculado al desarrollo de esta ciencia.

Uno de los sectores donde ha tenido gran aplicación es la Salud, quizás por el hecho de que en este se maneja gran cantidad de información que se necesita tener bien clasificada y accesible en cualquier momento, para que el servicio brindado resulte óptimo. Además de que es un sector de vital importancia para el desarrollo de cualquier país y a su vez constituye un elemento indispensable para medir el nivel de vida de la población. (1)

En un inicio los sistemas que existían en los hospitales se dedicaban a tareas administrativas, sistemas para admisión y altas, toma de órdenes, revisión de resultados, control de inventarios, entre otras. Luego, su uso se extendió al área clínica, acceso a la Historia Clínica, aporte de documentación bibliográfica de consulta, entre otros, con el objetivo de mejorar la calidad y rapidez de la atención y de facilitar las tareas de investigación. (2)

En la actualidad, un sistema informático que gestione la información generada en un hospital debe contener: el registro del paciente con expediente digital, control de laboratorios y quirófanos desde una sala de mando, generación de recetas, órdenes o formularios de trabajo y generación de reportes estadísticos.

Los sistemas informáticos que gestionan la información generada en un hospital, son conocidos por sus siglas en inglés como HIS (Hospital Information System), en español, Sistema de Información

Hospitalaria. Estos permiten la recolección, almacenamiento, procesamiento, recuperación y comunicación de la información relacionada con la atención al paciente y la labor administrativa de todas las actividades del hospital o centro de atención médica. La información comprende el estado de salud de la población y los datos para la toma de decisiones administrativas, clínico-epidemiológicas, estratégicas y operativas.

Además de las ventajas técnicas que ofrece la utilización de un HIS, están las mejoras en la calidad de los servicios ofrecidos impactando directamente en la población que los recibe. Un ejemplo es contar con un registro histórico de la información médica del paciente que proporciona un fácil manejo de la información, lo que ayuda a disminuir el tiempo de atención de los pacientes y aumenta la veracidad de los diagnósticos. (3)

Cabe resaltar lo engorroso que resulta el proceso de generar estadísticas con un mínimo de error, ya que se ha de consultar el Libro de pacientes atendidos para obtener la cifra de casos atendidos en un rango de fecha realizándose un conteo día por día. Para saber la cantidad de casos atendidos por una especialidad el proceso se vuelve más complejo y la probabilidad de cometer errores en la obtención de las cifras es mayor. Las estadísticas en esta área son de suma importancia ya que pueden influir en la toma de decisiones y en futuras planificaciones tanto a nivel primario como a nivel superior por lo que se necesita que sean cifras reales.

Por otra parte el área de Emergencias no está vinculada con el resto de los servicios y áreas del hospital lo que trae inconvenientes en la comunicación de las solicitudes y respuesta de los exámenes realizados a un paciente. Esto se evidencia cuando se necesita realizar una solicitud a un área externa y se debe realizar la coordinación entre áreas enviando a un mensajero para confrontar la disponibilidad de recursos con que se cuenta. El mensajero debe estar pendiente al resultado de dicha solicitud e ir a buscarlo en reiteradas ocasiones hasta obtenerlo.

Los procesos manuales traen demoras adicionales en la prestación de los servicios. Muestra de esto es la realización de los exámenes complementarios, pues ha de hacerse primeramente la solicitud y enviarla junto con la muestra al laboratorio. Una vez obtenido los resultados ha de notificarse al área de Emergencias para que pase a buscarlos. Existen muchos protocolos en la gestión de la información lo que

aumenta el tiempo de espera el cual en esta área es vital para concluir el diagnóstico y aplicar un tratamiento al paciente. (4)

Como se ha descrito, la ausencia de un sistema informático que permita el manejo de la información en esta área y una interrelación con las demás áreas y servicios del hospital es una gran limitante para brindar un servicio óptimo.

En este sentido, se tiene como ***problema a resolver***: ¿Cómo facilitar la gestión de información relacionada con los procesos en el área de Emergencias de las instituciones hospitalarias? El ***objeto de estudio*** lo constituye el proceso de gestión de información en las instituciones hospitalarias. Enmarcado en el ***campo de acción*** proceso de gestión de información en el área de Emergencias en las instituciones hospitalarias.

Para resolver el problema identificado se propone el siguiente ***objetivo general***: Desarrollar el módulo de Emergencias del Sistema de Información Hospitalaria alas HIS, que facilite la gestión de información en esta área de las instituciones hospitalarias.

Para dar cumplimiento al objetivo anteriormente planteado se definen las siguientes ***tareas de la investigación:***

1. Analizar los procesos de negocio asociados al área de Emergencias de las instituciones hospitalarias.
2. Evaluar las tendencias actuales en el mundo de los Sistemas de Información Hospitalaria.
3. Asimilar la arquitectura definida por el Área Temática Gestión Hospitalaria para el desarrollo de sus aplicaciones.
4. Obtener mediante el Proceso Unificado de Desarrollo, los flujos de trabajo “Modelado de Negocio”, “Gestión de Requerimientos”, “Diseño” e “Implementación”.
5. Implementar los procesos Recibir paciente y Atender paciente del módulo Emergencias del Sistema de Información Hospitalaria alas HIS.

En este sentido se puede destacar que el desarrollo del módulo de Emergencias del Sistema de Información Hospitalaria alas HIS, proporcionará un grupo de beneficios a la sociedad, dígase pacientes, médicos y sector administrativo:

1. Calidad en los informes médicos que se le entregan a los pacientes.
2. Seguridad y confidencialidad de la información médica del paciente.
3. Organización de los diferentes procesos que se llevan a cabo en el área de Emergencias garantizando la calidad del servicio médico.
4. Disposición de herramientas de gestión clínica y gestión administrativa que den respuesta a las necesidades reales del profesional médico.
5. Disponer y acceder a información única e integrada del paciente para facilitar los procesos de diagnóstico, tratamiento y otros programas de cuidado y seguimiento.

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

Capítulo 1: Fundamentación Teórica: Estudio preliminar de los Sistemas de Información Hospitalarios existentes en nuestro país y en el mundo. Tecnologías, metodologías y herramientas de desarrollo a utilizar.

Capítulo 2: Características del sistema: Descripción de las funcionalidades de la solución propuesta a la problemática planteada.

Capítulo 3: Diseño del sistema: Modelación y construcción de la estructura de la aplicación.

Capítulo 4: Implementación: Implementación de las clases y subsistemas en términos de componentes de la solución propuesta.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

El presente capítulo describe de forma breve los sistemas que gestionan la información del área de Emergencias de las instituciones hospitalarias, dando a conocer el estado del arte y sus características fundamentales. Se explican las tecnologías y herramientas empleadas en el desarrollo de la solución propuesta.

1.1 Conceptos básicos relacionados con el dominio del problema.

Para comprender los términos empleados en esta investigación se ha de conocer el ámbito hospitalario, por lo que a continuación se explican algunos conceptos que no forman parte del lenguaje común.

Se menciona *Emergencias* como el área donde se reciben a los pacientes que necesitan atención médica de forma inmediata. También es preciso aclarar que bajo este mismo concepto se emplea ocasionalmente el término Urgencias respetando la bibliografía consultada.

Una vez llegado el paciente a la Emergencia se realiza el *triaje*. Consiste en clasificar el nivel de gravedad del paciente dado un código de colores (Rojo, Amarillo y Verde). El color Rojo indica que el paciente está muy grave, el color Amarillo es menos grave y el Verde significa gravedad leve. La información asociada a la atención del paciente se registra en la Hoja de emergencia y esta se crea cada vez que dicho paciente visita esta área por alguna dolencia.

La Hoja de evolución es otro documento que se crea durante la atención al paciente para registrar periódicamente cómo va reaccionando ante el tratamiento y las valoraciones del médico acerca del caso.

1.2 Antecedentes.

La idea de usar computadoras en los hospitales con el objetivo de facilitar el trabajo, nació tan pronto como estuvieron disponibles en el mercado. Su uso temprano fue calcular las estadísticas hospitalarias y llevar el manejo de inventarios y sistemas de finanzas.

No fue hasta 1968 que surgió el respaldo al desarrollo de los HIS por parte del Centro Nacional de Investigación y Servicios de Salud (National Center of Research & Health Services) de los Estados Unidos de América. Se inició el uso de computadoras y aplicaciones orientadas a labores administrativas en ambientes multiusuario y multitareas. La información era procesada y almacenada en un mismo lugar, pero no era integral, por lo tanto se debía conectar todas las partes de un hospital en un único HIS.

Mientras esto sucedía los HIS eran “Islas Computarizadas” debido a que diferentes partes del hospital o diferentes usuarios dentro del propio hospital desarrollaban sus propios sistemas completamente independientes uno del otro.

Estos sistemas corrían en computadoras muy grandes y difíciles de utilizar, además tenían baja capacidad y rendimiento. El número de terminales que podían ser conectadas era limitado. Las razones para mantener un sistema funcionando era tener acceso a la información con propósitos de planeación y la producción de estadísticas del hospital.

Floreció la edad de la computadora personal más conocida por sus siglas en ingles como PC. Algunos sistemas especializados por departamentos empezaron a desarrollarse, ejecutando el conjunto de tareas asociadas a sus áreas. Los costos disminuyeron y se podían resolver más problemas. Se generó la necesidad de compartir información, debido a que se tenían resueltos los problemas de manera aislada pero no existía una solución integral.

Más tarde surgió la tecnología de red, lo cual motivó un segundo intento por desarrollar HIS sobre infraestructuras de redes locales. La utilización de esta tecnología facilita la realización de más del 80% de las tareas operativas, que permitía optimizar funciones y dedicar más tiempo a la atención del paciente y la investigación.

La informatización de los procesos hospitalarios trae mejoras en la eficiencia de la atención médica y en los circuitos administrativos. El hecho de mantener agrupada y disponible la información relacionada con un paciente para su uso posterior, garantiza la continuidad asistencial al ciudadano. También se vela por la confidencialidad de sus datos clínicos y personales, evitando la pérdida de información y disminuyendo el espacio físico necesario para el almacenamiento de la misma. (5)

1.3 HIS existentes que incluyen en su solución el módulo de Emergencias.

Se han encontrado diferentes HIS que gestionan la información hospitalaria del área de Emergencias. Uno de ellos es *Sigho*. Este software se apoya de estándares internacionales para el diagnóstico de enfermedades y realización de procedimientos, tales como el CIE-10 y CIE-9MC. Es una aplicación web y para su desarrollo utilizaron las mejoras realizadas en los Servicios de Internet Information Server 6.0 (IIS 6.0), Microsoft ASP.NET y Microsoft .NET Framework. Como gestor de base de datos utiliza SQL Server 2000. El primer componente que debe tener instalado el equipo Servidor es el Sistema Operativo Windows y en una versión Server 2000 o superior.

Este sistema está compuesto por 14 módulos (2 administrativos y 12 relacionados con la atención al paciente). Permite realizar registros individuales alrededor de la Historia Clínica Electrónica. Entre los módulos que lo componen está Hospitalización, que gestiona la información que se genera en la atención de las áreas de Urgencia, Enfermería e Ingreso.

Sigho brinda como funcionalidades: listar los pacientes por cada uno de los servicios y áreas, registrar la Historia Clínica del paciente; elaborar notas médicas tales como Ingreso, Evolución, Egreso, Traslado, Interconsulta; registrar Indicaciones médicas (dietas, medicamentos, soluciones, cuidados y control de líquidos); realizar el seguimiento de las indicaciones por parte de enfermería; registrar los diagnósticos mediante una codificación; notificar al médico los diagnósticos asignados cuando no corresponden de acuerdo al sexo y edad del paciente y es capaz de notificar a Epidemiología la existencia de diagnósticos de interés.

Este sistema permite, además, revisar la Historia Clínica Electrónica del paciente que contiene la información de la atención médica precedente y elaborar solicitudes a las distintas áreas. Pueden buscarse las solicitudes de intervenciones quirúrgicas y de procedimientos. También ofrece la funcionalidad del manejo de Referencia y Contrarreferencia de pacientes entre Unidades Médicas. (6)

Otro software que gestiona la información hospitalaria es el *Galen Hospital*. Es un producto cubano que facilita la actualización y registro de codificaciones de productos, contratos con proveedores, procesamiento de pedidos, inventarios, emisión de reportes y la gestión, recepción y entrega de medicamentos. Entre sus facilidades está registrar el aprovisionamiento y entrega de los fármacos

necesarios para los servicios internos y externos del hospital, además de las materias primas para elaborar fórmulas en los dispensarios. Uno de los módulos del *Galen Hospital* es el de Urgencias.

El módulo de Urgencias registra la recepción y atención de los pacientes cuyo estado de salud es grave; mantiene el control de los atendidos y el de los servicios y medicamentos usados; indica cambios en la atención; emite reportes variados y consulta el registro general de pacientes. Este módulo consta de tres programas: Recepción de Cuerpo de Guardia, Hoja de Cargo de Cuerpo de Guardia, Hoja de Cargo Médica de Cuerpo de Guardia.

En la Recepción de Cuerpo de Guardia se realiza el registro de los datos primarios de la consulta del paciente que acude con alguna dolencia o padecimiento. Aquí también se registra la información sobre la hora de atención y el médico que atenderá al paciente. Permite listarse los datos de los pacientes que han sido atendidos en el día, donde se muestran los campos: nombre del paciente, hora de llegada y tiempo de espera. Este programa brinda otras funcionalidades como: establecer los médicos de guardia, agregar consulta, buscar un paciente, pasar a consulta, buscar consultas, eliminar consultas y generar reporte de casos atendidos.

El programa Hoja de Cargo de Cuerpo de Guardia tiene como función completar la información del Cuerpo de Guardia, así como corregir los posibles errores en la entrada de datos en el módulo recepción. Este permite: cambiar fecha, buscar consulta, agregar registro, editar registro y borrar registro.

Por último, el programa Hoja de Cargo Médica de Cuerpo de Guardia acumula las Hojas de Cargo que han sido creadas con los datos de los pacientes que han llegado a la Recepción y las guarda. Para ello debe seleccionarse la fecha deseada.

El *Galen Hospital* versión 2.0, fue desarrollado en plataforma Windows 32 bits (Windows NT y Windows 98) con una configuración Cliente/Servidor y el uso del gestor de bases de datos relacional SQL Server 7.0 como reservorio de la información. (7)

El producto *MediSys®EM* también forma parte de un HIS. Es un sistema informático diseñado y construido para agilizar las actividades realizadas en el área de Emergencias de un establecimiento de salud. Permite registrar la información general del paciente y su clasificación en la Hoja de emergencia.

Los médicos convergen a la estación de trabajo para obtener información de los pacientes, registrar sus notas, diagnósticos provisionales, una Historia Clínica resumida, resultado del examen físico, ordenes de incapacidad, referencias, tratamiento y órdenes al resto de los servicios de apoyo al diagnóstico del establecimiento.

Este sistema permite obtener un reporte de las consultas efectuadas en el servicio, así como las que fueron realizadas por cada médico. Con *MediSys®EM* los médicos, el personal de enfermería y administrativo pueden acceder y actualizar la información del paciente, y enviar las órdenes médicas. Es una solución web que utiliza como gestor de base de datos para plataforma cliente/servidor SQL Server 6.5. (8)

Por último, se encontró el sistema *X-HIS*, solución global que ofrece la última tecnología en gestión clínica y administrativa en un entorno abierto, que permite la integración con otros sistemas de información. Es una herramienta puramente clínica que sitúa al paciente en el centro del sistema.

Tras la identificación del paciente, el usuario del sistema podrá navegar por todo su historial clínico. Está conformado por los módulos: Recursos Humanos, Farmacia, Admisión, Consultas Externas, Bloque Quirúrgico, Anatomía Patológica, Urgencias, entre otros.

A través del módulo Urgencias los usuarios del sistema podrán realizar el registro, control y tratamiento de un paciente en Urgencias, desde que es ingresado hasta que es dado de alta, enlazándose además con los módulos de Admisión y Consultas Externas. Así quedan registrados todos los datos clínicos y administrativos del paciente, además de generar toda la información administrativa como: reportes judiciales, informes clínicos, autorizaciones, entre otros.

Este sistema está desarrollado de acuerdo a los siguientes estándares: CMBD, Codificación ICD-9-CM/ICD 10, SNOMED, HL7, NOC, NANDA, entre otros. Es multiplataforma (Unix, Windows, Linux), tiene soporte para arquitectura cliente/servidor en tres capas. Tiene independencia de bases de datos: MS SQL SERVER, ORACLE, INFORMIX, ADAPTIVE SERVER, SYBASE. (9)

Los HIS anteriormente enunciados se han listado para alinear sus características y establecer una comparación y llegar a conclusiones sobre las tendencias actuales en el desarrollo de software médico.

	Tipo de software	ID de desarrollo	Licencia	Bases de dato	Plataforma	Estándares
Sigho	Web	Microsoft ASP.NET Microsoft Net Framwork	Propietario	SQL Server 2000	Windows	CIE-10 CIE 9CM
Galen Hospital	Desktop	Visual Basic 6.0	Propietario	SQL Server 7.0	Windows	CIE-10
MediSys®EM	Web	-	Propietario	SQL Server 6.5	Windows	CIE-10
X-HIS	Desktop	-	Propietario	independencia de bases de datos: MS SQL SERVER, ORACLE, INFORMIX, ADAPTIVE SERVER, SYBASE	Multi-plataforma	CMBD, Codificación ICD-9-CM/ICD 10, SNOMED, HL7, NOC, NANDA

Tabla 1.1 Comparación de los sistemas existentes que gestionan la información en el área de Emergencias.

Luego de analizar las distintas soluciones o productos existentes en el mercado, que dan solución total o parcial al problema relacionado con la gestión de la información hospitalaria en el área de Emergencias, se arribó a las conclusiones siguientes:

Dichas soluciones en su mayoría presentan licencias de software propietario. Esto trae como consecuencia que, además del costo correspondiente al producto, se debe invertir en el pago de licencias de sistemas gestores de base de datos, servidores y sistema operativo. Si se tiene en cuenta además, que esto debe repetirse por cada estación de trabajo, servidor de aplicaciones o de base de datos que se necesite, y los altos precios de sus licencias, entonces se puede apreciar que se necesita la inversión de grandes sumas de dinero por parte de quien desee informatizar el área de Emergencias de un sistema de salud.

Dichas soluciones no cumplen con la gestión de todos los procesos del área de Emergencias y no se integran con el resto de las áreas hospitalarias que prestan servicios complementarios para la atención del paciente. Por estas razones cabría plantear la necesidad de desarrollar una solución para software libre, que abarque la mayor cantidad de procesos realizados en el área de Emergencias y que manejen la información asociada a la atención al paciente de forma cómoda, fácil y segura.

1.4 Tendencias y tecnologías actuales a considerar.

El desempeño de un proyecto está antecedido siempre por la investigación de las tecnologías de punta que se utilizan, así como las ventajas y desventajas que trae su aplicación. Esta investigación fue realizada, analizando las tecnologías existentes que se ajustan a la solución a desarrollar.

El sistema debe ser accesible desde las más disímiles ubicaciones, debe estar libre de costos adicionales relativos a pago de licencias de software, debe ser adaptable a las reglas del negocio de cualquier institución hospitalaria, así como contar con la debida documentación para su mantenimiento y desarrollo.

1.4.1 Accesibilidad.

Para lograr que el sistema sea accesible desde las más disímiles ubicaciones se propone el desarrollo de una aplicación web. Las aplicaciones web se basan en la arquitectura cliente-servidor. Este modelo consiste básicamente en que un programa (el cliente) realiza peticiones a otro programa (el servidor) que le da respuesta. Los tipos específicos de servidores incluyen los servidores web, los servidores de archivo y los servidores del correo. Mientras que sus propósitos varían de unos servicios a otros, la arquitectura básica seguirá siendo la misma. (10)

Este tipo de aplicación posibilita tener los datos almacenados centralmente en bases de datos, accesibles desde un navegador web o una terminal móvil, gracias a que la lógica se ejecuta en el servidor y el diseño del interfaz es transferido a dichas terminales. Los requerimientos de hardware para estas aplicaciones solo se limitan a contar con un servidor web potente y una conectividad permanente y relativamente rápida.

La instalación y despliegue de las mismas es muy sencilla, pues solo necesita montarse el servidor de aplicaciones y este será accedido por los clientes, independientemente de la plataforma, la arquitectura de máquina y la localización física que tengan dichas PCs. El mantenimiento de estas aplicaciones es también muy sencillo, pues los cambios son publicados en el servidor y los clientes acceden a estos, por tanto no hay necesidad de tener que ir máquina por máquina instalando las actualizaciones requeridas.

Para el desarrollo de la aplicación web se propone la utilización del patrón de diseño Modelo-Vista-Controlador y el Patrón en capas. Un patrón es un modelo que se puede seguir para realizar algo. Los patrones surgen de la experiencia de seres humanos de tratar de lograr ciertos objetivos. Ellos capturan la experiencia existente y probada para promover buenas prácticas.

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Modelo Vista Controlador (MVC), es un patrón de diseño de software, que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

El patrón en capas es un estilo de programación cuyo objetivo primordial es la separación y agrupamiento de los componentes del software atendiendo a la función que cumplen en el mismo. Para realizar el agrupamiento se tiene en cuenta las funcionalidades relacionadas con el usuario del sistema, así como la información que este maneja y las operaciones que realiza sobre la misma en dependencia de la complejidad que se necesita que tenga el sistema. Esta división muchas veces se hace en tres capas: la capa de presentación, capa de negocio y la capa de datos. (11)

1.4.2 Mínimo costo.

Otra característica necesaria del sistema es que debe estar libre del costo relacionado con patentes de software**,** asociadas al servidor de aplicaciones, servidor de base de datos, sistema operativo huésped u otras herramientas o tecnologías utilizadas para su desarrollo.

Para lograr este objetivo se propone el uso de un lenguaje de programación multiplataforma, como es el caso de Java. Este es orientado a objetos. El lenguaje en sí mismo toma mucha de su sintaxis de otros como C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria.

Para el desarrollo de aplicaciones web, Java se volvió más popular a partir de la aparición de la especificación de Servlets y JSP (Java Server Pages). Los servlets y las JSPs supusieron un importante avance ya que el API (Interfaz de Programación de Aplicaciones) de programación es muy sencillo, flexible y extensible. (12)

Además se propone la utilización de tecnologías y herramientas que permitan su uso sin necesidad del pago por su licencia. Las tecnologías aparecerán relacionadas a continuación según su ubicación en las capas de presentación, negocio y acceso a datos, separadas las que no estén ubicadas en ninguna de estas capas, así como una relación de las herramientas propuestas.

1.4.2.1 Capa de presentación.

La capa de presentación es la que presenta 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. (13)

1.4.2.1.1 Java Server Faces (JSF).

JSF, es un framework que facilita y agiliza el diseño de interfaces de usuario, pues implementa una serie de componentes, estado de los mismos, eventos del lado de servidor, entre otras ventajas. (12)

1.4.2.1.2 RichFaces.

Rich Faces**:** es un framework de código abierto que añade capacidad Ajax dentro de aplicaciones JSF existentes sin recurrir a JavaScript. Rich Faces incluye ciclo de vida, validaciones, conversiones y la gestión de recursos estáticos y dinámicos. Los componentes de Rich Faces están construidos con soporte Ajax que puede ser fácilmente incorporado dentro de las aplicaciones JSF. (14)

- **Ajax.**

En la interfaz de usuario en aplicaciones web se ha hecho muy común el uso de Ajax con el objetivo de lograr aplicaciones más amigables y rápidas. Ajax es el acrónimo para Asynchronous JavaScript + XML. Este conjunto de tecnologías permite que al ser necesario que desde una página web se ejecute una acción en el servidor, esta se realice de forma asíncrona y se busquen los datos que son usados para actualizar la página mostrándose u ocultándose porciones de la misma sin necesidad de que sea recargada toda la pagina nuevamente. (15)

1.4.2.1.3 Ajax4JSF.

Ajax4jsf es una librería de código abierto que se integra totalmente en la arquitectura de JSF y extiende la funcionalidad de sus etiquetas dotándolas con tecnología Ajax de forma limpia y sin añadir código Javascript. Mediante este framework se puede variar el ciclo de vida de una petición JSF, recargar determinados componentes de la página sin necesidad de recargarla por completo, realizar peticiones automáticas al servidor y controlar cualquier evento de usuario. (16)

1.4.2.1.4 Facelets.

JavaServer Facelets es un framework para plantillas centrado en la tecnología JSF (JavaServer Faces), lo cual permite que JSP (JavaServer Pages) y JSF (JavaServer Faces) puedan funcionar conjuntamente en una misma aplicación web. Estos no se complementan naturalmente. JSP procesa los elementos de la página de arriba a abajo, mientras que JSF dicta su propio re-rendering (ya que su ciclo de vida está dividido en fases marcadas). Facelets llena este vacío entre JSP y JSF, siendo una tecnología centrada en crear árboles de componentes y estar relacionado con el complejo ciclo de vida JSF. (17)

1.4.2.1.5 XHTML.

XHTML, acrónimo inglés de eXtensible Hypertext Markup Language (lenguaje extensible de marcado de hipertexto), es el lenguaje de marcado pensado para sustituir a HTML como estándar para las páginas web. Su objetivo es avanzar en el proyecto del World Wide Web Consortium de lograr una web semántica, donde la información y la forma de presentarla estén claramente separadas. (18)

1.4.2.1.6 Cascading Style Sheets (CSS).

Las hojas de estilo en cascada son un lenguaje formal usado para definir la presentación de un documento estructurado escrito en HTML o XML (y por extensión en XHTML). El W3C (World Wide Web Consortium) es el encargado de formular la especificación de las hojas de estilo que servirán de estándar para los agentes de usuario o navegadores. Lo que se persigue con el desarrollo de CSS es separar la estructura de un documento de su presentación. (19)

1.4.2.2 Capa de negocio.

La capa de negocio es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del 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 almacenar o recuperar los mismos. (20)

1.4.2.2.1 JBoss Seam.

JBoss Seam es un framework que integra la capa de presentación (JSF) con la capa de negocios y persistencia (EJB), funcionando, según versa su significado en español, como una “costura” entre estos componentes. Seam también se integra perfectamente con otros frameworks como: RichFaces, ICEFaces, MyFaces, Hibernate y Spring. (21)

1.4.2.2.2 Drools.

Drools es una implementación del JSR 94 (Java Rule Engine API), una especificación que define una interfaz común para un motor de reglas estándar dentro de la plataforma Java. Para definir las reglas emplea XML y permite adaptarse a la semántica de un determinado dominio definiendo un esquema que la represente. Su licencia es BSD (*Berkeley Software Distribution*) y, poco después de la liberación de la versión 2.0, se unió a la compañía JBoss, la cual ofrece servicios de consultoría, formación y soporte sobre el producto (al cual denomina "JBoss Rules").

Esta tecnología permitirá lograr otra importante característica que debe cumplir el sistema: ser configurable y adaptable a los procesos en el área de Emergencias de cualquier institución hospitalaria. (22)

1.4.2.3 Capa de acceso a datos.

La capa de acceso a datos contiene clases que interactúan con la base de datos, estas clases altamente especializadas permiten, utilizando los procedimientos almacenados (funciones para interactuar con la base de datos) que se generan, realizar todas las operaciones con la base de datos de forma transparente para la capa de negocio. (23)

1.4.2.3.1 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. Utiliza para esto archivos declarativos (XML) que permiten establecer estas relaciones. Es una tecnología de software libre distribuida bajo los términos de la licencia GNU LGPL.

Como todas las herramientas de su tipo, Hibernate busca solucionar el problema de la diferencia entre los dos modelos de datos coexistentes en una aplicación: el usado en la memoria de la computadora orientación a objetos) y el usado en las bases de datos (modelo relacional). Le permite a la aplicación manipular los datos de la base de datos operando sobre objetos, con todas las características de la

programación orientada a objetos. Hibernate convertirá los datos entre los tipos utilizados por Java y los definidos por SQL.

Esta herramienta genera las sentencias SQL y libera al desarrollador del manejo manual de los datos que resultan de la ejecución de dichas sentencias. Logra mantener la portabilidad entre todos los motores de bases de datos con un ligero incremento en el tiempo de ejecución. Hibernate ofrece también un lenguaje de consulta de datos llamado HQL (Hibernate Query Language), al mismo tiempo que una API para construir las consultas programáticamente conocida como "Criteria". (24)

1.4.2.3.2 EJB3.

Los Enterprise JavaBeans (también conocidos por sus siglas EJB) son una de las Interfaces de Programación de Aplicaciones, cuyo acrónimo en inglés es API *(Application Programming Interface)*. Estas forman parte del estándar de construcción de aplicaciones empresariales J2EE de Sun Microsystems (ahora JEE 5.0). Su especificación detalla cómo los servidores de aplicaciones proveen objetos desde el lado del servidor que son, precisamente, los EJB.

Los EJB proporcionan un modelo distribuido y estándar de componentes que se ejecutan en el servidor. El objetivo de los EJB es dotar al programador de un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (concurrencia, transacciones, persistencia, seguridad, etc.), para centrarse en el desarrollo de la lógica de negocio en sí. El hecho de estar basado en componentes permite que éstos sean flexibles y sobre todo reutilizables. (25)

1.4.2.3.3 JPA.

Java Persistence API, más conocida por su sigla 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 POJOs (*Plain Old Java Object*) (26)

1.4.2.4 Tecnologías horizontales.

Existen un conjunto de tecnologías que se extienden horizontalmente por todas las capas antes mencionadas y sirven de soporte a las tecnologías que se utilizan en cada una de ellas. Las mismas se describen a continuación:

JavaEE 5.

Java Platform Enterprise Edition o Java versión 5 es una plataforma de programación (parte de la Plataforma Java) para desarrollar y ejecutar software de aplicaciones en lenguaje de programación Java con arquitectura de N niveles distribuida. Se basa ampliamente en componentes de software modulares y se ejecuta sobre un servidor de aplicaciones. (27)

JRE 6.

JRE es el acrónimo de Java Runtime Environment (entorno en tiempo de ejecución Java) y se corresponde con un conjunto de utilidades que permite la ejecución de programas java sobre todas las plataformas soportadas. JVM (máquina virtual Java) es una instancia de JRE en tiempo de ejecución. Este interpreta el código Java y está compuesto además por las librerías de clases estándar que implementan el API de Java. Ambas JVM y API deben ser consistentes entre sí, de ahí que sean distribuidas de modo conjunto. (28)

1.4.2.5 Herramientas.

1.4.2.5.1 Eclipse.

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma típicamente ha sido usada para desarrollar entornos de desarrollo integrados (del inglés IDE), como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se entrega como parte del Eclipse.

El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la plataforma de los llamados clientes ricos. Esto lo diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no. Este mecanismo de módulos es una plataforma ligera para componentes de software, que adicionalmente permite al Eclipse extenderse, usando otros lenguajes de programación como C/C++, Python y Java. (29)

1.4.2.5.2 JBoss Tools.

Es un conjunto de plug-in para el Eclipse que permite el manejo de diferentes frameworks que facilitan el desarrollo de aplicaciones. Está constituido por varios módulos: RichFaces VE, Seam Tools, Hibernate Tools y JBoss AS Tools. (30)

1.4.2.5.3 JBoss Server.

JBoss Application Server es el servidor de aplicaciones de código abierto más ampliamente desarrollado del mercado. Por ser una plataforma certificada J2EE, soporta todas las funcionalidades de J2EE 1.4 e incluye servicios adicionales como clustering, caching y persistencia. JBoss es ideal para aplicaciones Java y aplicaciones basadas en la web. También soporta Enterprise Java Beans (EJB) 3.0, lo que hace el desarrollo de las aplicaciones mucho más simple. Además, al ser desarrollado con tecnología Java, es multiplataforma. (31)

1.4.2.5.4 Sistemas de Gestión de Base de Datos.

Los Sistemas de Gestión de Base de Datos (SGBD); (en inglés: Data Base Management System, siglas DBMS) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. El propósito general de los sistemas de gestión de base de datos es manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante. Entre los gestores de base de datos se destacan MySQL, Microsoft SQL-Server, POSTGRES, Oracle y PostgresSQL. (32)

1.4.2.5.4.1 PostgreSQL Server 8.3.

PostgreSQL es un Sistema de Gestión de Bases de Datos Objeto-Relacionales (Object-Relational Database Management System (ORDBMS)) libre, no tiene costo asociado por lo que cualquiera puede disponer de su código fuente, modificarlo a voluntad y redistribuirlo libremente. PostgreSQL presenta alta concurrencia, para esto utiliza la tecnología de Control de Concurrencia Multi-Versión (Multiversion concurrency control (MVCC)), con lo que se logra que ningún lector sea bloqueado por un escritor. Es altamente extensible, soporta operadores, funciones, métodos de acceso y tipos de datos definidos por el usuario.

Tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/PGSQL. Este lenguaje es comparable al lenguaje procedural del sistema de gestión de base de datos relacional Oracle, PL/SQL. En cuanto a sus funciones, poseen bloques de código que se ejecutan en el servidor los cuales pueden ser escritos en varios lenguajes, con la potencia que cada uno de ellos brinda. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que se intente conectar a la base de datos. Tiene una excelente documentación y está bien organizada, además de contar con una comunidad de usuarios y desarrolladores a los que acudir en caso de tener problemas. (33)

1.4.2.5.5 Visual Paradigm para UML.

“Visual Paradigm para UML (Lenguaje Unificado de Modelado) es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML”. (34)

1.4.3 Documentación.

Para lograr esta característica se dispuso como metodología de desarrollo El Proceso Unificado de Desarrollo (RUP)**.** Como lenguaje de modelado se propone el Lenguaje Unificado de Modelado (UML) y como notación para la descripción de los procesos del negocio a informatizar, la Notación para el Modelado de Procesos de Negocio (BPMN).

1.4.3.1 Proceso unificado de desarrollo.

RUP es el resultado de varios años de trabajo y uso práctico en el que se han unificado técnicas de desarrollo, a través del UML, y trabajo de muchas metodologías utilizadas por los clientes. En RUP se han agrupado las actividades en grupos lógicos en los que se definen nueve flujos de trabajo principales. Los seis primeros son conocidos como flujos de ingeniería y los tres últimos como flujos de apoyo. El ciclo de vida de RUP se caracteriza por ser dirigido por caso de uso, centrado en la arquitectura, iterativo e incremental. (35)

1.4.3.2 Lenguaje Unificado de Modelado.

UML es un lenguaje para visualizar, especificar, construir y documentar los artefactos de un sistema que involucra una gran cantidad de software. Permite la modelación de sistemas con tecnología orientada a objetos. Se puede aplicar en el desarrollo de software entregando gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como RUP), pero no especifica en sí mismo qué metodología o proceso utilizar.

Este lenguaje de modelado formal permite tener un mayor rigor en la especificación, realizar una verificación y validación del modelo desarrollado, automatizar determinados procesos y generar código a partir de los modelos y a la inversa. Esto último permite que el modelo y el código estén actualizados. (36)

1.4.3.3 Notación para Gestión de Procesos de Negocio.

BPMN (Business Process Management Notation) es un nuevo estándar de modelado de procesos de negocio donde se presentan gráficamente las diferentes etapas de su proceso. La notación ha sido

diseñada específicamente para coordinar la secuencia de procesos y los mensajes que fluyen entre los diferentes procesos participantes.

Antes de conocer para qué se utiliza esta notación, es necesario resaltar en qué consiste un proceso de negocio. Este es una colección de actividades que, tomando una o varias clases de entradas, crean una salida que tiene valor para un cliente. Los Procesos de Negocio representan el flujo de trabajo y de información a través del negocio.

El objetivo principal de esta notación es mejorar la eficiencia a través de la gestión sistemática de los procesos de negocio que se deben modelar, automatizar, integrar, monitorizar y optimizar de forma continua. A través del modelado de las actividades y los procesos puede lograrse un mejor entendimiento del negocio. Muchas veces esto brinda un mejor enfoque, lo que permite mejorarlos. (37)

Como resultado del estudio realizado en este capítulo, se puede concluir que los sistemas existentes que gestionan la información del área de Emergencias en las instituciones hospitalarias no cumplen con los requisitos necesarios. Por lo que se evidenció la necesidad de desarrollar un sistema en el que el usuario tenga una mayor accesibilidad, sea adaptable a diferentes instituciones hospitalarias, que además este debidamente documentado para desarrollos futuros del mismo y este desarrollado con tecnologías y herramientas no privativas. Se justificó la aplicación de los patrones de diseño y el empleo de las tecnologías y herramientas propuestas para la obtención del sistema.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA.

En este capítulo se describen los procesos actuales Recibir paciente y Atender paciente referentes a las instituciones hospitalarias que constan de área de Emergencias. En aras de tener una representación gráfica de esos procesos se muestra el modelo de negocio, así como los actores involucrados. También se especifican los requisitos funcionales y no funcionales del sistema y los casos de uso del sistema a desarrollar como respuesta a la situación problémica actual.

2.1 Flujo actual de los procesos involucrados en el campo de acción.

Por la importancia que tiene la atención a tiempo del paciente cuando llega al área de Emergencias se estableció como campo de acción los procesos de recibir al paciente y darle la primera atención emergente que requiere. A continuación se describen los procesos:

Recibir paciente.

El área de Emergencias está concebida para recibir pacientes que requieran atención médica urgente, quienes son atendidos por médicos emergenciólogos preparados para asistir estos casos. Si la atención que requiere no es urgente, el paciente será remitido a una consulta especializada. Dada la urgencia que requiera la atención del paciente ha de clasificarse su estado de salud según el estándar de colores en: Roja, Amarilla o Verde. Esto determinará su prioridad respecto al resto de las personas que están en espera.

Es frecuente que lleguen casos de pacientes accidentados e inconscientes. Si estos no portan sus documentos y no vienen acompañados de alguna persona que los pueda identificar, se les toman las huellas dactilares y se hace una descripción física del individuo, además de definir su sexo. La información del desconocido se le entrega al departamento de vigilancia para que proceda a la investigación del caso.

Para llevar un control de los pacientes que llegan a área se lleva un Registro de Pacientes Atendidos. En él se incorporan los datos de identificación del paciente tales como: nombre, apellidos y el número de

identificación; además del motivo de consulta y hora de llegada. Aquellos que no poseen identificación, se les registra una descripción física y el sexo, además se le asigna un número consecutivo para no cometer errores de identificación.

A todos los pacientes atendidos se les ha de crear una Hoja de emergencia para registrar en ella toda la información asociada al caso médico. A esta se deben agregar también los datos de identificación del paciente. Si el departamento de vigilancia identifica a uno de los pacientes desconocidos, se modificaría la Hoja de emergencia escribiéndole los datos personales del paciente.

Dado el gran número de casos accidentados y que presentan lesiones por arma blanca o de fuego se hace preciso crear un registro para posteriores investigaciones. Este registro se llama Libro de casos médico legal y contiene la información asociada al evento.

Emergencias puede recibir casos referidos de otras instituciones hospitalarias. A esos casos se les recepciona la referencia para que quede registrado que se le dio respuesta a la solicitud.

Los pacientes cuya gravedad fue clasificada con el color Rojo, se han de conducir a un área especializada según su dolencia. Estas pueden ser: Terapia respiratoria, Trauma shock, Quirófano de Emergencias u otra área que exista en el centro. Si el estado del paciente es estable y por tanto fue clasificado con el color Amarillo o Verde, este será atendido en una sala que comúnmente es denominada Triaje.

En cualquiera de las áreas que sea ubicado el paciente se le dará la atención requerida. Dicha atención se describe en el proceso Atender paciente que aparece descrito a continuación:

Atender paciente.

La atención médica es el proceso más importante dentro del área de Emergencias dada su función de salvaguardar la vida del paciente. Muchas veces este llega descompensado y la primera acción que se realiza es medir sus signos vitales. También se les hace un reconocimiento mediante el examen físico que permite identificar los órganos o sistemas de órganos lesionados y describir los hallazgos positivos encontrados en ellos. Si necesita ser transfundido se solicita una transfusión.

Cuando el paciente está consciente y en condiciones de contestar las preguntas del médico, se le realiza un interrogatorio. Esta actividad aporta datos importantes como: los medicamentos habituales que este consume, las enfermedades que padece, así como sus antecedentes patológicos. En este punto de la atención el médico ha de tener una impresión diagnóstica del caso.

Durante la atención médica puede requerirse la realización de exámenes complementarios que ayuden a la conclusión de un diagnóstico. Dichos exámenes tienen prioridad por ser del área de Emergencias, por lo que su resultado ha de estar en breve tiempo. Cuando se obtengan los resultados, el médico emite un diagnóstico y este ha de ser revisado por la unidad de vigilancia epidemiológica, para comprobar si es una enfermedad de notificación obligatoria.

Dada la situación de salud del paciente ha de tomarse una conducta a seguir que puede ser: solicitar Interconsulta, dejar al paciente en Observación, hospitalizarlo, egresarlo o registrar su fallecimiento en caso de muerte.

La solicitud de interconsulta a una especialidad se realiza en caso que al médico le quede alguna duda respecto al diagnóstico realizado. Una vez hecha la solicitud de interconsulta, acudirá al área de Emergencias un especialista que evaluará al paciente siguiendo el proceso antes descrito.

El diagnóstico puede variar respecto al emitido por el médico tratante, lo que los llevará a ambos a un debate médico respecto al caso. Puede ocurrir que el médico interconsultante determine que la patología del paciente no tiene que ver con su especialidad por lo que debe realizarse la interconsulta con otro especialista.

Si no se logra compensar al paciente, este debe quedarse en la sala de Observación bajo la supervisión de enfermería. Esta atención se le brindará durante 72 horas como máximo. Si en ese tiempo el paciente aún no mejora deberá ser hospitalizado en una sala del servicio al que corresponda su patología. De ser un paciente que fue referido al área de Emergencias desde otra institución hospitalaria, se le realiza una contrareferencia para que continúe hospitalizado en esta.

Una vez que el paciente se encuentre estable, el médico puede determinar que debe ser egresado de Emergencias. Para ello se le ha de dejar algún tratamiento con las indicaciones a seguir. También puede

indicársele una referencia a la consulta externa para que continúe el seguimiento de su patología. Si los familiares del paciente desean llevárselo a otro hospital sin importarle la opinión del médico, este se verá obligado a realizar el egreso y notificará que fue en contra de su opinión.

Cuando los pacientes fallecen se le solicita el Certificado de defunción a la enfermera. En él se registran los datos del caso y se entrega a los familiares. Si es de interés médico o los familiares lo desean, se solicita la autopsia del cadáver a Anatomía Patológica para profundizar en las causas de la muerte.

2.2 Objeto de automatización.

Una vez descrito y analizado el funcionamiento del negocio se definen los procesos a automatizar y se realiza un análisis donde se depuran las actividades manuales, y queden solo aquellas que pasen a ser funcionalidades del sistema. Con el desarrollo de un Sistema de Información Hospitalaria que abarque el área de Emergencias, estos procesos serán automatizados, lo que proporciona mayor eficacia en el trabajo realizado en esta área del hospital.

El primer objeto a automatizar es la Hoja de emergencia que será asociada a la Historia Clínica del paciente, como elemento contenedor de toda su atención médica en el transcurso de su vida. Esto posibilitará realizar búsquedas sobre las Historias Clínicas lo que proporciona un fácil y rápido acceso a la información médica del paciente.

También se contemplan las actividades de creación de Historia Clínica para aquellos pacientes que no la posean, así como una Historia Clínica temporal para los pacientes que llegan inconscientes y sin documentos personales que posibiliten su identificación.

A cada paciente que llega a Emergencias se le crea una Hoja de emergencia, por lo que si se listan estas se podrá obtener la relación de pacientes atendidos, mostrándose los campos más importantes como el nivel de gravedad del paciente, sus datos de identificación, motivo de la consulta, entre otros.

Mediante la asociación de la Hoja de emergencia a la Historia Clínica se logra visualizar automáticamente información valiosa para la atención urgente del paciente, ejemplo de ello lo constituye: los antecedentes personales, familiares, hábitos psicobiológicos y las inmunizaciones. Esto le permite al médico tener una

impresión diagnóstica del caso. Otros datos que ayudan a la decisión del diagnóstico son: el examen físico y el interrogatorio. Estos aportan la información de los medicamentos habituales que el paciente ingiere y la enfermedad que padece, así como los órganos o sistemas de órganos que tiene afectado.

Las solicitudes de los exámenes complementarios también serán informatizadas por lo que el médico podrá solicitar los exámenes complementarios de análisis de laboratorio y solicitudes de estudios radiológicos e imagenológicos desde el sistema, en tiempo real. También podrá solicitar las transfusiones de sangre en caso que el paciente lo necesite y describirá cómo transcurrió el proceso para que quede un historial de las reacciones del paciente.

Otra funcionalidad a automatizar es solicitar una interconsulta con un especialista. Esto brinda la posibilidad de tener un nuevo diagnóstico, por lo que se crea una Hoja de interconsulta donde se registra el criterio del médico interconsultante.

En el proceso manual existen conductas a seguir con el paciente como: Egresar, Hospitalizar, Dejar en Observación o Registrar el fallecimiento del paciente. Estas actividades serán automatizadas para tener un mejor control del seguimiento del paciente ya que conforman las salidas del proceso de atención médica.

Cuando se toma la decisión de egresar al paciente ha de creársele una indicación médica y si esta incluye algún medicamento ha de crearse una receta. El sistema permitirá seleccionar los medicamentos. También debe brindarse la funcionalidad de referir a los pacientes a otro centro hospitalario o a la consulta externa del propio hospital mediante una referencia o una contrarreferencia. Esta última ocurrirá en caso que el paciente haya llegado referido de otro hospital a Emergencias.

Si es preciso dejar al paciente en Observación, se le crea una Orden médica para que la enfermera pueda guiarse en la atención que debe prestarle, también de ser necesario una alimentación especial se le crea una Orden de dieta.

Para tomar la decisión de hospitalizar al paciente se consulta primeramente la disponibilidad de camas en el servicio que se admitirá. De haber cama disponible para el paciente, se le crea la Orden de admisión seleccionando un cupo.

Cuando un paciente fallece durante la atención o ya estaba muerto al llegar a Emergencias es necesario registrar el fallecimiento. Se debe crear el Certificado de defunción y opcionalmente se solicita la realización de la autopsia al paciente.

2.3 Modelado del Negocio.

Para realizar el modelado del negocio por procesos se siguió la notación BPMN. Esto aporta una mayor visibilidad de las actividades que se realizan y ayuda a lograr un mejor entendimiento del flujo de trabajo existente entre las áreas hospitalarias, lo que permite definir con claridad las actividades innecesarias a la hora de automatizar el negocio.

Mediante los elementos gráficos que la notación BPMN define se pueden construir los diagramas de procesos. Para un mejor entendimiento es preciso describir los elementos gráficos empleados:

Existen eventos de inicio y fin, los cuales indican una acción lógica en el flujo del proceso. Se identifican subprocesos y dentro de ellos se encuentran actividades atómicas que se relacionan por una línea continua que representa un flujo de secuencia. Las bifurcaciones determinan la ramificación del flujo a través de decisiones inclusivas o exclusivas. Los artefactos brindan la información de los documentos o registros que se requieren y se producen en cada actividad. Estos se relacionan mediante líneas discontinuas con la actividad que lo emplea.

Los diagramas están compuestos por calles, que representan a los actores y trabajadores del negocio; donde el mayor cúmulo de actividades está en las calles de los trabajadores. En estas se muestran las actividades que realizan en orden lógico, respondiendo al proceso de negocio descrito.

A continuación se representa el diagrama de procesos de negocio:

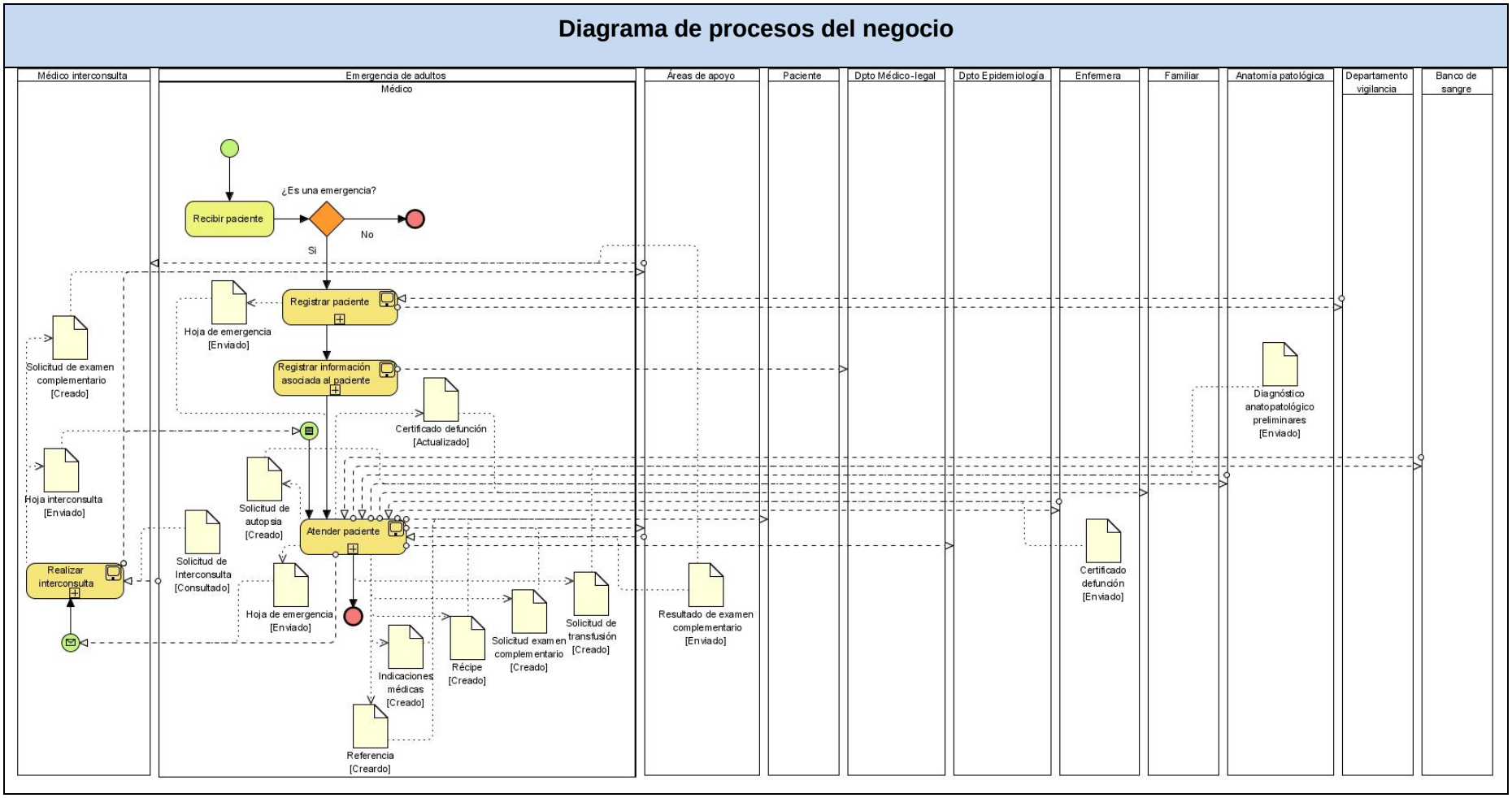


Figura 2.1 Diagrama de procesos del negocio.

En este diagrama se visualizan los procesos que constituyen el objeto de estudio del trabajo de diploma. Estos son modelados puntualmente en un diagrama de procesos del negocio que describe el flujo de actividades que se realizan en ellos.

Una vez presentados los diagramas de procesos se describen los principales actores involucrados representados en los mismos:

Roles	Funciones
Médico de Emergencias.	Es el responsable de registrar los datos del paciente en el Registro de Pacientes Atendidos, así como brindarle la atención médica necesaria y registrarla en la Hoja de emergencia.
Técnico(a) de registros y estadísticas de salud.	Es la persona que tiene la responsabilidad de crear la Hoja de emergencia del paciente que es atendido en el Servicio.
Médico interconsultante.	Es el médico que brinda una consulta especializada en caso de ser llamado para dar una interconsulta. Este emite su criterio sobre el caso y el Médico de la emergencia valora su diagnóstico para llegar a una conclusión del caso.

Tabla 2.1 Trabajadores del negocio.

2.3.1 Diagrama de actividades por procesos.

Proceso: Registrar paciente.

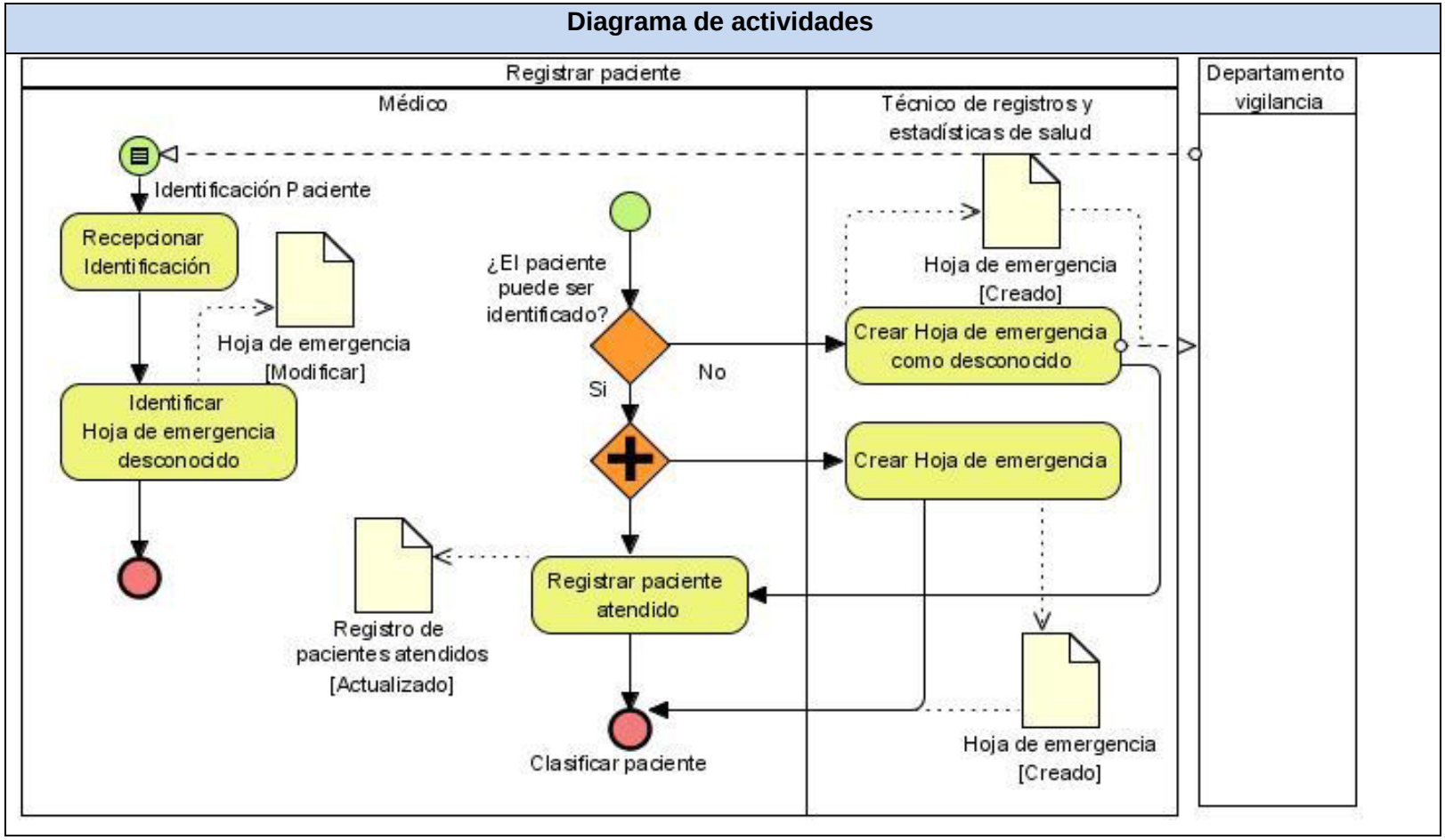


Figura 2.2 Diagrama de actividades. Registrar paciente.

Proceso: Clasificar paciente.

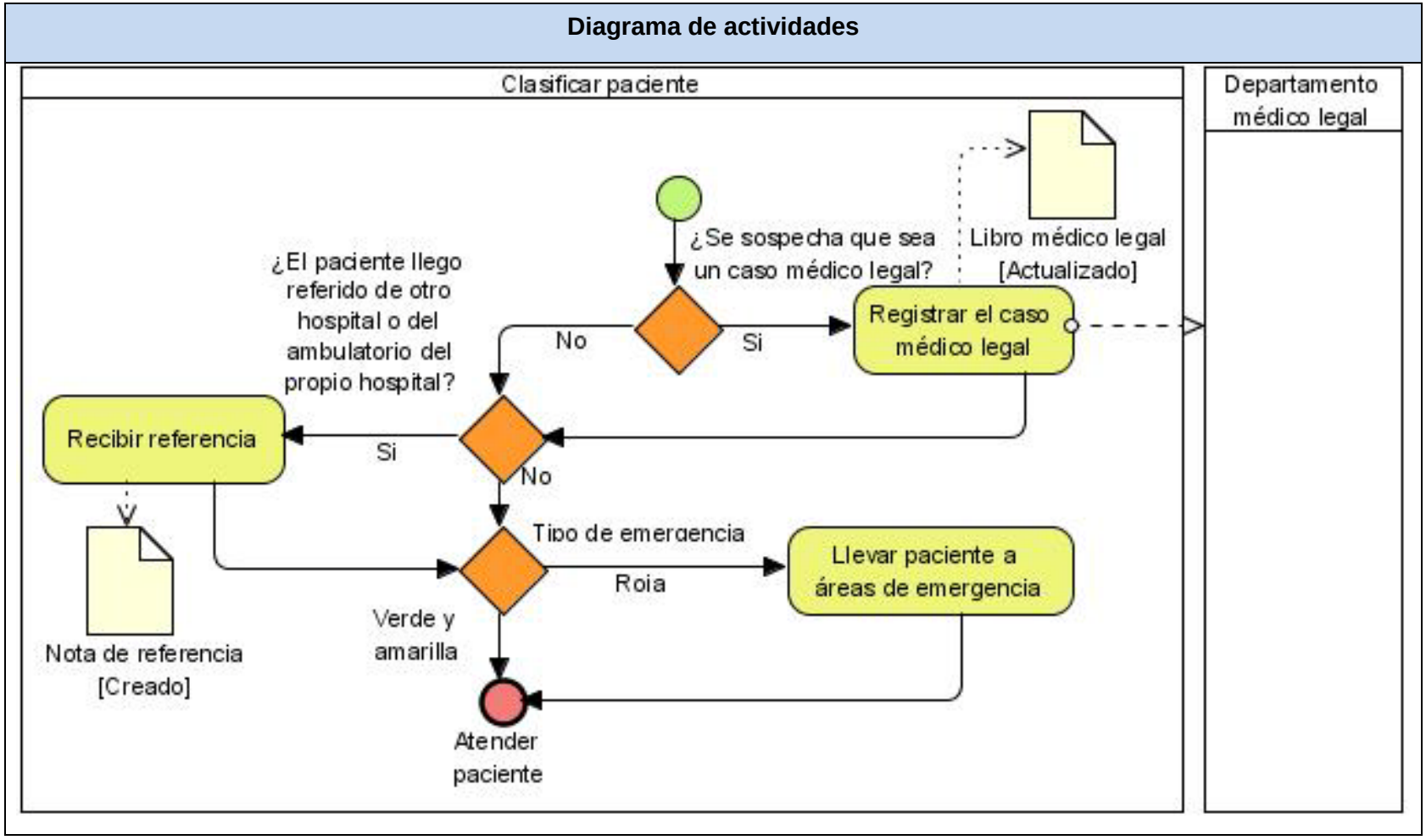


Figura 2.3 Diagrama de actividades. Clasificar paciente.

Proceso: Atender paciente.

Diagrama de actividades

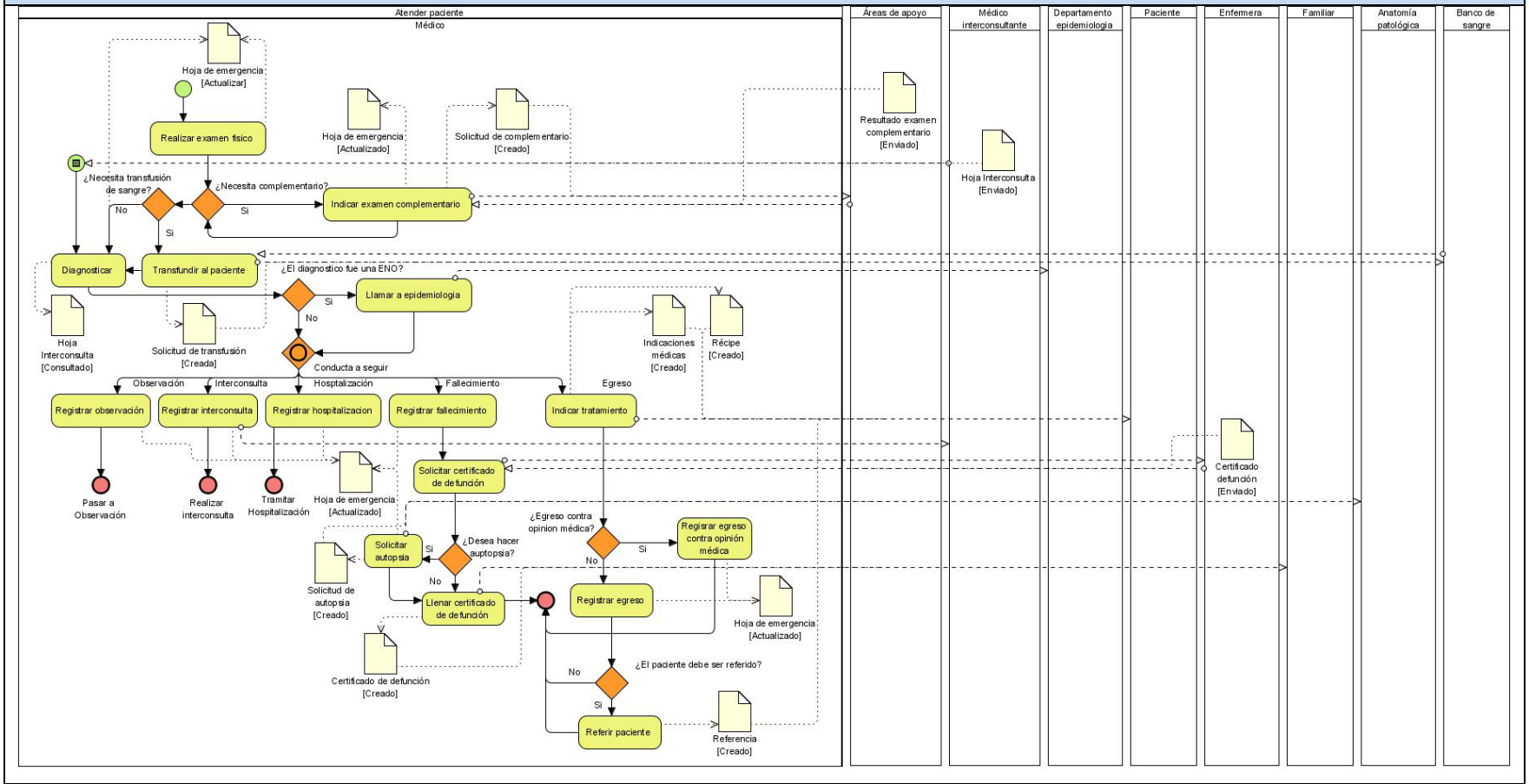


Figura 2.4 Diagrama de actividades. Atender paciente.

Proceso: Realizar interconsulta.

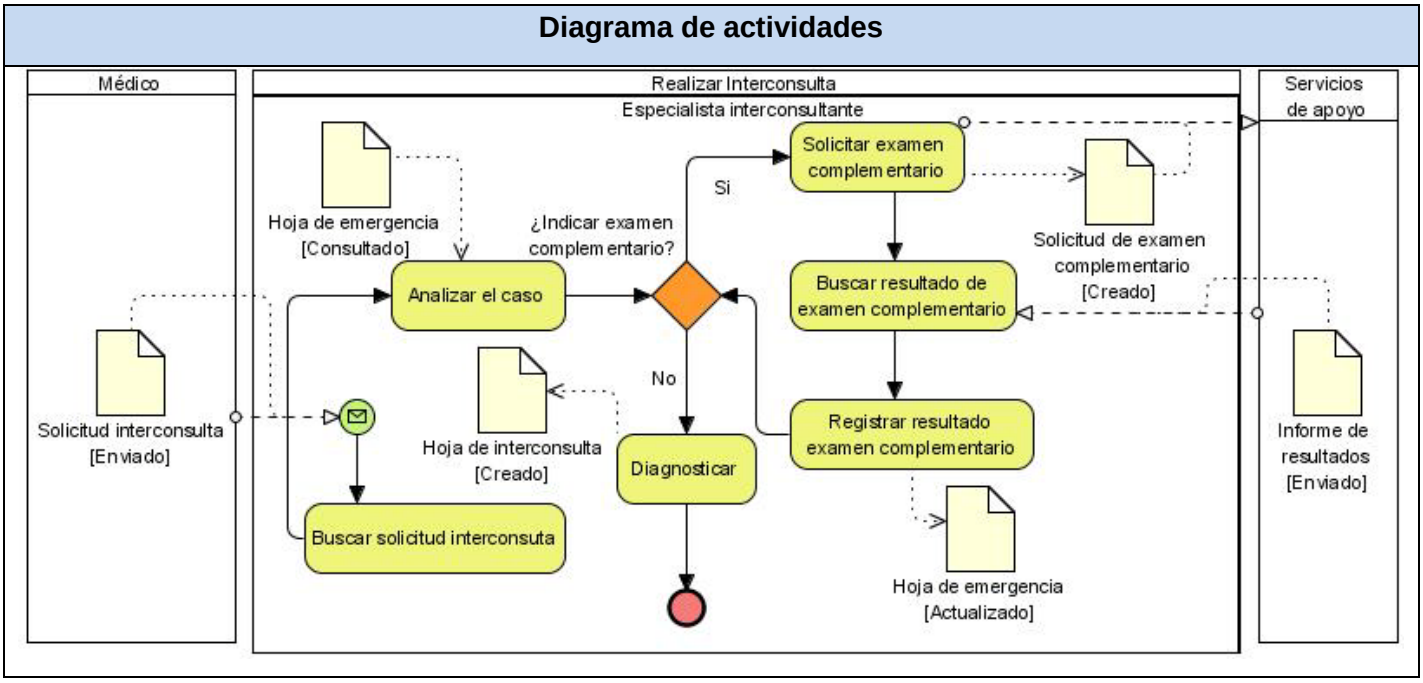


Figura 2.5 Diagrama de actividades. Realizar interconsulta.

2.4 Especificación de los requerimientos del software.

2.4.1 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.

1. Seleccionar Historia Clínica Electrónica.
2. Crear Hoja de emergencia.
3. Crear Hoja de emergencia a paciente desconocido.
4. Ver detalle de la información del la recepción del paciente.

5. Modificar Información de la recepción del paciente.
6. Ver Libro de emergencia alternativo.
7. Ver Información de la recepción del paciente.
8. Registrar caso médico legal.
9. Ver caso médico legal.
10. Modificar caso médico legal.
11. Eliminar caso médico legal.
12. Ubicar paciente en las áreas de emergencia.
13. Seleccionar Hoja de emergencia.
14. Registrar atención al paciente.
15. Modificar Hoja de emergencia.
16. Consultar acciones realizadas en la atención al paciente.
17. Crear solicitud de análisis de laboratorio.
18. Modificar Solicitud de Análisis de laboratorio.
19. Eliminar Solicitud de Análisis de laboratorio.
20. Ver datos de solicitud de análisis de laboratorio.
21. Ver datos de Informe de resultados de laboratorio.
22. Ver datos de solicitud de transfusión.
23. Crear solicitud de transfusión.
24. Registrar datos de la transfusión.
25. Modificar solicitud de transfusión.
26. Seleccionar solicitud de transfusión.
27. Asignar cita para estudio radiológico e imagenológico.
28. Buscar cupo para cita.
29. Ver datos de cita para estudio radiológico e imagenológico.
30. Modificar datos de cita para estudio radiológico e imagenológico.
31. Eliminar cita para estudio radiológico e imagenológico.
32. Crear Orden de Admisión.
33. Consultar disponibilidad de cama.
34. Eliminar Orden de Admisión.

35. Ver Orden de Admisión.
36. Modificar Orden de Admisión.
37. Eliminar solicitud de autopsia.
38. Crear solicitud autopsia.
39. Ver solicitud de autopsia.
40. Modificar solicitud de autopsia.
41. Ingresar en Observación.
42. Ver ingreso a Observación.
43. Modificar ingreso a Observación.
44. Eliminar ingreso a Observación.
45. Registrar Egreso.
46. Ver Egreso.
47. Eliminar Egreso.
48. Modificar egreso.
49. Consultar resultado de solicitud de estudio radiológico e imagenológico.
50. Dibujar órganos o sistemas de órganos afectados.
51. Localizar lesiones gráficamente.
52. Crear solicitud de interconsulta.
53. Modificar solicitud de interconsulta.
54. Eliminar solicitud de interconsulta.
55. Ver datos de solicitud de interconsulta.
56. Buscar Hoja de interconsulta.
57. Anular Certificado de defunción.
58. Crear Certificado de defunción.
59. Ver Certificado de defunción.
60. Ver datos de contrarreferencia.
61. Modificar datos de contrarreferencia.
62. Crear contrarreferencia.
63. Eliminar datos de contrarreferencia.
64. Crear indicaciones médicas.

65. Eliminar indicaciones médicas.
66. Modificar indicaciones médicas.
67. Ver datos de indicación médica.
68. Crear referencia.
69. Modificar referencia.
70. Eliminar referencia.
71. Ver datos de una referencia.
72. Seleccionar tratamiento.
73. Ver tratamiento.
74. Eliminar tratamiento.
75. Modificar tratamiento.
76. Ver datos de una constancia.
77. Modificar constancia.
78. Emitir constancia.
79. Eliminar constancia.
80. Actualizar inmunizaciones.
81. Actualizar antecedentes familiares.
82. Modificar hábitos psicobiológicos.
83. Modificar antecedentes personales.
84. Crear Orden de dieta.
85. Eliminar Orden de dieta.
86. Modificar Orden de dieta.
87. Ver Orden de dieta.
88. Crear Orden Médica.
89. Modificar Orden médica.
90. Ver Orden Médica.
91. Eliminar Orden Médica.
92. Seleccionar medicamentos.
93. Seleccionar enfermedades.
94. Notificar enfermedad de declaración obligatoria.

95. Notificar intervención quirúrgica.
96. Consultar Historia Clínica Electrónica.
97. Registrar fallecimiento.

2.4.2 Requisitos no funcionales del sistema.

Los requisitos no funcionales son condiciones que debe cumplir un sistema para satisfacer un contrato o una especificación. Están regidos por las necesidades del usuario para poder resolver un problema o conseguir un beneficio determinado. Se refieren a las propiedades emergentes del sistema como la fiabilidad, el tiempo de respuesta, la capacidad de almacenamiento, la capacidad de los dispositivos de entrada/salida, y la representación de datos que se utilizan en las interfaces del sistema.

Estos requisitos son de gran significación en la aceptación del software, debido a que representan las ventajas más visibles al usuario y repercuten en el óptimo funcionamiento y mantenimiento del sistema. (38)

2.4.2.1 Usabilidad.

El sistema estará diseñado de manera que los usuarios adquieran las habilidades necesarias para explotarlo en un tiempo reducido:

- Usuarios normales: 20 días.
- Usuarios avanzados: 30 días.

2.4.2.2 Seguridad.

Se mantendrá seguridad y control a nivel de usuario, que garantice el acceso de los mismos sólo a los niveles establecidos de acuerdo a la función que realizan. Las contraseñas podrán cambiarse sólo por el propio usuario o por el administrador del sistema.

Se registrarán todas las acciones que se realizan, llevando el control de las actividades de cada usuario en todo momento. El sistema proporcionará un registro de actividades de cada usuario en el sistema.

Ninguna información que se haya ingresado en el sistema será eliminada físicamente de la base de datos. El sistema permitirá la recuperación de la información de la base de datos a partir de los respaldos o salvas realizadas.

2.4.2.3 Rendimiento.

El sistema minimizará el volumen de datos en las peticiones y además optimizará el uso de recursos críticos como la memoria. Respetará buenas prácticas de programación para incrementar el rendimiento en operaciones costosas para la máquina virtual de Java como la creación de objetos.

2.4.2.4 Soporte.

- Se permitirá la creación de usuarios, otorgamiento de privilegios y roles, asignación de perfiles y activación de permisos.
- Se permitirá administración remota, monitoreo del funcionamiento del sistema en los centros hospitalarios y detección de fallas de comunicación.
- Se permitirá realizar copias de seguridad de la base de datos hacia otro dispositivo de almacenamiento externo, además de recuperar la base de datos a partir de los respaldos realizados.
- Se permitirá el chequeo de las operaciones y acceso de los usuarios al sistema. Se permitirá establecer parámetros de configuración del sistema y actualización de nomencladores.

2.4.2.5 Hardware.

Estaciones de trabajo.

En la solución se incluyen estaciones de trabajo para las consultas del Sistema de Información Hospitalaria alas HIS, las que necesitan capacidad de hardware que soporte un sistema operativo que cuente con un navegador actualizado y siga los estándares web (se recomienda Internet Explorer 7 o superior o Firefox 2.x).

Por lo que se escogieron estaciones de trabajo de 256 MB de memoria RAM y un microprocesador de 2.0 Hz con Sistema operativo GNU/Linux.

2.4.2.6 Servidores.

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

Servidores de Base de datos: 1 DL380 G5, Procesador Intel® Xeon® 5140 Dual-Core 4GB de memoria y 2x72GB de disco y sistema operativo GNU/Linux.

Servidores de Aplicaciones: 2 DL380 G5, Procesador Intel® Xeon® 5140 Dual-Core 4GB de memoria y 2x72GB de disco y sistema operativo GNU/Linux.

Servidores de Intercambio: 1 DL380 G5, Procesador Intel® Xeon® 5140 Dual-Core 2 GB de memoria y 2x72GB de disco y sistema operativo GNU/Linux.

2.4.2.7 Software.

El servidor debe correr en sistemas operativos Windows, Unix y Linux, utilizando la plataforma JAVA (Java virtual machine, JBoss AS y PostgreSql). Los clientes deberán disponer de un navegador web que debe ser Firefox 2 o superior.

2.4.2.8 Restricciones de diseño.

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

2.4.2.9 Requisitos para la documentación de usuarios en línea y ayuda del sistema.

Se posibilitará el uso de ayudas dinámicas y tutoriales en línea sobre el funcionamiento del sistema. El objetivo principal de estos materiales es servirle de ayuda al usuario para que pueda aclarar sus dudas respecto al sistema.

2.4.2.10 Interfaz de usuario.

Las ventanas del sistema contendrán bien estructurados los datos, además de permitir la interpretación correcta de la información. La interfaz contará con accesos directos y menús desplegables que faciliten y aceleren su utilización. La entrada de datos incorrecta será detectada e informada al usuario. Todos los textos y mensajes en pantalla aparecerán en idioma español.

2.4.2.11 Portabilidad.

El producto podrá ser utilizado bajo los sistemas operativos Linux o Windows ya que son los sistemas operativos más comunes.

2.4.3 Modelo de casos de uso del sistema.

Los requisitos o capacidades internas del sistema a desarrollar han sido expresados como casos de uso teniendo en cuenta que estos son actividades atómicas que ofrecen un resultado definido para cada acción que realiza el usuario.

El modelo de casos de uso del sistema documenta el comportamiento del sistema desde el punto de vista del usuario, permitiendo representar las funciones que se desean en el sistema (*casos de uso*), el entorno del sistema (*actores*), y las relaciones entre ellos. Aunque la parte más visible de dicho modelo son los diagramas de casos de uso, suele ir acompañado de una especificación textual de cada uno de los casos de uso.

Los actores del sistema no forman parte del sistema, sino que representan elementos que interactúan con él. Estos elementos son nombrados roles que puede estar formado por una o varias personas, un equipo o un sistema automatizado. Un actor puede introducir o recibir información del sistema. (39)

Los actores del sistema identificados son:

Actores del sistema	Funciones
Paciente.	Se le entregan las indicaciones médicas, el récipe y la referencia una vez egresado de la Emergencia.
Familiar.	Se le entrega el Certificado de defunción del familiar muerto.
Enfermera.	Es la persona a quien se le solicita el Certificado de defunción.

Tabla 2.2 Actores del negocio.

2.4.3.1 Definición de los actores del sistema.

Personal de emergencia: Puede ser un técnico de registros y estadísticas de salud, un enfermero(a) o un médico. Es el personal de salud autorizado para registrar la atención del paciente en el sistema.

Médico de emergencia: Este rol tiene permisos para registrar la atención del paciente y modificar la Hoja de emergencia.

Técnico de registro y estadísticas de salud de emergencia: Este rol es el encargado de registrar el movimiento del paciente por las áreas de Emergencias y atender la solicitud de cupos. También crea la Hoja de emergencia y registra algunos datos, pero tiene acceso restringido sobre la Hoja de emergencia.

Diagrama de actores

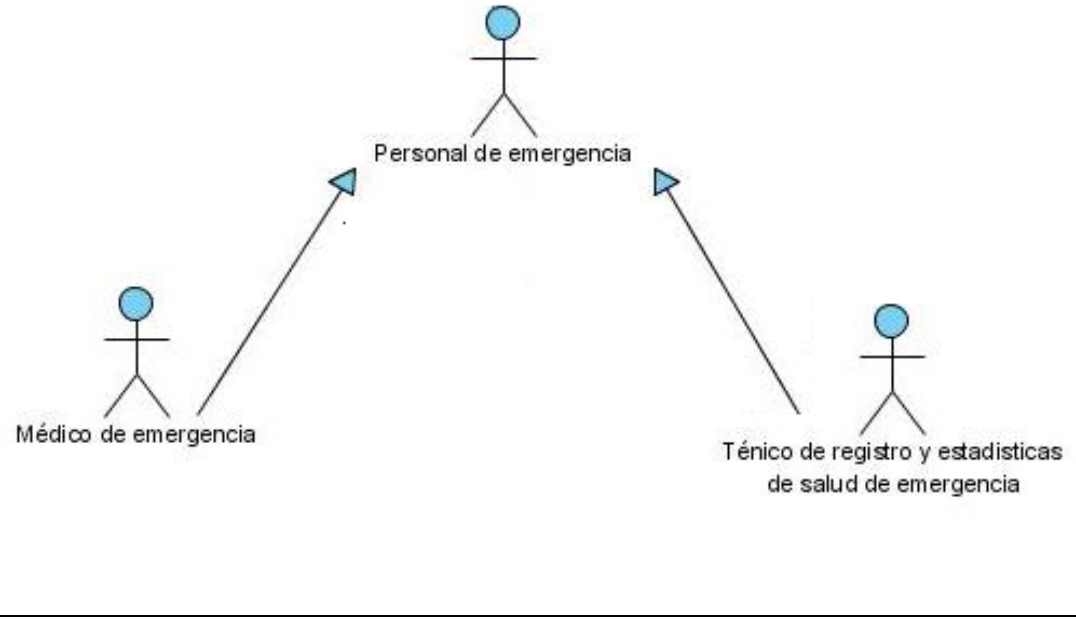


Figura 2.6 Diagrama de actores.

2.4.3.2 Diagrama de Casos de Uso del Sistema.

Proceso: Recibir paciente.

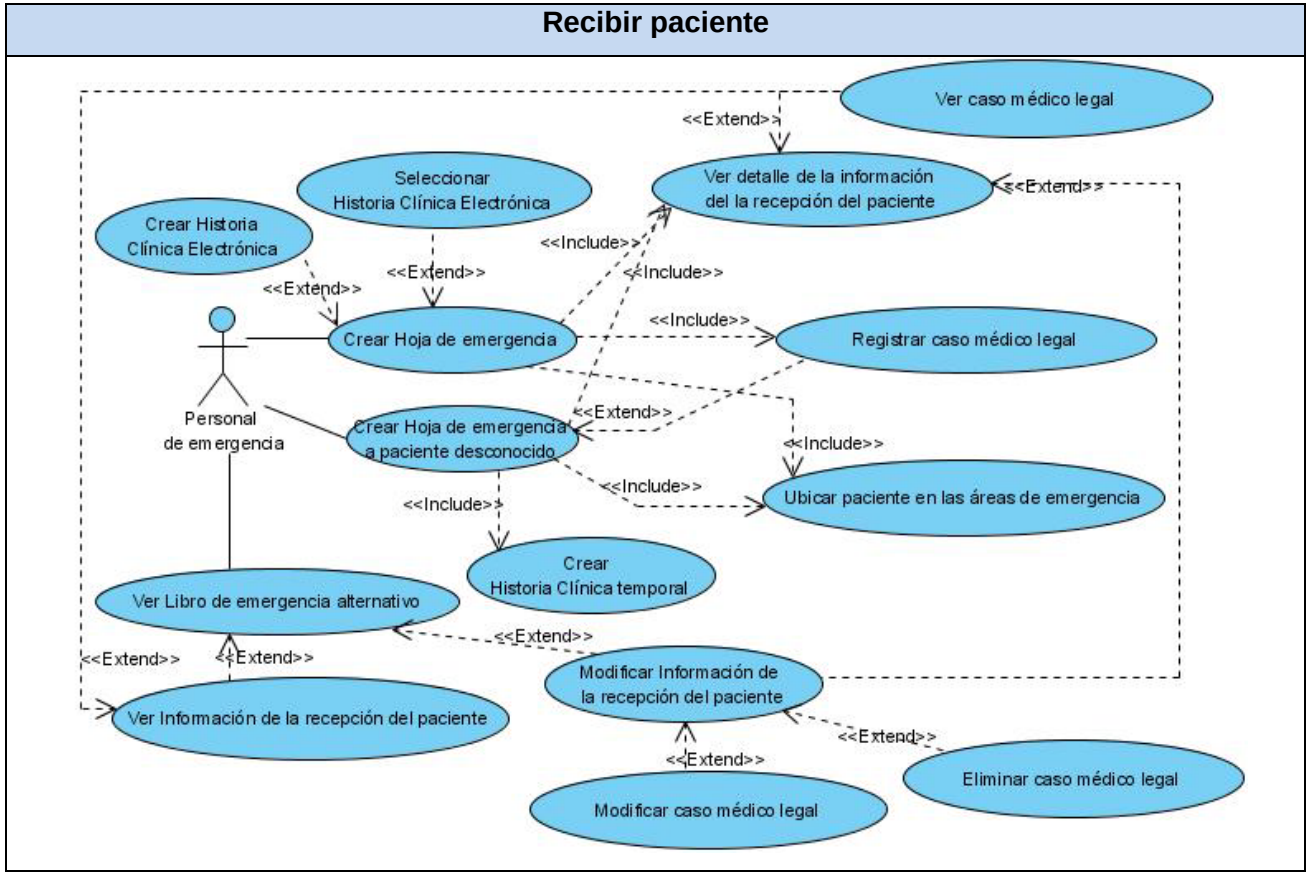


Figura 2.7 Diagrama de casos de uso del sistema. Recibir paciente.

Proceso: Atender paciente.

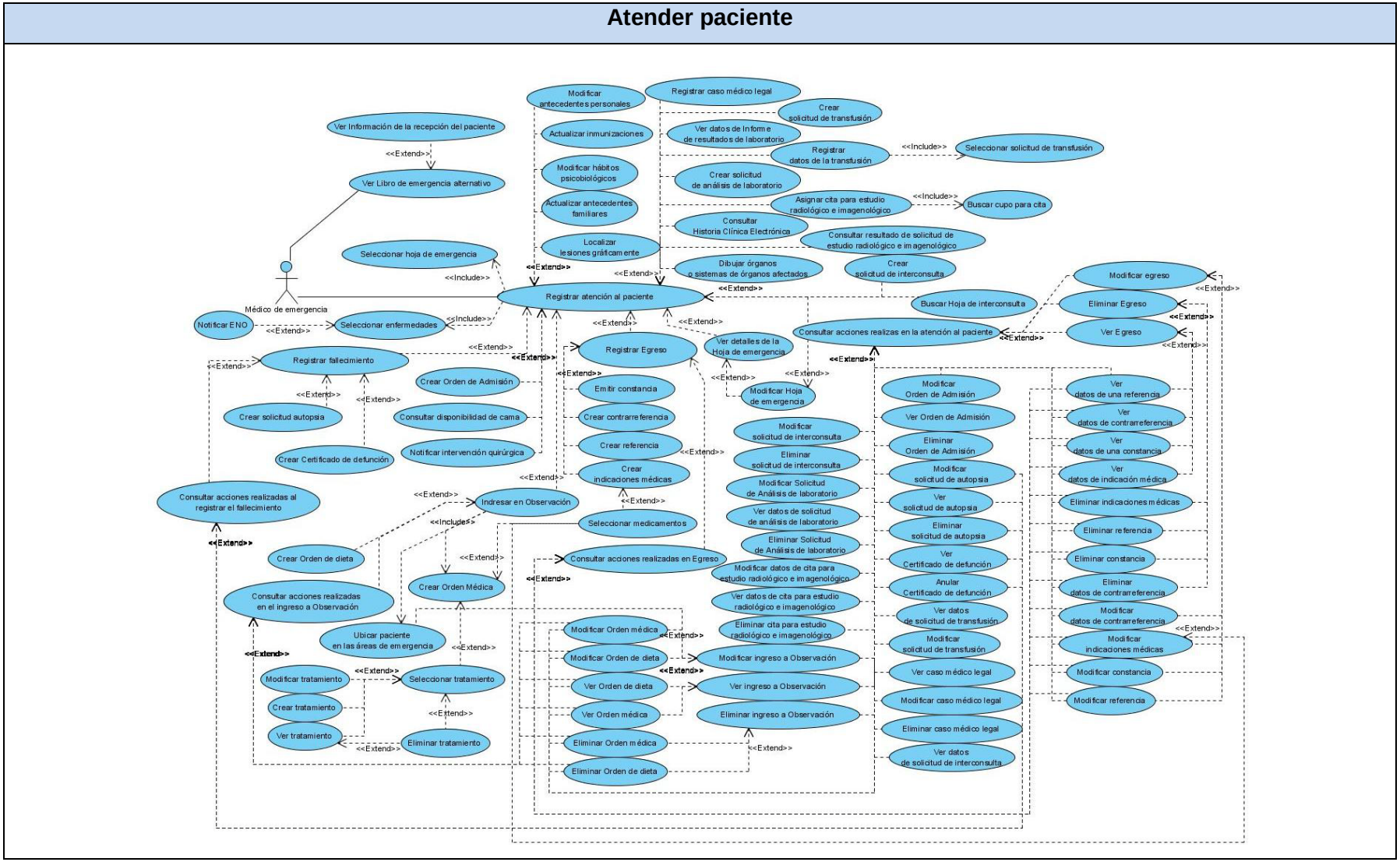


Figura 2.8 Diagrama de casos de uso del sistema. Atender paciente.

2.4.3.3 Descripción Textual de los Casos de Uso.

Crear Hoja de emergencia	
Actores:	Personal de emergencia
Resumen:	El caso de uso inicia cuando el actor accede a la opción Recibir paciente. El sistema brinda la posibilidad de buscar la Historia Clínica Electrónica del paciente dado su número de cédula. Una vez encontrada la Historia Clínica del paciente el sistema permite introducir los datos para crear la Hoja de emergencia. El sistema crea la Hoja de emergencia y la asocia a la Historia Clínica. Se muestra la Hoja de emergencia creada permitiendo modificar sus datos. El caso de uso termina.
Precondición:	El usuario debe tener creado una Historia clínica previamente.
Referencias:	Ubicar paciente en las áreas de emergencia. Registrar caso médico legal. Seleccionar Historia Clínica Electrónica. Crear Historia Clínica Electrónica. Ver detalle de la información del la recepción del paciente.

Tabla 2.3 Descripción Textual del Caso de Uso. Crear Hoja de emergencia.

Registrar caso médico legal	
Actores:	Personal de emergencia.
Resumen:	El caso de uso inicia cuando el actor accede a la opción Registrar caso médico legal. El sistema brinda la posibilidad de introducir los datos para registrar el caso. El actor introduce los datos. El sistema registra la información. El caso de uso termina.
Precondición:	No existe.
Referencias:	No existe.

Tabla 2.4 Descripción Textual del Caso de Uso. Registrar caso médico legal.

Ver Libro de emergencia	
Actores:	Personal de emergencia.
Resumen:	El caso de uso inicia cuando el actor accede a la opción Ver Libro de emergencia. El sistema busca y muestra todas las Hojas de emergencia del día presente. El actor selecciona la Hoja de emergencia deseada. El sistema carga en la vista anterior la Hoja de emergencia seleccionada. El caso de uso termina.
Precondición:	No existe.
Referencias:	Ver Información de la recepción del paciente.

Tabla 2.5 Descripción Textual del Caso de Uso. Ver Libro de emergencia.

Ver información de la recepción del paciente	
Actores:	Personal de emergencia.
Resumen:	El caso de uso inicia cuando el actor selecciona una Hoja de emergencia de las listadas y accede a la opción Ver información de la recepción del paciente. El sistema muestra los datos de la recepción del paciente. El caso de uso termina.
Precondición:	Para ver la información de la recepción del paciente debe haber sido seleccionada una Hoja de emergencia en el Libro de emergencia.
Referencias:	No existe.

Tabla 2.6 Descripción Textual del Caso de Uso. Ver información de la recepción del paciente.

Modificar información de la recepción del paciente	
Actores:	Personal de emergencia.
Resumen:	El caso de uso inicia cuando el actor accede a la opción Modificar información de la recepción del paciente. El sistema muestra los datos anteriormente registrados y permite modificarlos. El caso de uso termina.
Precondición:	Para modificar la información de la recepción del paciente debe haber sido seleccionada una Hoja de emergencia en el Libro de emergencia.
Referencias:	Modificar caso médico legal. Eliminar caso médico legal.

Tabla 2.7 Descripción Textual del Caso de Uso. Modificar información de la recepción del paciente.

Registrar atención al paciente	
Actores:	Médico de emergencia.
Resumen:	El caso de uso se inicia cuando el actor selecciona la Hoja de emergencia del paciente. El sistema da la posibilidad de buscar la Hoja de emergencia que se desea modificar y brinda la posibilidad de introducir los valores necesarios para registrar la atención al paciente. El actor introduce los datos. El sistema actualiza los datos de la Hoja de emergencia, el caso de uso termina.
Precondición:	La Hoja de emergencia debe haber sido creada anteriormente por el técnico de registros y estadísticas y seleccionada por el médico que brinda la atención al paciente.
Referencias:	Seleccionar Hoja de emergencia. Localizar lesiones gráficamente. Actualizar antecedentes familiares. Modificar hábitos psicobiológicos. Actualizar inmunizaciones.

	Modificar antecedentes personales. Dibujar órganos o sistemas de órganos afectados. Consultar resultado de solicitud de estudio radiológico e imagenológico. Consultar Historia Clínica Electrónica. Asignar cita para estudio radiológico e imagenológico. Crear solicitud de análisis de laboratorio. Registrar datos de la transfusión. Ver datos de Informe de resultados de laboratorio. Crear solicitud de transfusión. Registrar caso médico legal. Crear solicitud de interconsulta. Hoja de interconsulta. Consultar acciones realizas en la atención al paciente. Registrar Egreso. Dejar en Observación. Registrar cirugía de emergencia solicitada. Consultar disponibilidad de cama. Crear Orden de Admisión. Registrar fallecimiento. Seleccionar enfermedades.

Tabla 2.8 Descripción Textual del Caso de Uso. Registrar atención al paciente.

Registrar egreso	
Actores:	Médico de emergencia.
Resumen:	El caso de uso inicia cuando el actor accede a la opción Egresar. El sistema brinda la posibilidad de introducir los datos para registrar el egreso. El actor introduce los datos. El sistema registra la información. El caso de uso termina.
Precondición:	No existe.
Referencias:	Crear referencia. Emitir constancia. Crear indicaciones médicas. Consultar acciones realizadas en Egreso. Crear contrarreferencia.

Tabla 2.9 Descripción Textual del Caso de Uso. Registrar egreso.

Los sistemas que gestionan la información del área de Emergencias de las instituciones hospitalarias existentes presentan licencias de software propietario y corren sobre el sistema operativo Windows, por lo que no cumplen los requisitos que el usuario necesita, lo que hace necesario un software que gestione los procesos del área de Emergencias de las instituciones hospitalarias. Este deberá cumplir con los requerimientos expuestos anteriormente con el objetivo de facilitar y hacer más eficiente y eficaz el trabajo de los usuarios del sistema.

En el presente capítulo se analizaron los procesos del negocio que se desarrollan en el área de Emergencias lo que permitió hallar las actividades innecesarias y depurarlas hasta obtener los requisitos funcionales del sistema. Se definieron los actores y el diagrama de casos de uso del sistema así como los requerimientos no funcionales del software.

CAPÍTULO 3 ANÁLISIS Y DISEÑO DEL SISTEMA.

En el presente capítulo se realiza una descripción detallada de sistema propuesto. Se muestra la arquitectura definida por los arquitectos de software del Área Temática Gestión Hospitalaria y se expone el diseño de la solución de la aplicación.

3.1 Descripción de la arquitectura, fundamentación.

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema. Una Arquitectura de Software, también denominada Arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información.

El sistema que se propone presenta una arquitectura basada en uno de los estilos más utilizados del patrón arquitectónico Modelo Vista Controlador (MVC), expuesto en el *Capítulo 1*: el patrón en capas, el cual separa los elementos de la presentación, el negocio y el acceso a datos del sistema, para lograr que cada capa se comunique con sus adyacentes, permitiendo que los cambios de una capa puedan realizarse sin afectar a todo el sistema.

El uso del framework JSF permite contar con la implementación de dicho patrón. Brindándole a la aplicación la posibilidad de tener una separación clara entre cómo se muestra la información al usuario, cómo se manejan las acciones que el usuario desea hacer sobre el sistema y cómo se realizan estas acciones modificando y validando la información.

Este patrón se evidencia de la siguiente forma: la Vista se corresponde con las páginas xhtml las cuales son interfaces de usuario que le presentan el sistema a este, manejan las acciones realizadas sobre la interfaz por el usuario y recogen la información entrada por este. El controlador se corresponde con las clases controladoras para cada caso de uso, que se encargan del procesamiento de la información en correspondencia con la lógica del negocio en cuestión. La información manejada en todo el sistema

coincide con el modelo que es una representación orientada a objetos, en forma de clases de entidad, de las tablas de la base de datos del sistema.

Otros patrones utilizados fueron los llamados patrones GRASP (Patrones para asignar responsabilidades), que tuvieron una importante utilidad en el diseño realizado. A cada clase le fueron asignadas 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. Con esto se logró conservar el encapsulamiento ya que los objetos realizan lo que se les pide utilizando información que ellos poseen. (40)

3.2 Modelo de diseño.

Mediante el modelo de diseño se hace un refinamiento del proceso de análisis anteriormente realizado. Para ello se tienen en cuenta los requisitos no funcionales del sistema ya que el principal propósito del modelado del diseño es crear un plano del modelo de implementación. También se define la arquitectura del sistema. Los casos de uso son realizados por las clases del diseño y sus objetos, a partir de los cuales se forma el diagrama de clases del diseño.

Es preciso describir los términos empleados en el modelo de diseño para un mejor entendimiento del mismo.

Clase de diseño: es una abstracción de una clase o construcción en la implementación del sistema.

Diagramas de clases de diseño: exponen un conjunto de interfaces, colaboraciones y sus relaciones. Se utilizan para modelar la vista de diseño estática de un sistema. Permiten visualizar, especificar y documentar modelos estructurales. Estos forman parte de las realizaciones de casos de uso.

Diagramas de interacción: muestran gráficamente cómo los objetos se comunican entre sí a fin de cumplir con los requerimientos. Estos se emplean para modelar los aspectos dinámicos de un sistema.

Dentro de los diagramas de interacción se definen los diagramas de secuencia y colaboración. Generalmente se realizan los diagramas de colaboración en el análisis y los de secuencia en el diseño.

Diagrama de secuencia: es un diagrama de interacción que destaca la ordenación temporal de los mensajes. (41)

Para la elaboración del modelo de diseño, se define una estructura de paquetes que permite dividir el sistema en fragmentos manejables para su futura implementación. Se emplea el criterio de empaquetamiento por proceso, siguiendo la estructura de procesos definidos en el sistema.

Los paquetes son graficados mostrando la relación que guardan entre sí. Estos utilizan el paquete repositorio de clases para su funcionamiento.

Un paquete referente a procesos, está conformado por subpaquetes que responden a las realizaciones de casos de uso, donde cada una de ellas contiene un diagrama de clases del diseño y los respectivos diagramas de secuencia.

El paquete repositorio contiene dos subpaquetes, uno para las entidades y otro para las sesiones. En el subpaquete de entidades se encuentran las clases autogeneradas definidas en el diseño de acuerdo a las tecnologías que serán usadas en la implementación. Las clases autogeneradas, como su nombre lo indica, se autogeneran desde la base de datos utilizando el ORM Hibernate (permite la generación de objetos java desde la base de datos). Las clases personalizadas son aquellas que se modifican, por lo que pueden heredar de las entidades autogeneradas.

El subpaquete de sesiones está conformado por las clases controladoras autogeneradas por el entorno de desarrollo, además de las clases controladoras personalizadas y las controladoras del proceso.

A continuación se muestra el modelo de diseño del sistema:

Diagrama de paquetes

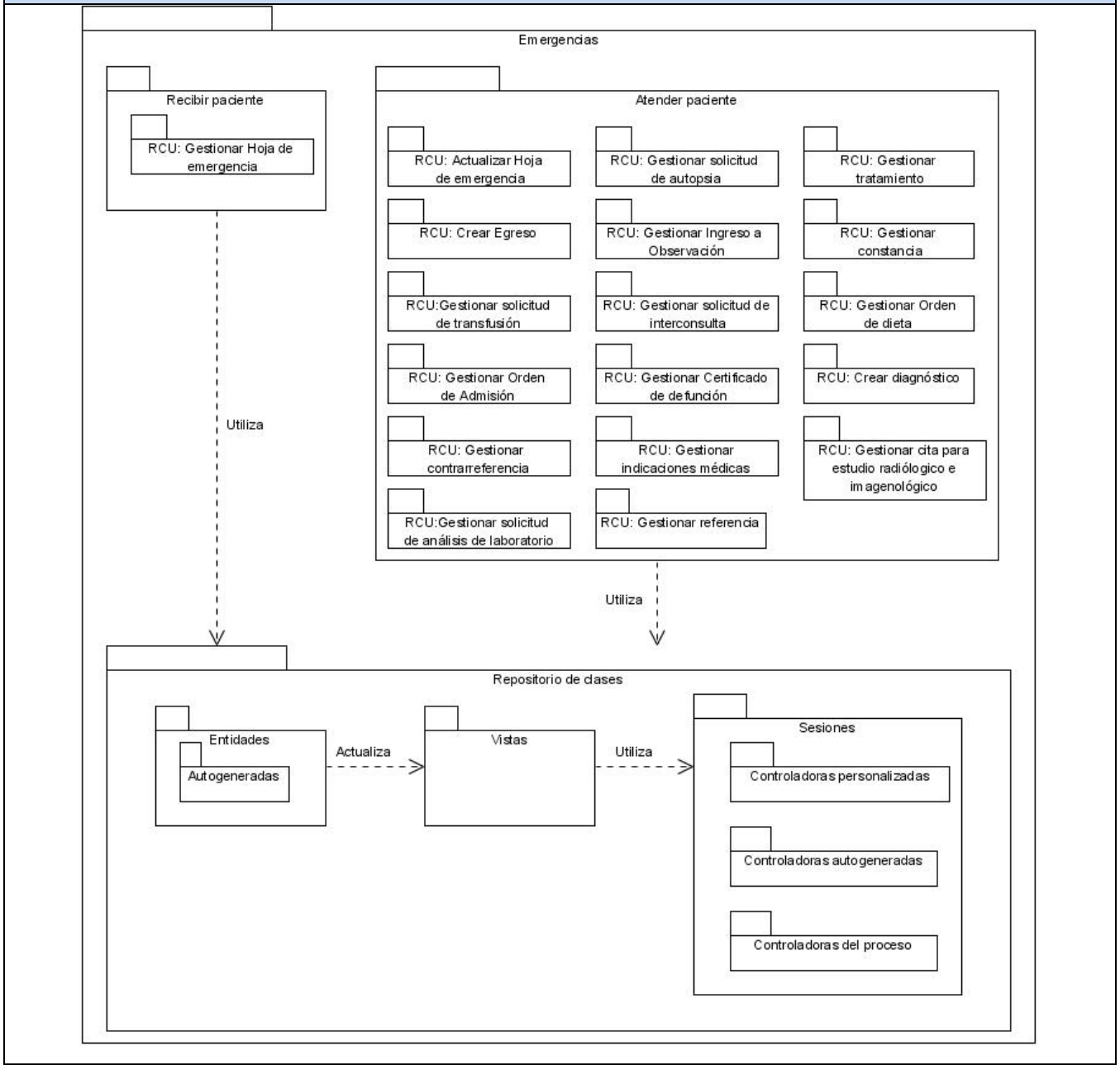


Figura 3.1 Diagrama de paquetes.

Como se evidencia en el diagrama de paquetes, el sistema está dividido en dos procesos: Recibir paciente y Atender paciente. Para cada proceso se modela un diagrama de clases del diseño y por cada escenario de dicho diagrama es modelado un diagrama de interacción. Como diagrama de interacción fue seleccionado el de secuencia.

Recibir paciente.

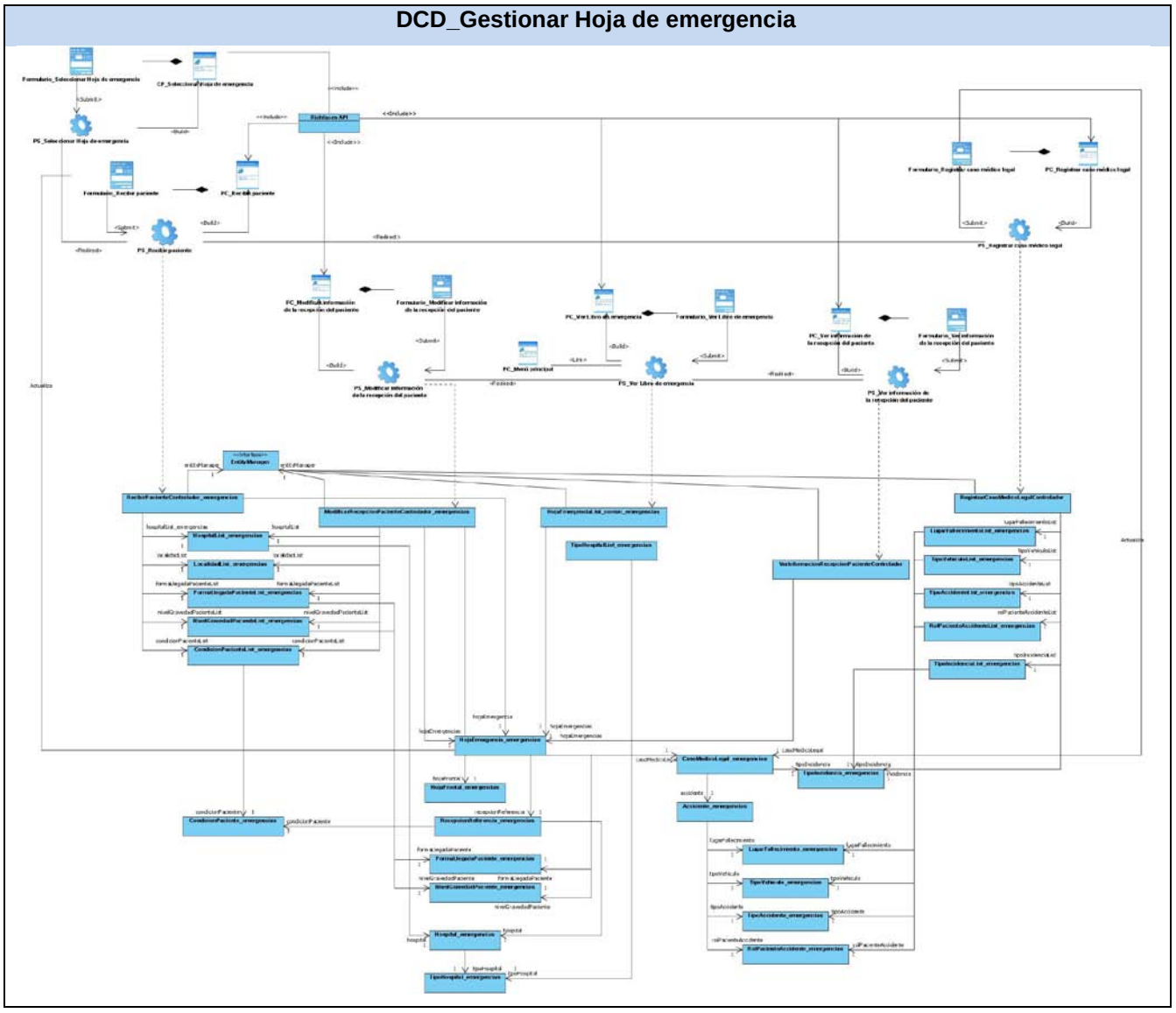


Figura 3.2 Diagrama de clases del diseño. Recibir paciente.

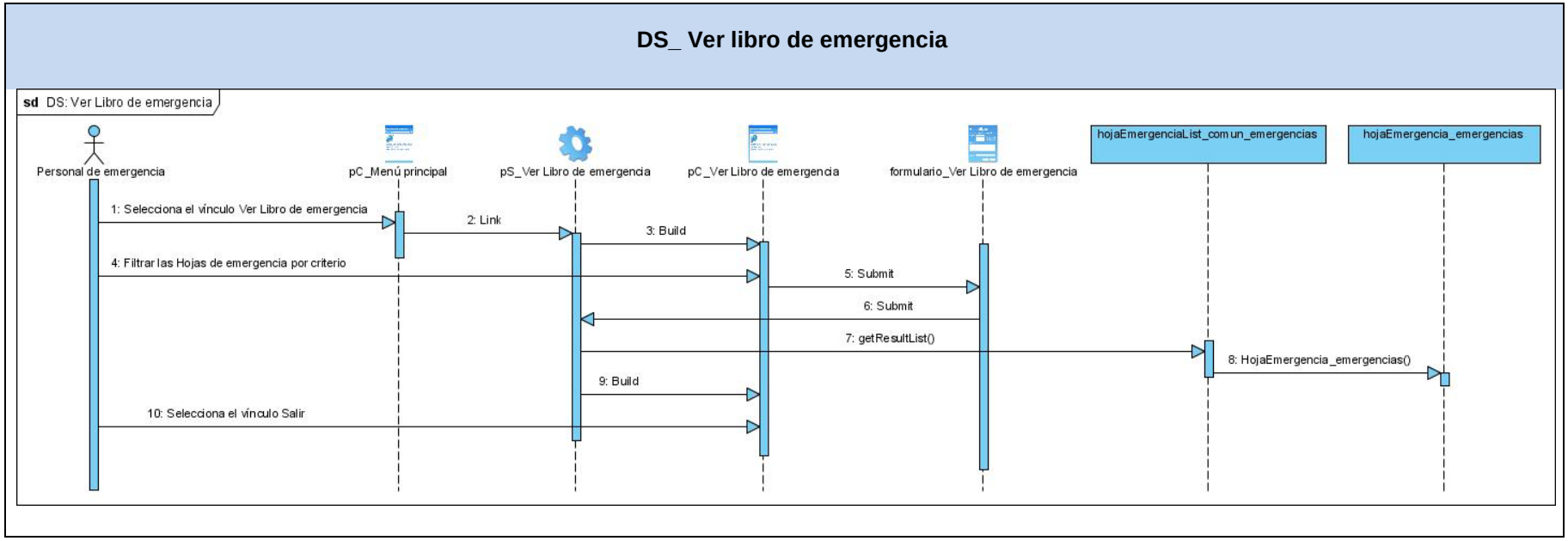


Figura 3.3 Diagrama de secuencias. Ver libro de emergencias.

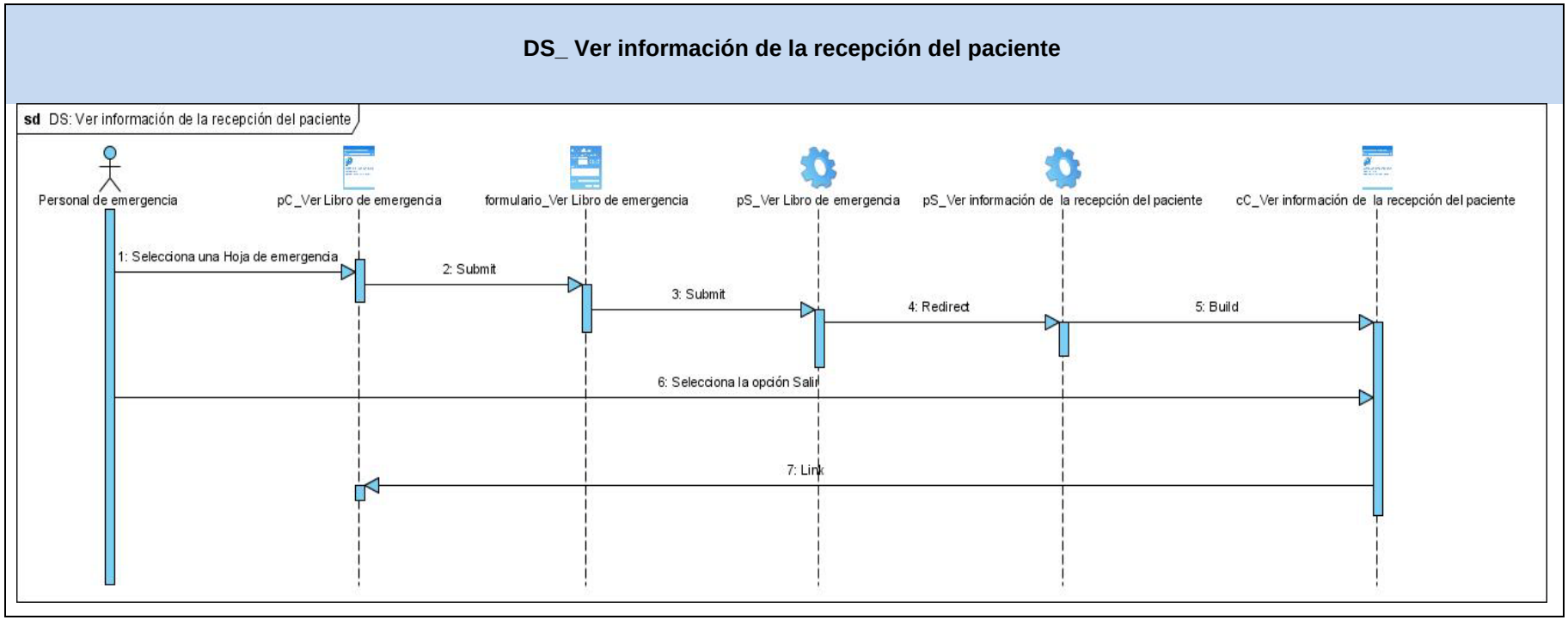


Figura 3.4 Diagrama de secuencias. Ver información de la recepción del paciente.

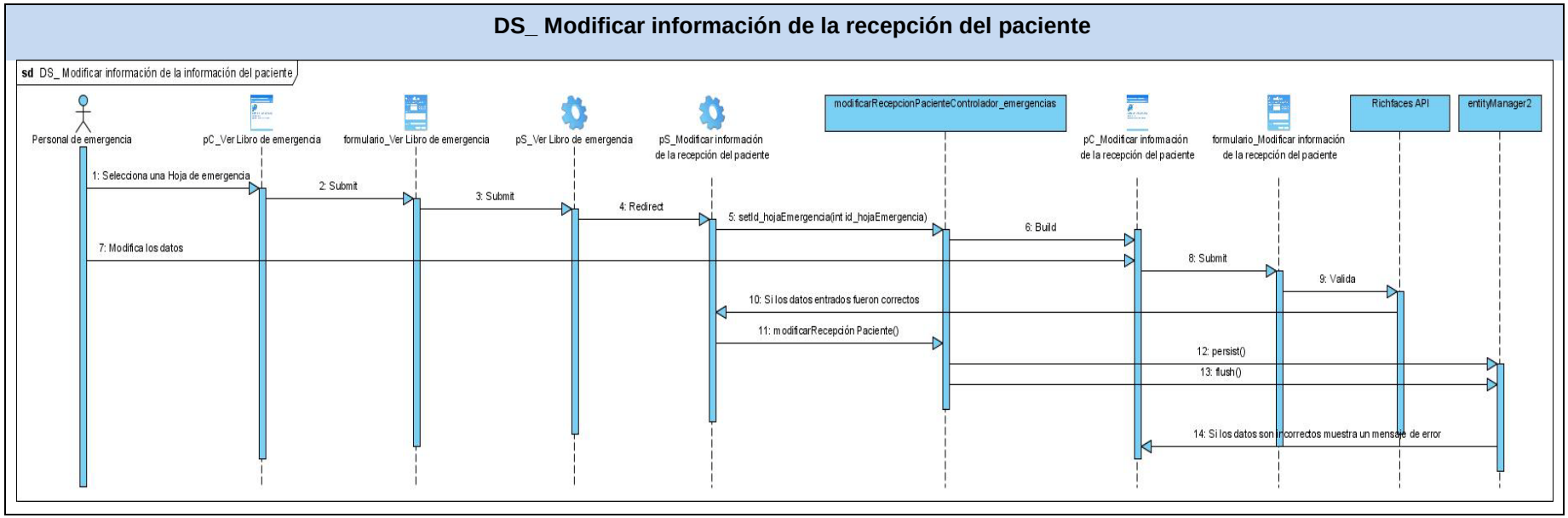


Figura 3.5 Diagrama de secuencias. Modificar información de la recepción del paciente.

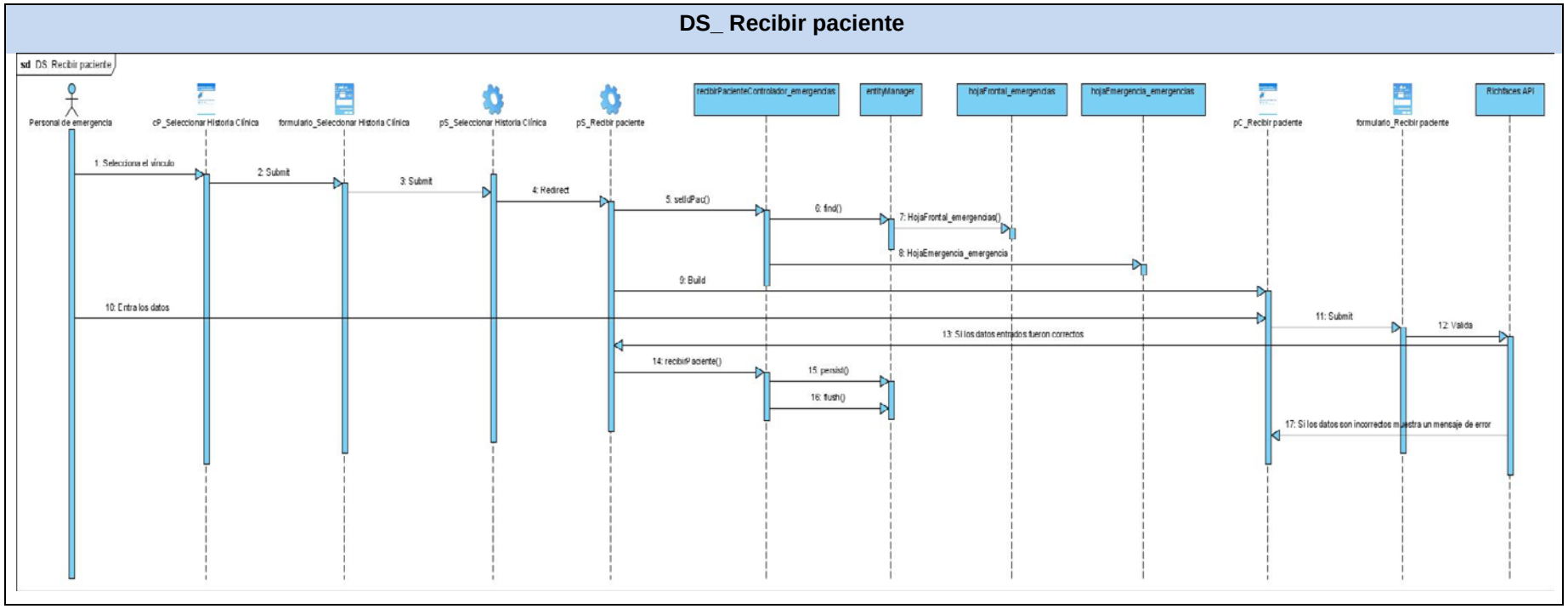


Figura 3.6 Diagrama de secuencias. Recibir paciente.

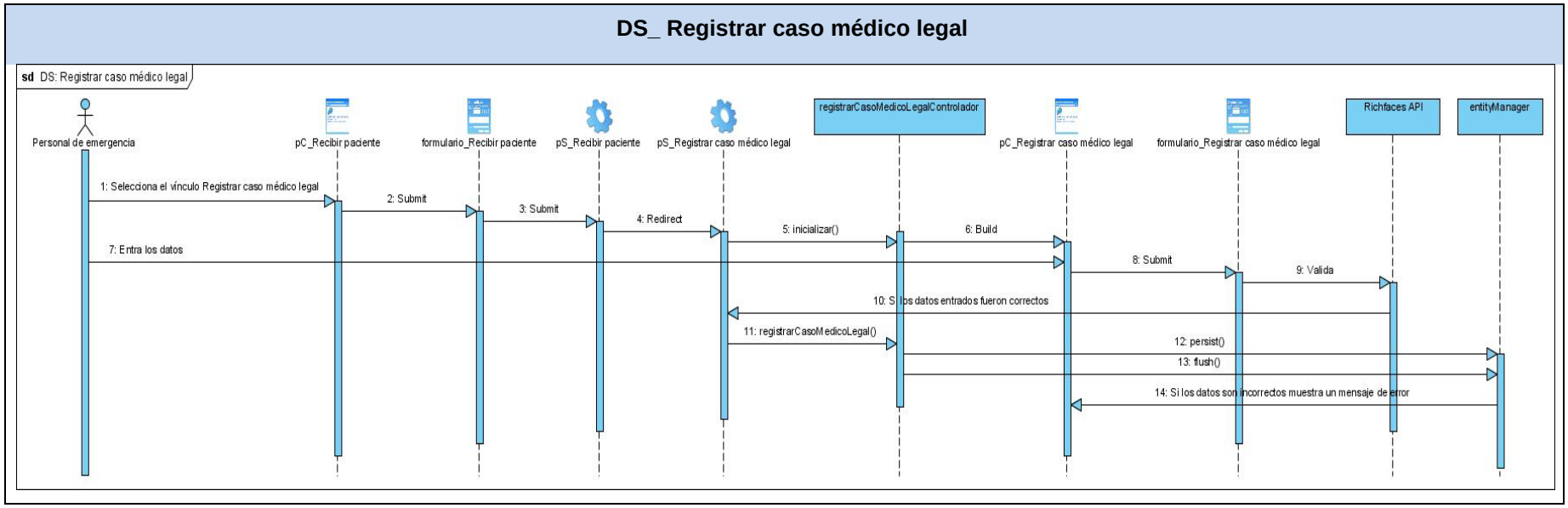


Figura 3.7 Diagrama de secuencias. Registrar caso médico legal.

Atender paciente.

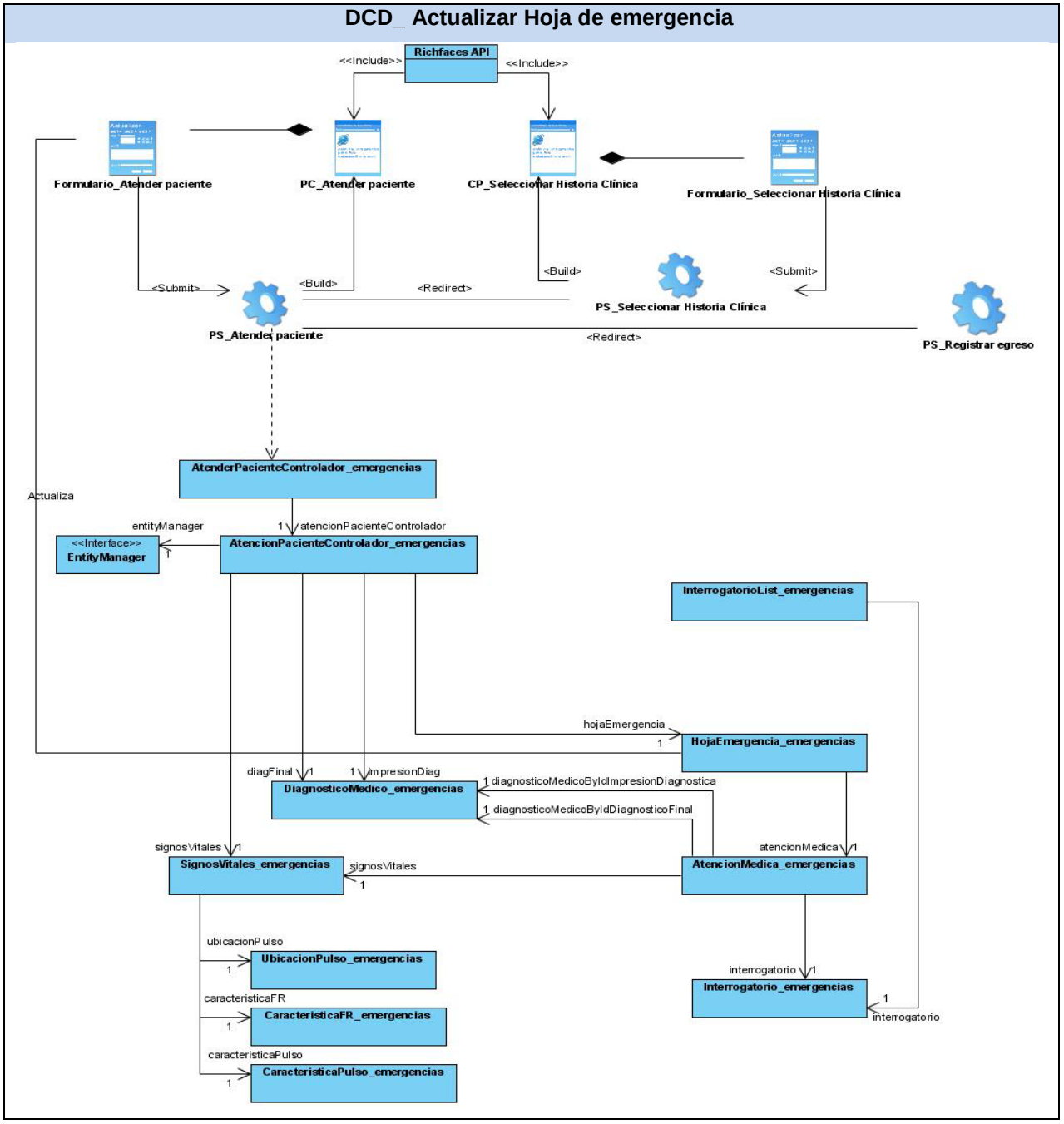


Figura 3.8 Diagrama de clases del diseño. Actualizar Hoja de emergencia.

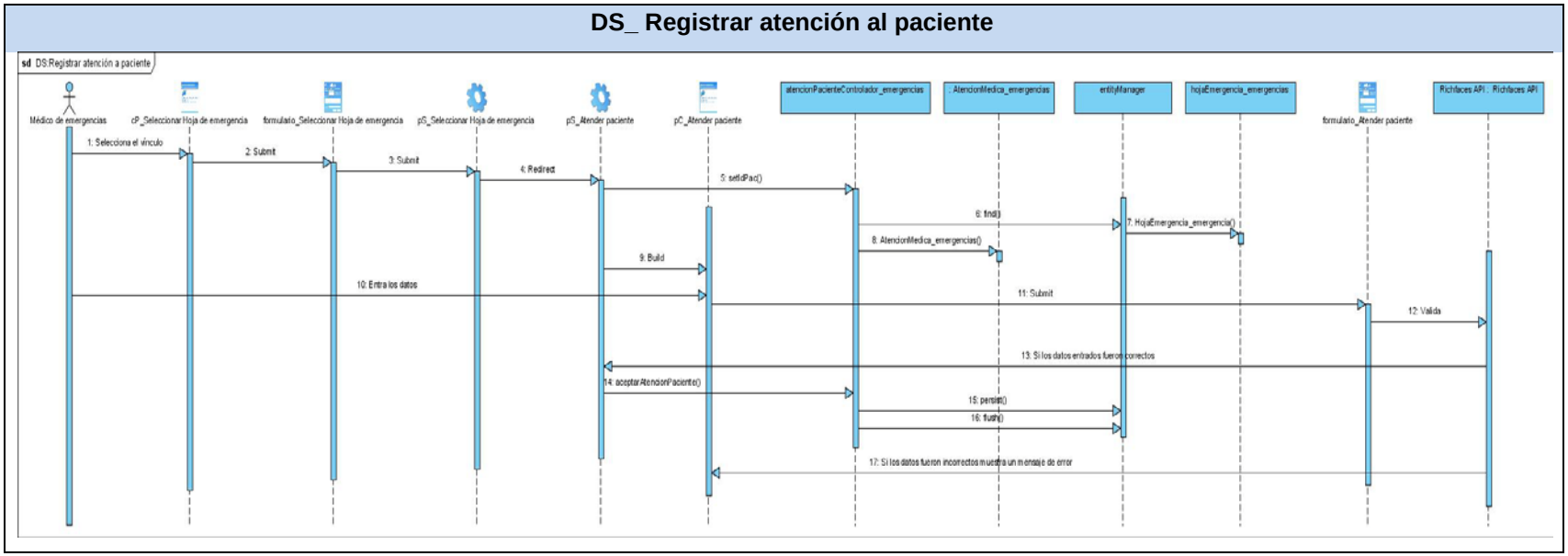


Figura 3.9 Diagrama de secuencias. Registrar atención al paciente.

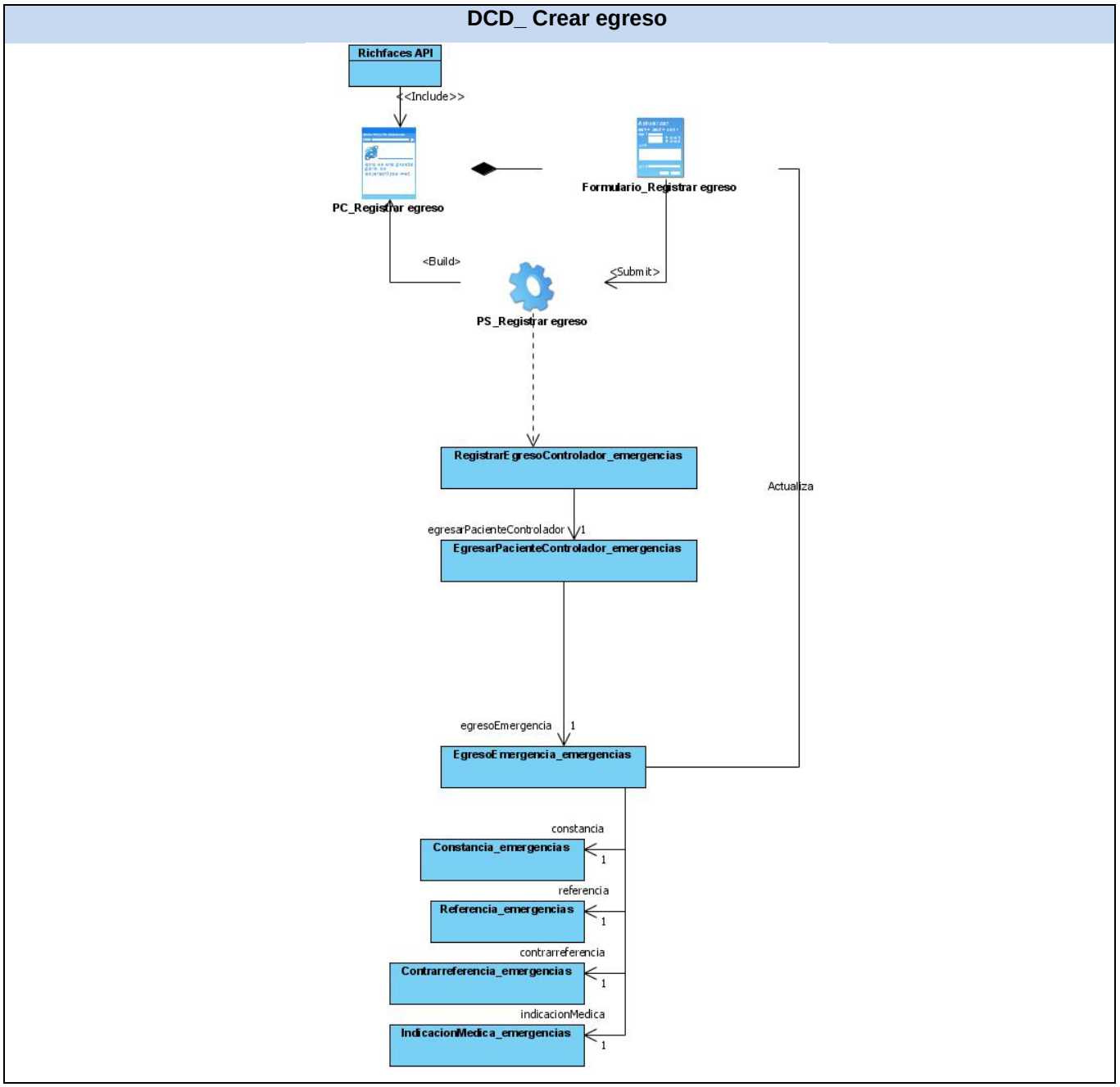


Figura 3.10 Diagrama de clases del diseño. Crear egreso.

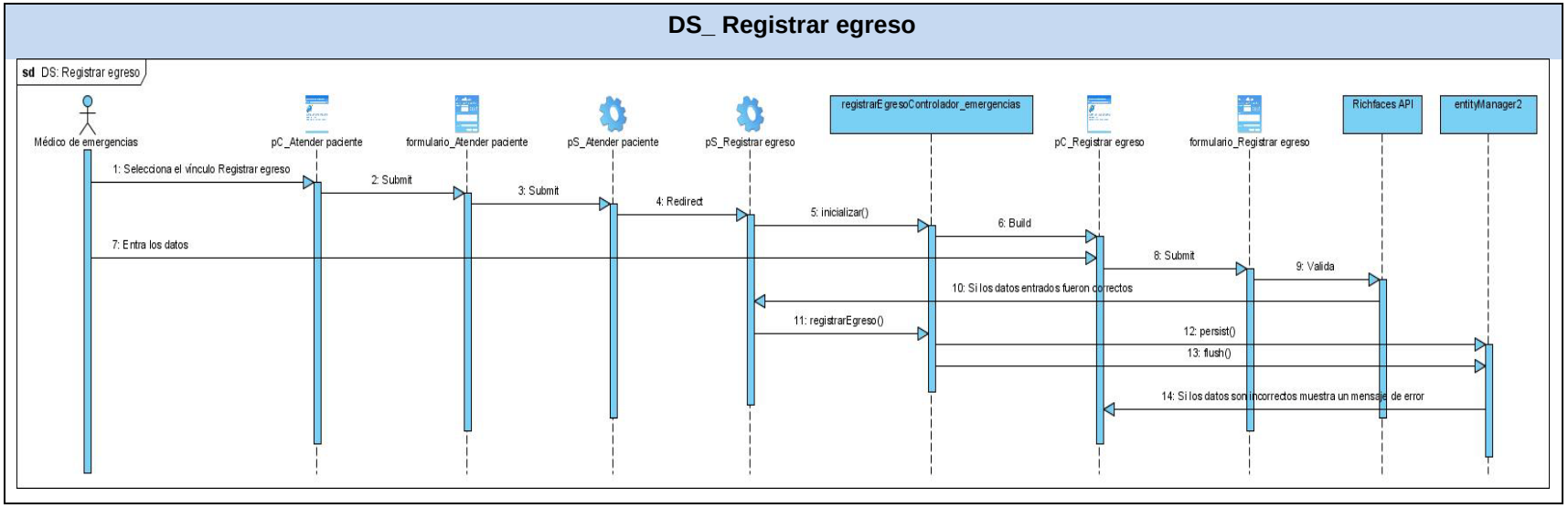


Figura 3.11 Diagrama de secuencias. Registrar egreso.

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 intepretadas 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 tags <form> y que puede contener elementos especiales llamados controles (casillas de verificación (checkboxes), radiobotones (radio buttons), menús, entre otros.), y rótulos (labels) en esos controles. Los usuarios normalmente "completan" un formulario modificando sus controles (introduciendo texto, seleccionando objetos de un menú, etc.), y lo envían al servidor donde estos son procesados. Es una manera de obtener en el servidor información entrada por el usuario en el cliente.

Controladoras: Las clases controladoras o controladoras como se les también se les dice son clases que implementan la lógica del negocio que se está informatizando, generalmente cada una de estas se encargan de la implementación de un caso de uso o un proceso en dependencia de la complejidad de los mismos.

Seguidamente serán explicadas algunas de las clases que han sido identificadas para su futura implementación describiéndose las responsabilidades que realizarán las páginas servidoras que responden a la Lógica de Negocio. De esta manera se tendrá una comprensión mayor del funcionamiento que tendrá el sistema en desarrollo.

Nombre: RecibirPacienteControlador_emergencias.	
Tipo de clase: Controladora.	
Para cada responsabilidad:	
Nombre:	void setId_hojaFrontal(int id_hojaFrontal).
Descripción:	Recibe y cambia el valor del id del paciente seleccionado. Busca este paciente y le crea una hojaEmergencia al mismo, donde se recogerá la información de su llegada.
Nombre:	String Aceptar().
Descripción:	Valida y guarda los campos llenados de la hojaEmergencia del paciente.
Nombre:	HojaEmergencia_emergencias getHojaEmergencia().
Descripción:	Retorna el atributo hojaEmergencia.
Nombre:	void setHojaEmergencia(HojaEmergencia_emergencias hojaEmergencia).
Descripción:	Modifica el atributo hojaEmergencia.

Tabla 3.1 Descripción de la clase controladora RecibirPacienteControlador_emergencias.

Nombre: VerInfomacionRecepcionControlador.	
Tipo de clase: Controladora.	
Para cada responsabilidad:	
Nombre:	void setIdHojaEmergencia(int idHojaEmergencia).
Descripción:	Recibe y cambia el valor del id de la hojaEmergencia seleccionada. Busca esta hojaEmergencia para que esta sea mostrada.

Tabla 3.2 Descripción de la clase controladora VerInfomacionRecepcionControlador.

Nombre: HojaEmergenciaList_comun_emergencias.	
Tipo de clase: Controladora.	
Para cada responsabilidad:	
Nombre:	boolean isBusquedaSimple().
Descripción:	Devuelve un booleano que indica si es una búsqueda simple o no la que se está efectuando.

Tabla 3.3 Descripción de la clase controladora HojaEmergenciaList_comun_emergencias.

Nombre: ModificarRecepcionPacienteControlador.	
Tipo de clase: Controladora.	
Para cada responsabilidad:	
Nombre:	void setId_hojaEmergencia(int id_hojaEmergencia).
Descripción:	Recibe y cambia el valor del id de la hojaEmergencia seleccionada. Busca esta hojaEmergencia para que esta sea modificada.
Nombre:	String Modificar().
Descripción:	Persiste los cambios sobre la entidad.
Nombre:	String Cancelar().
Descripción:	Cancela la operación de modificación de la entidad.

Tabla 3.4 Descripción de la clase controladora ModificarRecepcionPacienteControlador.

Nombre: RegistrarCasoMedicoLegal.	
Tipo de clase: Controladora.	
Para cada responsabilidad:	
Nombre:	void Inicializar().
Descripción:	Inicializa los campos del caso médico legal.
Nombre:	void Aceptar().
Descripción:	Permite aceptar la creación del caso médico-legal.
Nombre:	String Cancelar().
Descripción:	Cancela la operación de registrar el caso médico-legal.

Tabla 3.5 Descripción de la clase controladora RegistrarCasoMedicoLegal.

Nombre: RegistrarEgresoControlador_emergencias.	
Tipo de clase: Controladora.	
Para cada responsabilidad:	
Nombre:	void Inicializar().
Descripción:	Inicializa los campos del caso médico legal.
Nombre:	String eliminarAccionesRealizadas(int accion).
Descripción:	Elimina según el parámetro, una de las entidades creadas en el egreso.
Nombre:	String aceptar() .
Descripción:	Permite aceptar la creación del egreso.
Nombre:	String Cancelar().
Descripción:	Cancela la operación de registrar el egreso.

Tabla 3.6 Descripción de la clase controladora RegistrarEgresoControlador_emergencias.

Como resultado del estudio realizado en este capítulo, correspondiente al flujo de diseño, se identificaron las fundamentales clases que deben ser definidas para que el sistema funcione satisfactoriamente. Así como, los atributos y métodos que deben tener las mismas para brindarle al desarrollador una idea clara de lo que se debe implementar. Además se obtuvo la realización de casos de uso por procesos, donde se elaboraron los diagramas de clases y los diagramas de secuencia correspondientes.

CAPÍTULO 4: IMPLEMENTACIÓN.

La fase de Implementación es la consecuencia de la fase de Diseño. En este capítulo se describen las clases y subsistemas implementados en términos de componentes. Se muestra el Diagrama de Despliegue como parte del Modelo de Implementación, que indica la distribución física de la solución implementada. Además se proporciona una detallada explicación de dicha solución.

4.1. Modelo de datos

El modelo de datos es la traducción del análisis de requisitos al esquema conceptual mediante una representación grafica de las entidades y sus relaciones. Este, ayuda a entender y nombrar la información, evita la redundancia, asegura la corrección, validación y completitud de los datos y su organización refleja la política del negocio. Está compuesto por entidades, atributos y sus relaciones. Las entidades son objetos de los que el sistema necesita guardar información. Por ser un concepto o abstracción se suele emplear el sustantivo en singular para nombrar la entidad. Se simbolizan representando un rectángulo.

Los atributos son cada una de las características asociadas a una entidad. Se representan colocando su nombre dentro del rectángulo de la entidad. Los atributos pueden ser clasificados en: obligatorios, opcionales, claves foráneas y claves primarias (estas se dividen en simples y compuestas). La representación gráfica de la clave primaria es un signo de suma y la de la foránea es un símbolo de número.

Las relaciones muestran la forma en que dos entidades se asocian. Se representan mediante una línea que une las dos entidades implicadas y tienen principalmente dos características: cardinalidad y obligatoriedad. La cardinalidad se refiere al número de ocurrencias de una entidad 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. Si no se muestra el número de la cardinalidad se asume que la ocurrencia es solo una vez.

La obligatoriedad determina que ante la existencia de una entidad deberán o podrán haber ocurrencias de otras relacionadas. Esto se define a través de una línea continua en caso de ser obligatoria la ocurrencia o con una línea discontinua en caso de no serlo. (42)

La descripción de los componentes del modelo de datos ayudará a entender mejor el diagrama que a continuación se presenta:

Modelo de datos

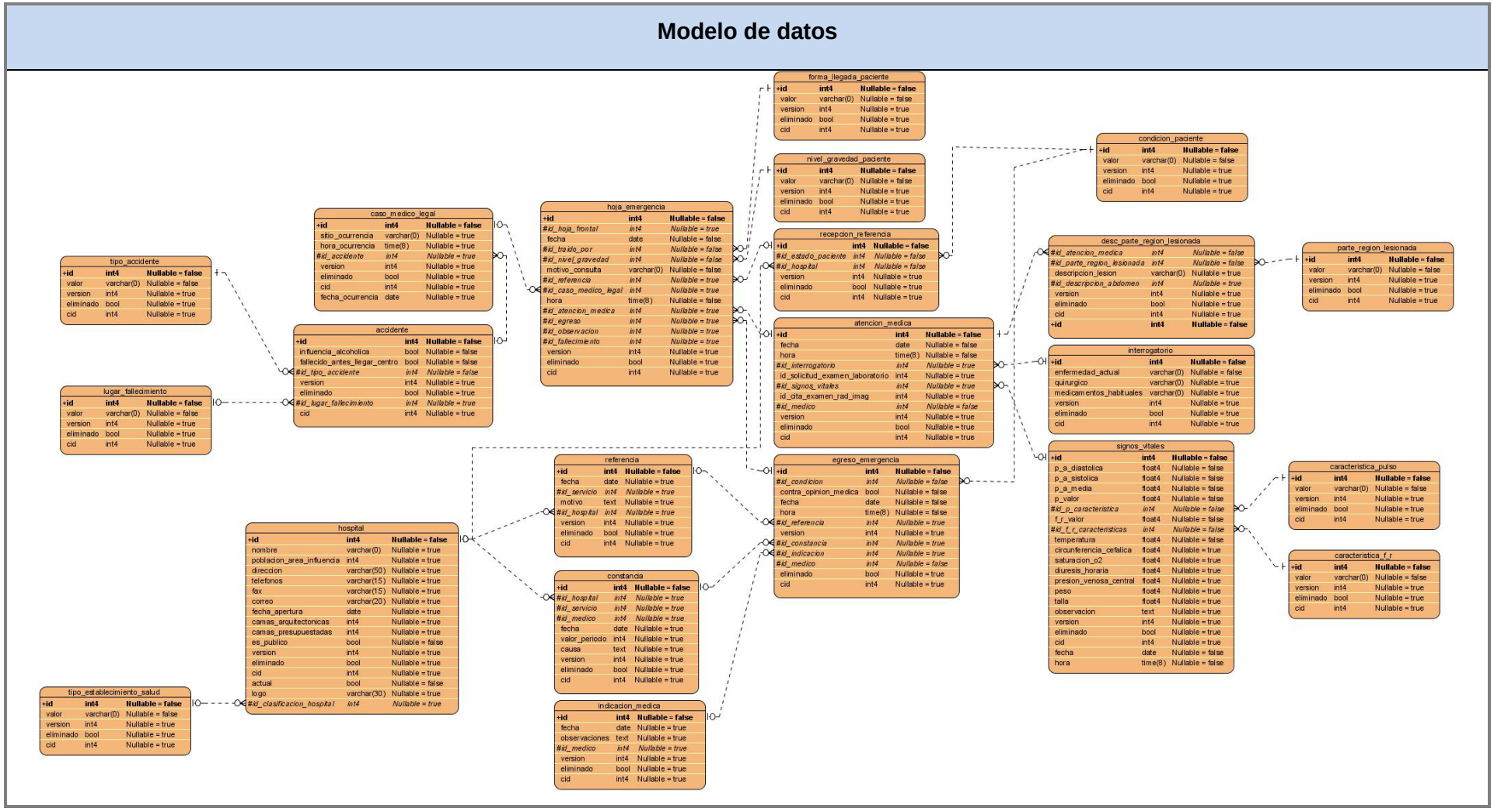


Figura 4.1 Modelo de datos.

4.1.1. Descripción de las tablas de la base de datos.

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

Atributo	Tipo	Descripción
Id.	int4.	Id necesario en cada entidad para las referencias en las relaciones entre tablas.
versión.	int4.	Indica con qué versión de la entidad se está trabajando. Es usado para garantizar que se está trabajando con la versión de la entidad más actualizada que existe en la base de datos.
eliminado.	bool.	Permite la eliminación lógica con que cuenta el sistema, cuando está en verdadero indica que la entidad está eliminada.
cid.	int4.	Permite identificar quién realiza alguna acción sobre la entidad.

Tabla 4.1 Descripción de atributos comunes entre todas las entidades.

hoja_emergencia		
Entidad que recoge la información relacionada con la estancia de un paciente en el área se Emergencias.		
Atributo	**Tipo**	**Descripción**
Fecha	date	Fecha de creación de la entidad.
motivo_consulta	varchar	Razón por la cual el paciente acude al área de Emergencias.
Hora	time(8)	Hora de creación de la entidad.

Tabla 4.2 Descripción de la tabla hoja_emergencia.

caso_medico_legal		
Recoge la información relacionada con un caso que pudiera ser de interés legal.		
Atributo	**Tipo**	**Descripción**
sitio_ocurrencia	varchar	Sitio aproximado donde ocurre el hecho.
hora_ocurrencia	time	Hora aproximada cuando ocurre el hecho.
fecha_ocurrencia	date	Fecha en que ocurre el hecho.

Tabla 4.3 Descripción de la tabla caso_médico_legal.

Accidente		
Recoge la información de los casos médico-legales que son accidentes del tránsito.		
Atributo	**Tipo**	**Descripción**
influencia_alcoholica	bool	Indica si el accidente fue causado por la ingestión de bebidas alcohólicas.
fallecido_antes_llegar_centro	bool	Indica si el paciente muere antes de llegar al centro.

Tabla 4.4 Descripción de la tabla accidente.

recepcion_referencia
Recoge la información de la referencia con que llega un paciente que es enviado desde otro centro.

Tabla 4.5 Descripción de la tabla recepción_referencia.

atencion_medica		
Recoge la información de la atención médica al paciente.		
Atributo	**Tipo**	**Descripción**
Fecha	date	Fecha de creación de la entidad.
Hora	time	Hora de creación de la entidad.

Tabla 4.6 Descripción de la tabla atencion_medica.

egreso_emergencia		
Recoge la información de la salida del paciente de la emergencia por que fue egresado.		
Atributo	**Tipo**	**Descripción**
contra_opinion_medica	bool	Indica si el egreso fue o no contra opinión médica.
Fecha	date	Fecha de creación de la entidad.
Hora	time	Hora de creación de la entidad.

Tabla 4.7 Descripción de la tabla egreso_emergencia.

Referencia		
Recoge la información de la remisión que se le puede hacer a un paciente.		
Atributo	**Tipo**	**Descripción**
Fecha	date	Fecha de creación de la entidad.
Motivo	text	Motivo de la referencia.

Tabla 4.8 Descripción de la tabla referencia.

Constancia		
Recoge la información de una constancia de que el paciente estuvo en el hospital.		
Atributo	**Tipo**	**Descripción**
Fecha	date	Fecha de creación de la entidad.
valor_periodo	int4	Valor del periodo asignado para la justificación.
Causa	text	Causa de la estancia del paciente en el hospital.

Tabla 4.9 Descripción de la tabla constancia.

indicacion_medica		
Recoge la información de la indicación médica que se le puede hacer a un paciente.		
Atributo	**Tipo**	**Descripción**
Fecha	date	Fecha de creación de la entidad.
observaciones	text	Observación médica.

Tabla 4.10 Descripción de la tabla indicacion_medica.

Hospital		
Recoge la información de los hospitales existentes.		
Atributo	**Tipo**	**Descripción**
nombre	varchar	Nombre del centro.
direccion	varchar	Dirección en que se encuentra el centro.
telefonos	varchar	Teléfono del centro.
fax	varchar	Fax del centro.
Correo	varchar	Correo electrónico del centro.
fecha_apertura	date	Fecha de apertura del centro.
es_publico	bool	Indica si el centro es público o privado.
actual	bool	Indica si es el centro donde se encuentra el sistema.
logo	varchar	url del logo del hospital.

Tabla 4.11 Descripción de la tabla hospital.

desc_parte_region_lesionada		
Descripción asociada a la parte lesionada que fue seleccionada.		
Atributo	**Tipo**	**Descripción**
descripcion_lesion	varchar	Descripción de la lesión de la parte lesionada.

Tabla 4.12 Descripción de la tabla desc_parte_region_lesionada.

Interrogatorio		
Recoge la información del interrogatorio que el médico realiza al paciente.		
Atributo	**Tipo**	**Descripción**
enfermedad_actual	varchar	Descripción de la historia de la enfermedad actual.
quirurgico	varchar	Descripción de las intervenciones quirúrgicas.
medicamentos_habituales	varchar	Descripción de los medicamentos que habitualmente consume.

Tabla 4.13 Descripción de la tabla interrogatorio.

signos_vitales		
Recoge la información de los signos vitales tomados al paciente por el médico.		
Atributo	**Tipo**	**Descripción**
fecha	date	Fecha en que se registran los signos vitales.
Hora	time	Hora en que se registran los signos vitales.
p_a_diastolica	float4	Presión arterial diastólica.
p_a_sistolica	float4	Presión arterial sistólica.
p_a_media	float4	Presión arterial media.
p_valor	float4	Valor del pulso.
f_r_valor	float4	Valor de la frecuencia respiratoria.
temperatura	float4	Temperatura corporal.
circunferencia_cefalica	float4	Longitud de la circunferencia cefálica.
saturacion_o2	float4	Valor de la saturación de oxigeno.
diuresis_horaria	float4	Valor de la diuresis horaria del paciente.
presion_venosa_central	float4	Valor de la presión venosa central.
Peso	float4	Valor del peso del paciente.
Talla	float4	Valor de la talla del paciente.
observacion	text	Comentario.

Tabla 4.14 Descripción de la tabla signos_vitales.

Existen entidades que representan los valores posibles que puede tomar un campo, ejemplo: los días de la semana los cueles pueden ser solo uno de los valores: lunes, martes, miércoles, jueves, viernes,

sábado, domingo; estas entidades son llamadas nomencladores y en la implementación a realizar este tipo de entidades tienen los siguientes campos, además de los antes mencionados como comunes para todas las entidades:

Nomenclador		
Atributo	**Tipo**	**Descripción**
Valor	varchar	Valor que puede tomar, siguiendo el ejemplo anterior seria lunes.
Código	varchar	Código que permitirá la identificación del valor independientemente del idioma en que esté la aplicación. Este valor permite la comparación del valor.

Tabla 4.15 Descripción abstracta para las tablas de nomencladores.

4.2. Modelo de implementación.

El modelo de implementación describe cómo los elementos del diseño se implementan en componentes. Entre los componentes se puede encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe entre los paquetes y clases del modelo de diseño y los subsistemas y componentes físicos.

El propósito del modelo de implementación es definir la organización del código, planificar las integraciones de sistema necesarias en cada iteración e implementar las clases y subsistemas encontrados durante el Diseño.

Se debe proponer una estrategia de codificación que defina los formatos para la asignación de nombres a las variables, estilo de programación y métodos de documentación. (43)

4.2.1. Diagramas de Componentes.

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Estos muestran las opciones de realización como el código fuente, binario y ejecutable. Los componentes representan todos los tipos de elementos software que entran en la fabricación de aplicaciones

informáticas. Pueden ser simples archivos, paquetes o bibliotecas cargadas dinámicamente. Estos tienen relaciones de traza con los elementos del modelo de diseño.

De los estereotipos estándar que se aplican a los componentes según el lenguaje de modelado UML se emplean: *library y file,* los cuales representan, una biblioteca de objetos estática o dinámica y un documento que contiene código fuente, respectivamente.

Las relaciones de dependencia se utilizan en los diagramas de componentes para indicar que un componente se refiere a los servicios ofrecidos por otro componente. Los distintos componentes pueden agruparse en paquetes según un criterio lógico y con vistas a simplificar la implementación. Estos paquetes son estereotipados como <<subsistemas>>.

Cada subsistema puede contener componentes y otros subsistemas. La descomposición en subsistemas no es necesariamente una descomposición funcional. La relación entre paquetes y clases en el nivel lógico es el que existe entre subsistemas y componentes en el nivel físico. (44)

A continuación se exponen los Diagramas de Componentes asociados a varios de los subsistemas de implementación identificados. Siguiendo la arquitectura en capas la estructuración en subsistemas de implementación es la siguiente:

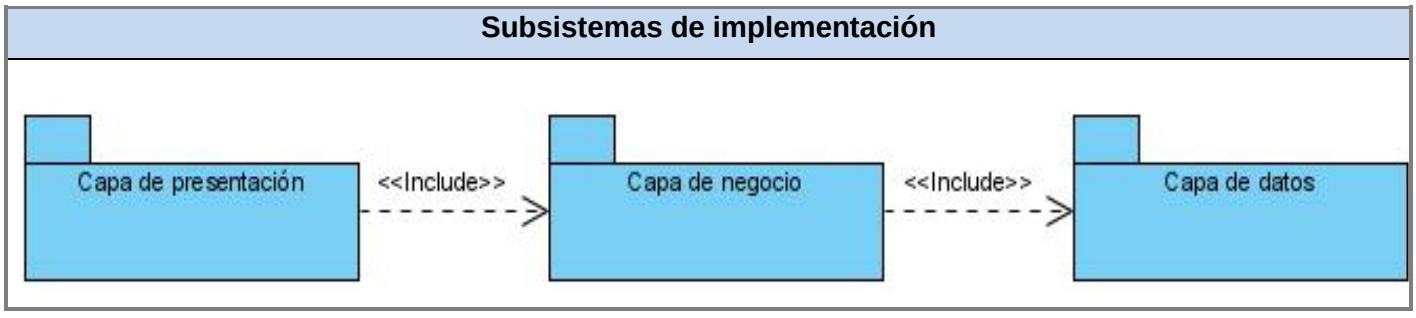


Figura 4.2 Subsistemas de implementación.

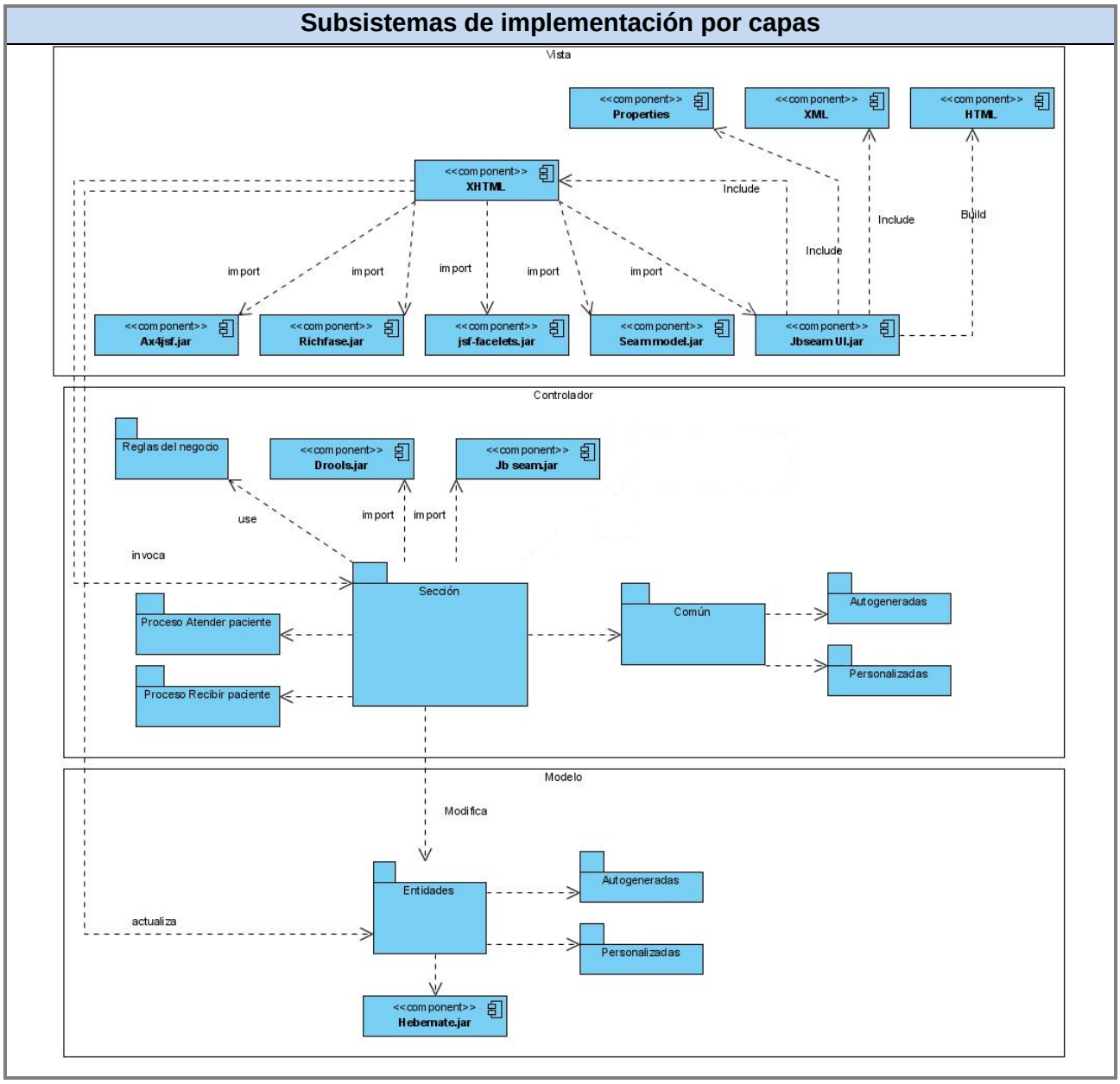


Figura 4.3 Subsistemas de implementación por capas.

4.2.2. Diagramas de despliegue.

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

Los estereotipos permiten precisar la naturaleza del equipo:
• **Procesadores**: Nodo con capacidad de procesamiento. Puede ejecutar un componente.
• **Dispositivos**: Nodo sin capacidad de procesamiento. Representa cualquier otro dispositivo hardware. (45)

La aplicación está distribuida en tres nodos: uno representa las PCs clientes las cuales pueden tener instalado cualquier sistema operativo, el segundo se corresponde con el servidor de aplicaciones JBoss AS 2.0 que está conectado punto a punto con el tercer nodo que es el servidor de datos Postgresql-server 8.3. A la PC cliente está conectado un dispositivo de impresión mediante conexión USB.

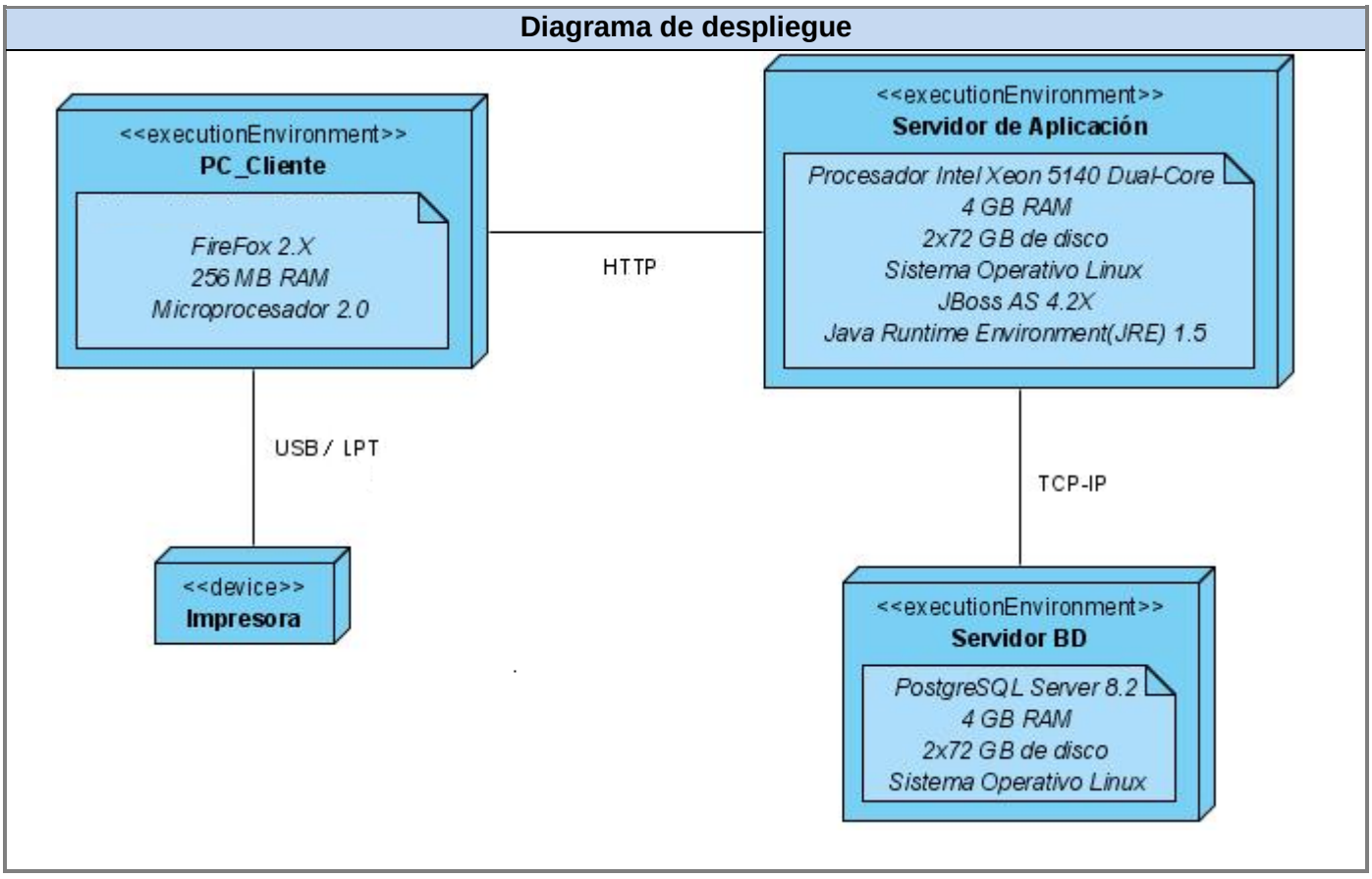


Figura 4.4 Diagrama de despliegue.

4.3. Tratamiento de errores.

Durante el tiempo de ejecución de un sistema pueden fracasar diferentes rutinas; es esto a lo que comúnmente se le llama excepción. Mediante el tratamiento de excepciones se restaura un estado en el que la rutina pueda seguir la ejecución, lo que permite obtener un sistema más robusto y fiable.
En el sistema propuesto, el control de las excepciones se lleva a cabo a toda porción de código, donde pueda surgir alguna situación inesperada, especialmente donde se ejecutan sentencias que manipulan los datos que viajan desde y hacia la base de datos. También se controlan los errores que pueden surgir en la

validación de datos provenientes de la interfaz de usuario, puesto que encierran una lógica compleja en cierta medida.

Para el manejo de las excepciones o errores, en las clases controladoras de procesos, se utilizará el bloque try para detectar cuando ocurra algún fallo y mediante el catch se manejarán dichas excepciones, mediante mensajes que se muestran en la interfaz de usuario, por las facilidades que brinda el FacesMessages, componente del framework Seam.

Ejemplo:

```
try
{
   //declaración que causa la excepción
}
```

Entre las llaves de try se escribe el código que hará funcionar el programa. Para capturar la excepción que puede generar este código se necesita otra instrucción llamada **catch** (capturar).

```
catch(NombredeExcepcion obj){
  //código para tratar el error
}
```

Entre las llaves de catch se escribe el código que se quiera para tratar el error.

Existe además un archivo XML, denominado page.xml, que engloba la configuración de todos los mensajes que se deben mostrar por cada tipo de excepción, así como la página a la que el sistema redirecciona en caso de la aparición de un error sorpresivo. (46)

4.4. Seguridad.

La informatización de los procesos de atención al paciente en el área de Emergencias involucra problemas éticos y legales relacionados con la confidencialidad y seguridad de la información que se manipula, por esto debe tenerse un gran control para garantizar que la misma no sea accedida ni modificada por personas que no tengan acceso a ella.

Las funcionalidades del sistema encargadas de la seguridad son: iniciar y cerrar sesión de trabajo, registrar trazas y administrar seguridad.

Para iniciar la sesión de trabajo un usuario debe acceder al sistema e insertar su nombre de usuario y contraseña. El sistema verifica que los datos introducidos sean válidos y dados los permisos de este usuario tendrá acceso al módulo al que desea entrar. Para terminar las tareas realizadas en el sistema, este permite cerrar sesión y salir del módulo.

El registro de trazas es vital en el sistema ya que permite archivar las acciones que realiza el usuario sobre el este, que pueden ser: inicio o cierre de sesión, acceso a un módulo, modificación de un atributo de una entidad o cualquier otra operación. En cualquiera de los casos anteriores la aplicación registra una traza en la base de datos. Se hace necesario administrar los permisos que se asignan a los usuarios para la navegación en el sistema y ello se logra a través de la funcionalidad: administrar seguridad. El sistema brinda la posibilidad de asignar o denegar permiso a roles y usuarios en las funcionalidades de los módulos.

4.5. Estrategias de codificación. Estándares y estilos a utilizar.

Los estándares de codificación son reglas que se aplican para logar uniformidad en el código producido por un grupo de desarrollo de un sistema. Estos reducen perceptiblemente el riesgo de que los desarrolladores introduzcan errores. Los estándares de codificación no destapan problemas existentes, evitan más bien que los errores ocurran, lo que permite obtener un código de alta calidad.

Usar técnicas de codificación sólidas y realizar buenas prácticas de programación, con vistas a generar un código de alta calidad, es de gran importancia para la calidad del *software*. La aplicación de estándares de codificación además posibilita que el software que se obtiene sea fácil de comprender y de mantener en el tiempo. (47)

Para la solución del problema tratado en este trabajo se ha utilizado el estándar de la SUN Microsystems (Stanford University Network) para Java, con la utilización la Notación Camello, para denotar variables, parámetros y métodos.

4.5.1. Variables, parámetros y métodos.

En esta notación, el identificador para las variables, los parámetros y los métodos se define escribiendo las palabras de la siguiente forma, la primera con minúsculas y a partir de la segunda palabra, en caso de existir, con letra inicial mayúscula, ejemplo: *int cantidadReal; public void aceptar().*

A continuación se especifican algunas restricciones para la nomenclatura, basada en el estándar a utilizar:

4.5.2. Identación.

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

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

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

Ubicación de comentarios: Se debe comentar al inicio de la clase o función especificando el objetivo de la misma así como los parámetros que usa (especificar tipos de datos, y objetivo del parámetro).

Líneas en blanco: Se debe 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 deben usar espacios en blanco entre estos operadores para lograr una mayor legibilidad en el código. Ejemplo: producto = nomproducto.

4.5.4. Constantes.

Apariencia de constantes: Se deben declarar las constantes con todas sus letras en mayúscula.

En el capítulo, se obtuvieron los artefactos modelo de datos, subsistemas de implementación, diagrama de despliegue. Una vez concluido el proceso de implementación del sistema se cuenta con un producto acabado y totalmente funcional. El mismo se encuentra correctamente documentado y cumple los requerimientos de seguridad necesarios para garantizar la confidencialidad y privacidad de la información que se manipula en todo el sistema. En el proceso de codificación se cumplió con los estándares definidos, lo que permitió obtener un sistema fácil de mantener en el transcurso del tiempo.

Conclusiones.

El modelado de los procesos del negocio permitió depurar las actividades manuales y definir aquellas que fuesen funcionalidades del sistema.

Los Sistemas de Información Hospitalaria existentes presentan licencias de software propietario y corren sobre el sistema operativo Windows, por lo que no cumplen los requisitos necesarios.

El empleo de la arquitectura definida permite el desarrollo de un sistema robusto y flexible.

La identificación de las clases, los atributos y métodos cobran gran importancia al brindarle al desarrollador una idea clara de lo que se debe implementar.

La implementación del sistema permitió obtener un producto acabado y totalmente funcional. El mismo cumple los requerimientos de seguridad necesarios para garantizar la confidencialidad y privacidad de la información que se manipula en todo el sistema.

RECOMENDACIONES.

- Crear un sistema de ayuda en línea para los usuarios que interactuarán con la aplicación.
- Realizar la integración con el sistema alas RIS y garantizar así la gestión de las citas para estudios radiológicos e imagenológicos.
- Implementar las funciones necesarias para la creación e interpretación de documentos CDA (Clinical Document Architecture), lo cual permitirá la comunicación con otras aplicaciones que tengan incorporado dicho estándar.
- Registrar la atención brindada a las mujeres que se le realiza el parto en el área de Emergencias.

REFERENCIAS BIBLIOGRÁFICAS.

1. Cañedo Andalia, Lic. Rubén. D. R. (5 de Octubre de 2005). *Acimed. Revista cubana de los profesionales de la información y de la comunicación en salud*. Recuperado el 22 de Diciembre de 2008, de La Informática, la Computación y la Ciencia de la Información: una alianza para el desarrollo: http://bvs.sld.cu/revistas/aci/vol13_5_05/aci07505.htm

2. Sánchez Mansolo, Dr. Athos A. D. O. (s.f.). *Revista cubana de educación médica superior*. Recuperado el 12 de Febrero de 2009, de Registro Electrónico de Pacientes: http://www.bvs.sld.cu/revistas/ems/vol13_1_99/ems07199.htm

3. König, S. (s.f.). *Sistemas de información*. Recuperado el 22 de Diciembre de 2008, de El hospital como organización: http://socrates.ieem.edu.uy/articulos/archivos/387_sistemas_de_informacion.pdf

4. Millán, E. M. (28 de Abril de 2007). *Gestión de la información y la comunicación en emergencias, desastres y crisis sanitarias*. Recuperado el 23 de Diciembre de 2008, de http://www.semes.org/revista/vol20_2/9.pdf

5. Farrera, M. A. (s.f.). *Sistemas de Información para Hospitales*. Recuperado el 26 de Diciembre de 2008, de http://genesis.uag.mx/posgrado/revistaelect/compu/his_archivos/frame.htm

6. *Sigho. Sistema de información para la gerencia hospitalaria.* (2006). Recuperado el 27 de Diciembre de 2008, de http://www.ssn.gob.mx/html/sigho/sigho.html

7. Cira Rodríguez, César. (28 de Enero de 2005). *Crean software para beneficio de salud pública* . Recuperado el 26 de Diciembre de 2008, de http://www.idict.cu/UserFiles/File/Boletines/Novedades/boletin02_0105/especial020105.html

8. *Grupo Prides*. (s.f.). Recuperado el 26 de Dicimbre de 2008, de http://www.tools.co.cr/PRIDES/productos/em.html

9. ISOFT Sanidad, S. (2006). *Isoft*. Recuperado el 27 de Diciembre de 2008, de http://www.nuance.com/healthcare/pdf/cs_healthcare_Nuestra.pdf

10. *Desarrolloweb*. (30 de Agosto de 2007). Recuperado el 26 de Diciembre de 2008, de Arquitectura cliente-servidor: http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html

11. Sánchez González, Carlos. (28 de Septiembre de 2004). *Aplicaciones en capas. Capitulo 3.* Recuperado el 25 de Diciembre de 2008, de Aplicaciones en capas: http://oness.sourceforge.net/proyecto/html/ch03s02.html

12. Degiovan, Marcio. (10 de Febrero de 2007). *JavaHispano*. Recuperado el 28 de Diciembre de 2008, de Comparativa de frameworks Web: http://www.javahispano.org/contenidos/es/comparativa_de_frameworks_web/

13. Maldonado, Daniel M. (28 de Septiembre de 2007). *El CoDiGo K*. Recuperado el 28 de Diciembre de 2008, de Arquitectura de programación en 3 capas: http://www.elcodigok.com.ar/2007/09/arquitectura-de-programacion-en-3-capas/

14. *Jboss.org*. (2008). Recuperado el 27 de Diciembre de 2008, de Community Documentation. Introducción: http://www.jboss.org/file-access/default/members/jbossrichfaces/freezone/docs/devguide/en/html/Introduction.html

15. *Asynchronous Javascript And XML (AJAX)*. (2 de Enero de 2008). Recuperado el 27 de Diciembre de 2008, de http://tutoriales.maborak.com/ajax/

16. *JBoss Community*. (2007). Recuperado el 28 de Diciembre de 2008, de JBoss Ajax4jsf. Introducción: http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html

17. Hookom, Jacob. (17 de Agosto de 2005). *JSF Central tm*. Recuperado el 28 de Diciembre de 2008, de Inside Facelets Part 1: An Introduction: http://www.jsfcentral.com/articles/facelets_1.html

18. Egíluz Pérez, Javier. (s.f.). *Librosweb.es*. Recuperado el 28 de Diciembre de 2008, de HTML y XHTML. Capítulo 1: Introducción : http://www.librosweb.es/xhtml/capitulo1/html_y_xhtml.html

19. *ignside.net*. (14 de Enero de 2007). Recuperado el 2008 de Diciembre de 2008, de Cascade Style Sheets: http://www.ignside.net/man/css/index.php

20. Ver referencia número 13.

21. *Web Application - Plataforma J2EE*. (20 de Febrero de 2008). Recuperado el 28 de Diciembre de 2008, de JBoss Seam Framework: http://wilmanchamba.wordpress.com/2008/02/20/jboss-seam-framework/

22. *Java en castellano*. (3 de Junio de 2005). Recuperado el 28 de Diciembre de 2008, de Liberado Drools 2.0: http://www.programacion.com/java/noticia/1342/

23. Ver referencia número 13.

24. *Hibernate.org*. (s.f.). Recuperado el 30 de Enero de 2009, de Hibernate Tools for Eclipse and Ant: http://www.hibernate.org/255.html

25. Hennebrueder, Sebastian. (15 de Marzo de 2006). *Laliluna. Java tutorials and development*. Recuperado el 29 de Enero de 2009, de First EJB 3 Tutorial showing a session and entity beans with annotations and JBoss.: http://www.laliluna.de/ejb-3-tutorial-jboss.html

26. Ort, R. B. (Mayo de 2006). *Sun microsystems*. Recuperado el 29 de Enero de 2009, de The Java Persistence API - A Simpler Programming Model for Entity Persistence: http://java.sun.com/developer/technicalArticles/J2EE/jpa/

27. Franky, María Consuelo. (Abril de 2007). *Java EE 5 (sucesor de J2EE):*. Recuperado el 29 de Enero de 2009, de http://www.acis.org.co/fileadmin/Conferencias/ConfConsueloFranky_Abr19.pdf

28. Lucifer, Prometeo. (s.f.). Recuperado el 28 de Enero de 2009, de Java Runtime Environment - JRE: http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre/

29. *Kabuntu-es*. (s.f.). Recuperado el 28 de Enero de 2009, de Programas de desarrollo libres: http://www.kubuntu-es.org/wiki/desarrollo-programacion/programas-desarrollo-libres

30. Ottinger, Joseph. (25 de Junio de 2007). *TheServerSide.com*. Recuperado el 27 de Enero de 2009, de JBoss releases JBoss Tools, Eclipse Plugins including Exadel: http://www.theserverside.com/news/thread.tss?thread_id=45933

31. Jaramillo, Wilmer. (2006). *Software Libre de Venezuela 777, C.A.* Recuperado el 27 de Enero de 2009, de http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf

32. *Vial Ovalle*. (20 de Enero de 2007). Recuperado el 29 de Enero de 2009, de Sistema de gestión de base de datos: http://vialovalle.blogcindario.com/2007/01/00061-sistema-de-gestion-de-bases-de-datos.html

33. tldp.org. (23 de Septiembre de 2006). *Manuales de Ayuda.com*. Recuperado el 27 de Enero de 2009, de Breve historia de PostgreSQL : http://www.manualesdeayuda.com/manuales/bases-de-datos/postgresql/breve-historia-de-postgresql-01831.html

34. *Free Download Manager*. (5 de Marzo de 2007). Recuperado el 29 de Enero de 2009, de Paradigma visual para UML (Plataforma Java) (Visual Paradigm for UML [Java Platform]) 6.0 : http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/

35. *Itera*. (29 de Julio de 2008). Recuperado el 28 de Enero de 2008, de Rational Unified Process: http://www.iteraprocess.com/index.php?option=com_content&task=view&id=18&Itemid=42

36. Mora, Francisco. (2003). *UML: Lenguaje Unificado de Modelado*. Recuperado el 27 de Diciembre de 2008, de http://www.dccia.ua.es/dccia/inf/asignaturas/GPS/archivos/Uml.PDF

37. *Milestone consulting*. (s.f.). Recuperado el 27 de Enero de 2009, de El nuevo estándar de modelado de negocio llega por primera vez a México gracias a Milestone Consulting. En ningún otro lugar encontrarás un curso tan completo y actualizado para tus procesos de negocio.: http://www.milestone.com.mx/CursoModeladoNegociosBPMN.htm

38. *Especificaciones De Requerimientos*. (s.f.). Recuperado el 15 de Febrero de 2009, de http://www.mitecnologico.com/Main/EspecificacionesDeRequerimientos

39. Ferzández Vilas, Ana. (20 de Marzo de 2001). *Comportamiento del sistema*. Recuperado el 30 de Enero de 2009, de http://tvdi.det.uvigo.es/~avilas/UML/node24.html

40. *El Mundo Informático*. (8 de Mayo de 2007). Recuperado el 23 de Febrero de 2009, de PatronesGgraps. (Patrones de software para la asignación General de Responsabilidad). Parte II.: http://jorgesaavedra.wordpress.com/category/patrones-grasp/

41. *Software y Aplicaciones Web* . (14 de Febrero de 2008). Recuperado el 24 de Marzo de 2008, de UML Diagramas: http://www.jtentor.com.ar/post/UML-Diagramas.aspx

42. Quintero, Juan Bernardo. (s.f.). *La ingeniería de software aplicada a las bases de datos*. Recuperado el 26 de Marzo de 2009, de http://docencia.udea.edu.co/ingenieria/ArquitecturaSoftware/documentos/LaingenieriadesoftwareenlasBD.pdf

43. *Modelo de Implementación:Diagramas de Componentes y Despliegue*. (s.f.). Recuperado el 15 de Marzo de 2009, de http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf

44. Ver referencia número 43.

45. Ver referencia número 43.

46. *WEBTutoriales*. (24 de Noviembre de 2007). Recuperado el 23 de Marzo de 2009, de Tratamiento de errores en Java con try y catch: http://www.webtutoriales.com/tutoriales/programacion/java/try-and-catch.37.html

47. *MSDN*. (s.f.). Recuperado el 23 de Marzo de 2009, de Revisiones de código y estándares de codificación: http://msdn.microsoft.com/es-es/library/aa291591(VS.71).aspx

BIBLIOGRAFÍA.

- **2005.** Java en castellano. *Liberado Drools 2.0.* [En línea] 3 de Junio de 2005. [Citado el: 28 de Diciembre de 2008.] http://www.programacion.com/java/noticia/1342/.

- **2006.** Sigho. Sistema de información para la gerencia hospitalaria. [En línea] 2006. [Citado el: 27 de Diciembre de 2008.] http://www.ssn.gob.mx/html/sigho/sigho.html.

- **2007.** Desarrolloweb. *Arquitectura cliente-servidor.* [En línea] 30 de Agosto de 2007. [Citado el: 26 de Diciembre de 2008.] http://www.desarrolloweb.com/articulos/arquitectura-cliente-servidor.html.

- **2007.** El Mundo Informático. *PatronesGgraps. (Patrones de software para la asignación General de Responsabilidad). Parte II.* [En línea] 8 de Mayo de 2007. [Citado el: 23 de Febrero de 2009.] http://jorgesaavedra.wordpress.com/category/patrones-grasp/.

- **2007.** Free Download Manager. *Paradigma visual para UML (Plataforma Java) (Visual Paradigm for UML [Java Platform]) 6.0 .* [En línea] 5 de Marzo de 2007. [Citado el: 29 de Enero de 2009.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%5Bcuenta_de_Plataforma_de_Java_14715_p/.

- **2007.** ignside.net. *Cascade Style Sheets.* [En línea] 14 de Enero de 2007. [Citado el: 2008 de Diciembre de 2008.] http://www.ignside.net/man/css/index.php.

- **2007.** JBoss Community. *JBoss Ajax4jsf. Introducción.* [En línea] 2007. [Citado el: 28 de Diciembre de 2008.] http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html.

- **2007.** Vial Ovalle. *Sistema de gestión de base de datos.* [En línea] 20 de Enero de 2007. [Citado el: 29 de Enero de 2009.] http://vialovalle.blogcindario.com/2007/01/00061-sistema-de-gestion-de-bases-de-datos.html.

- **2007.** WEBTutoriales. *Tratamiento de errores en Java con try y catch.* [En línea] 24 de Noviembre de 2007. [Citado el: 23 de Marzo de 2009.] http://www.webtutoriales.com/tutoriales/programacion/java/try-and-catch.37.html.

- **2008.** Asynchronous Javascript And XML (AJAX). [En línea] 2 de Enero de 2008. [Citado el: 27 de Diciembre de 2008.] http://tutoriales.maborak.com/ajax/.

  2008. Itera. *Rational Unified Process.* [En línea] 29 de Julio de 2008. [Citado el: 28 de Enero de 2008.] http://www.iteraprocess.com/index.php?option=com_content&task=view&id=18&Itemid=42

- **2008.** Jboss.org. *Community Documentation. Introducción.* [En línea] 2008. [Citado el: 27 de Diciembre de 2008.] http://www.jboss.org/file-access/default/members/jbossrichfaces/freezone/docs/devguide/en/html/Introduction.html.

- **2008.** Software y Aplicaciones Web . *UML Diagramas.* [En línea] 14 de Febrero de 2008. [Citado el: 24 de Marzo de 2008.] http://www.jtentor.com.ar/post/UML-Diagramas.aspx.

- **2008.** Web Application - Plataforma J2EE. *JBoss Seam Framework.* [En línea] 20 de Febrero de 2008. [Citado el: 28 de Diciembre de 2008.] http://wilmanchamba.wordpress.com/2008/02/20/jboss-seam-framework/.

- **Alemán Antelo, Lorena. 2008.** IH-SW-DR-008 ALAS-HIS_Emergencia_Procesos elementales del negocio_Procesos actuales. Cuba : UCI, 2008.

- **—. 2008.** IH-SW-DR-069 ALAS-HIS_Emergencias_Modelo de casos de uso del sistema. Cuba : UCI, 2008.

- **Bauer, Christian y otros. 2007.** Java Persistence with Hibernate. 2007.

- **Biswas, Rahul. 2006 .** Sun microsystems. *The Java Persistence API - A Simpler Programming Model for Entity Persistence.* [En línea] Mayo de 2006 . [Citado el: 29 de Enero de 2009.] http://java.sun.com/developer/technicalArticles/J2EE/jpa/.

- **Cañedo Andalia, Lic. Rubén. 2005.** Acimed. Revista cubana de los profesionales de la información y de la comunicación en salud. *La Informática, la Computación y la Ciencia de la Información:.* [En línea] 5 de Octubre de 2005. [Citado el: 22 de Diciembre de 2008.] http://bvs.sld.cu/revistas/aci/vol13_5_05/aci07505.htm.

- **Degiovan, Marcio. 2007.** JavaHispano. *Comparativa de frameworks Web.* [En línea] 10 de Febrero de 2007. [Citado el: 28 de Diciembre de 2008.] http://www.javahispano.org/contenidos/es/comparativa_de_frameworks_web/.

- **Eguíluz Pérez, Javier.** Librosweb.es. *Introducción a XHTML. Capítulo 1: Introducción .* [En línea] [Citado el: 28 de Diciembre de 2008.] http://www.librosweb.es/xhtml/capitulo1.html.

- **Fernández Vilas, Ana. 2001.** Comportamiento del sistema. [En línea] 20 de Marzo de 2001. [Citado el: 30 de Enero de 2009.] http://tvdi.det.uvigo.es/~avilas/UML/node24.html.

- **Franky, María Consuelo. 2007.** Java EE 5 (sucesor de J2EE):. [En línea] Abril de 2007. [Citado el: 29 de Enero de 2009.] http://www.acis.org.co/fileadmin/Conferencias/ConfConsueloFranky_Abr19.pdf.

- **García Peñalvo, Francisco José. 2005.** Ingeniería del Software. *Tema 5: Principios del diseño del software.* [En línea] 23 de Noviembre de 2005. [Citado el: 21 de Marzo de 2009.] http://zarza.usal.es/~fgarcia/docencia/isoftware/05-06/Tema8.zip.

- Grupo Prides. [En línea] [Citado el: 26 de Dicimbre de 2008.] http://www.tools.co.cr/PRIDES/productos/em.html.

- **Hennebrueder, Sebastian. 2006.** Lalíluna. Java tutorials and development. *First EJB 3 Tutorial showing a session and entity beans with annotations and JBoss.* [En línea] 15 de Marzo de 2006. [Citado el: 29 de Enero de 2009.] http://www.lalíluna.de/ejb-3-tutorial-jboss.html.

- Hibernate.org. *Hibernate Tools for Eclipse and Ant.* [En línea] [Citado el: 30 de Enero de 2009.] http://www.hibernate.org/255.html.

- **Hookom, Jacob. 2005 .** JSF Central tm. *Inside Facelets Part 1: An Introduction.* [En línea] 17 de Agosto de 2005 . [Citado el: 28 de Diciembre de 2008.] http://www.jsfcentral.com/articles/facelets_1.html.

- **ISOFT Sanidad, S.A. 2006.** Isoft. [En línea] 2006. [Citado el: 27 de Diciembre de 2008.] http://www.nuance.com/healthcare/pdf/cs_healthcare_Nuestra.pdf.

- **Jacobson, Ivar y otros. 2000.** El Proceso Unificado de Desarrollo de Software. Madrid : Pearson Educación , 2000.

- **Jaramillo, Wilmer. 2006.** Software Libre de Venezuela 777, C.A. [En línea] 2006. [Citado el: 27 de Enero de 2009.] http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf.

- Kabuntu-es. *Programas de desarrollo libres.* [En línea] [Citado el: 28 de Enero de 2009.] http://www.kubuntu-es.org/wiki/desarrollo-programacion/programas-desarrollo-libres.

- **König, Sergio.** Sistemas de información. *El hospital como organización.* [En línea] [Citado el: 22 de Diciembre de 2008.] http://socrates.ieem.edu.uy/articulos/archivos/387_sistemas_de_informacion.pdf.

- **Larma, Craig. 1999.** UML y patrones. Tomo I. s.l. : Prentice Hall Iberoamericana, 1999, pág. 499.

- **Lucifer, Prometeo.** *Java Runtime Environment - JRE.* [En línea] [Citado el: 28 de Enero de 2009.] http://www.elleonplateadodeojosrojos.es/blog/java-runtime-environment-jre/.

- **Maldonado, Daniel M. 2007.** El CoDiGo K. *Arquitectura de programación en 3 capas.* [En línea] 28 de Septiembre de 2007. [Citado el: 28 de Diciembre de 2008.] http://www.elcodigok.com.ar/2007/09/arquitectura-de-programacion-en-3-capas/.

- **Martínez Farrera, Manuel A.** Sistemas de Información para Hospitales. [En línea] [Citado el: 26 de Diciembre de 2008.] http://genesis.uag.mx/posgrado/revistaelect/compu/his_archivos/frame.htm.

- Milestone consulting. *El nuevo estándar de modelado de negocio llega por primera vez a México gracias a Milestone Consulting. En ningún otro lugar encontrarás un curso tan completo y actualizado para tus procesos de negocio.* [En línea] [Citado el: 27 de Enero de 2009.] http://www.milestone.com.mx/CursoModeladoNegociosBPMN.htm.

- Modelo de Implementación:Diagramas de Componentes y Despliegue. [En línea] [Citado el: 15 de Marzo de 2009.] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

- **Mora, Francisco. 2003.** UML: Lenguaje Unificado de Modelado. [En línea] 2003. [Citado el: 27 de Diciembre de 2008.] http://www.dccia.ua.es/dccia/inf/asignaturas/GPS/archivos/Uml.PDF.

- **Moreno Millán, Emilio. 2007.** Gestión de la información y la comunicación en emergencias, desastres y crisis sanitarias. [En línea] 28 de Abril de 2007. [Citado el: 23 de Diciembre de 2008.] http://www.semes.org/revista/vol20_2/9.pdf.

- **Morales, Sergio. 2008.** Proyecto no.1 de programación de la ingeniería del software y la aequitectura del software. [En línea] 5 de Febrero de 2008. [Citado el: 2 de Marzo de 2009.] http://carloscar7.files.wordpress.com/2008/02/carlos-david-carballo-espana.doc.

- MSDN. *Revisiones de código y estándares de codificación.* [En línea] [Citado el: 23 de Marzo de 2009.] http://msdn.microsoft.com/es-es/library/aa291591(VS.71).aspx.

- **Ottinger, Joseph. 2007 .** TheServerSide.com. *JBoss releases JBoss Tools, Eclipse Plugins including Exadel.* [En línea] 25 de Junio de 2007 . [Citado el: 27 de Enero de 2009.] http://www.theserverside.com/news/thread.tss?thread_id=45933.

- **Quintero, Juan Bernardo.** La ingeniería de software aplicada a las bases de datos. [En línea] [Citado el: 26 de Marzo de 2009.] http://docencia.udea.edu.co/ingenieria/ArquitecturaSoftware/documentos/LaingenieriadesoftwareenlasBD.pdf.

- **Rodríguez César, Cira. 2005.** Crean software para beneficio de salud pública . [En línea] 28 de Enero de 2005. [Citado el: 26 de Diciembre de 2008.] http://www.idict.cu/UserFiles/File/Boletines/Novedades/boletin02_0105/especial020105.html.

- **Romero, Danielle.** CAPAS DE SESIÓN, PRESENTACIÓN Y APLICACIÓN. [En línea] [Citado el: 25 de Diciebre de 2008.] http://www.elrinconcito.com/articulos/Sesiones/sesiones.pdf.

- **Sánchez González, Carlos. 2004.** Un proyecto open source para el negocio textil mayorista desarrollado con tecnologías open source innovadoras. *Aplicaciones en capas.* [En línea] 28 de Septiembre de 2004. [Citado el: 25 de Diciembre de 2008.] http://oness.sourceforge.net/proyecto/html/ch03s02.html.

- **Sánchez Mansolo, Dr. Athos A. y otros.** Revista cubana de educación médica superior. *Registro Electrónico de Pacientes.* [En línea] [Citado el: 12 de Febrero de 2009.] http://www.bvs.sld.cu/revistas/ems/vol13_1_99/ems07199.htm.

- **tldp.org. 2006 .** Manuales de Ayuda.com. *Breve historia de PostgreSQL .* [En línea] 23 de Septiembre de 2006 . [Citado el: 27 de Enero de 2009.] http://www.manualesdeayuda.com/manuales/bases-de-datos/postgresql/breve-historia-de-postgresql-01831.html.

- **Weitzenfeld, Alfredo.** Ingeniería de software orientada a objetos con UML, Java e Internet. [En línea] [Citado el: 29 de Marzo de 2009.] http://books.google.com.cu/books?id=MOviEp0ApQcC&pg=PA523&lpg=PA523&dq=modelo+de+implementaci%C3%B3n&source=bl&ots=OVwQi7Rufv&sig=Xi24ijGqGwyH0a6MMB-Tju8XXFk&hl=es&ei=blLeSbjkJ5LrlQff0eBa&sa=X&oi=book_result&ct=result&resnum=1#PPA10,M1.

GLOSARIO DE TÉRMINOS.

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

API: Interfaz de Programación de Aplicaciones, cuyo acrónimo en inglés es API *(Application Programming Interface)*, es un conjunto de funciones residentes en bibliotecas generalmente dinámicas. Permiten que una aplicación corra bajo un determinado sistema operativo.

CIE10: es la décima versión de la Clasificación Estadística Internacional de Enfermedades y otros Problemas de Salud. Provee los códigos para clasificar las enfermedades y una amplia variedad de signos, síntomas, hallazgos anormales, denuncias, circunstancias sociales y causas externas de daños y/o enfermedad. Cada condición de salud puede ser asignada a una categoría y darle un código de hasta seis caracteres de longitud.

Estándar: es una especificación que regula la realización de ciertos procesos o la fabricación de componentes para garantizar la interoperabilidad.

HL7: acrónimo de Health Level Seven. Es un conjunto de estándares para el intercambio electrónico de información médica. Los estándares HL7 son desarrollados por la organización ANSI del mismo nombre.

HTML: acrónimo de HyperText Markup Language (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de "etiquetas", rodeadas por corchetes angulares (<,>). HTML también puede describir, hasta un cierto

punto, la apariencia de un documento, y puede incluir un script (por ejemplo Javascript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

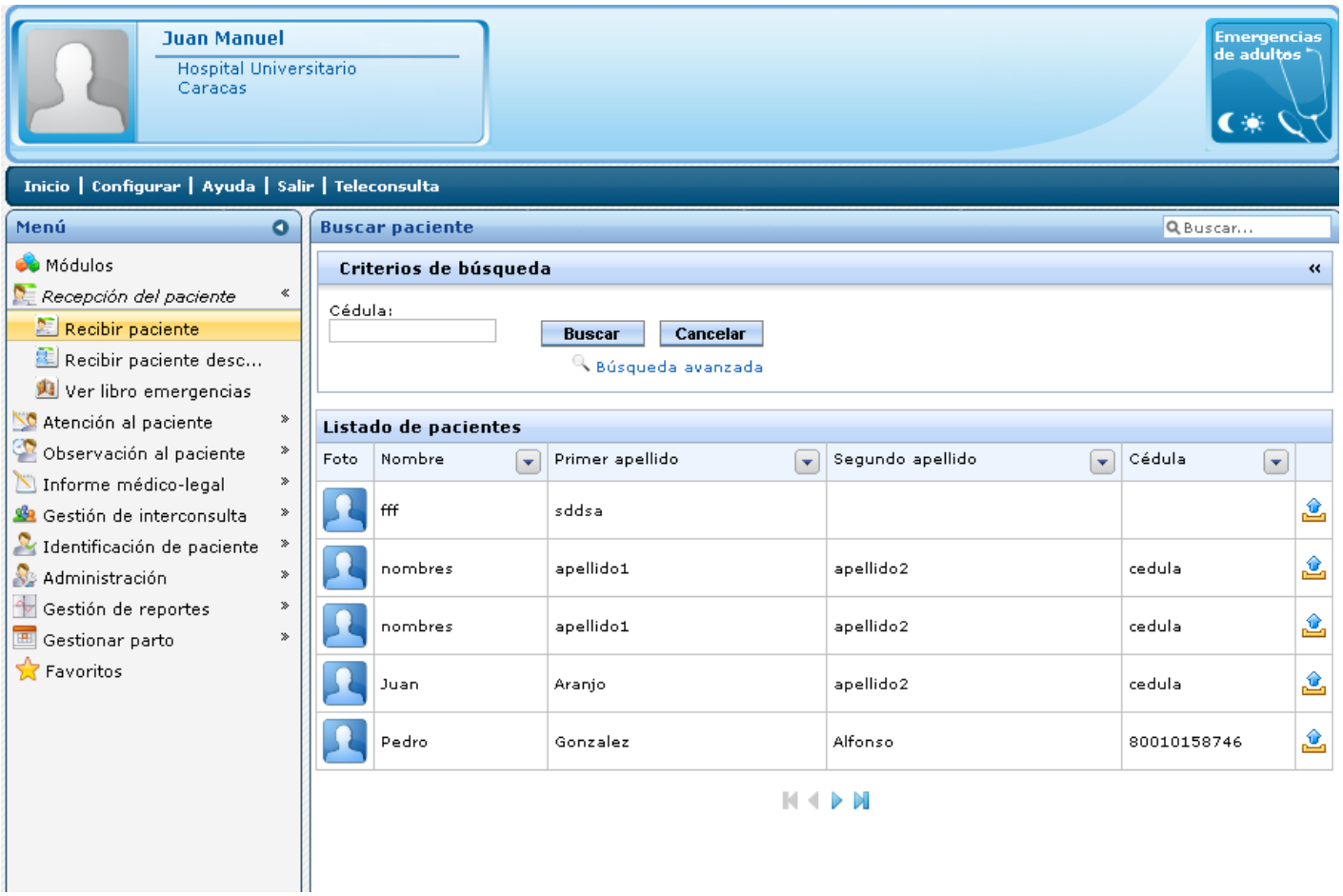
Multiplataforma: es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas.

NANDA: acrónimo de North American Nursing Diagnosis Association. Es una sociedad científica de enfermería cuyo objetivo es estandarizar el diagnóstico de enfermería.

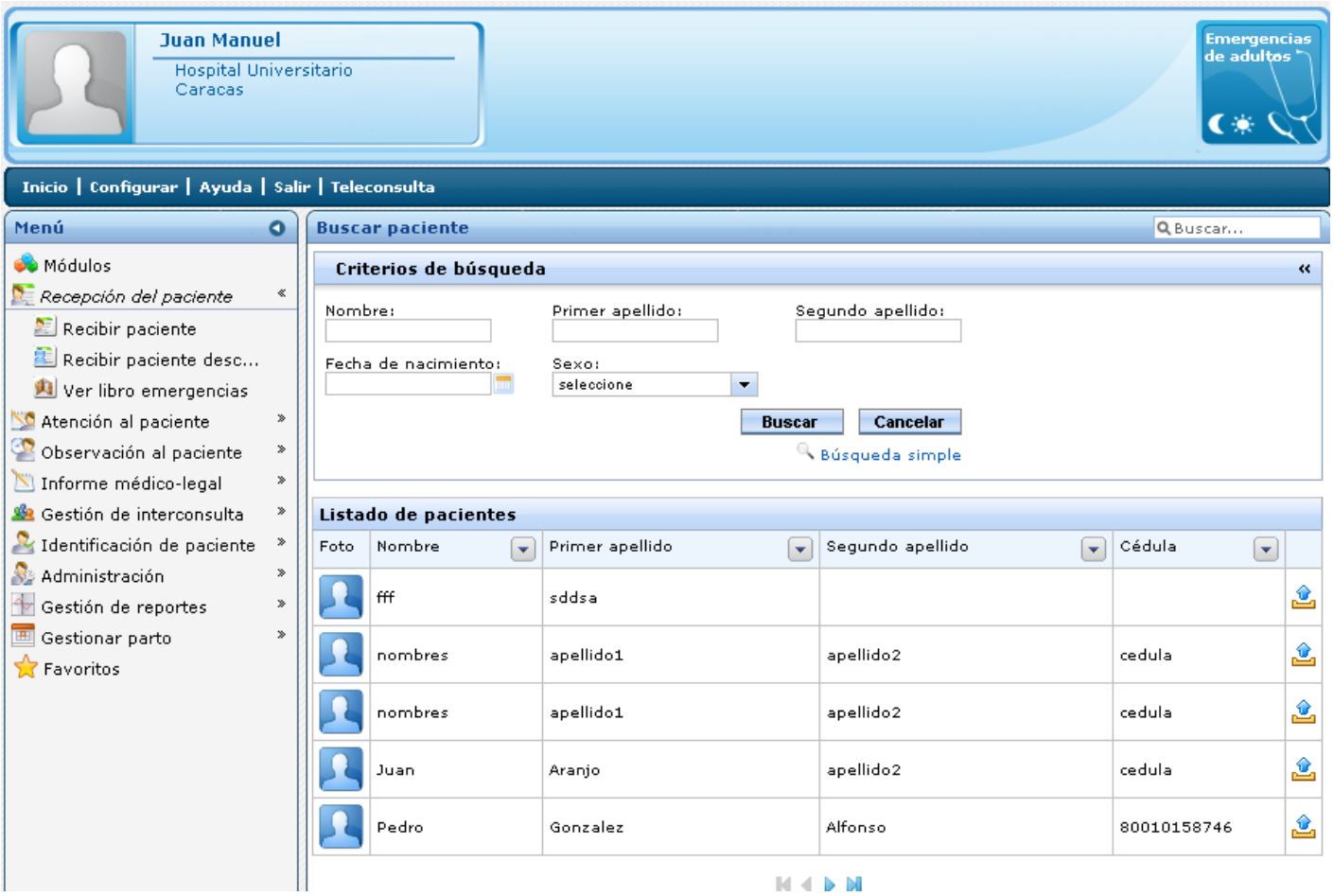
ANEXOS.

Interfaz de usuario

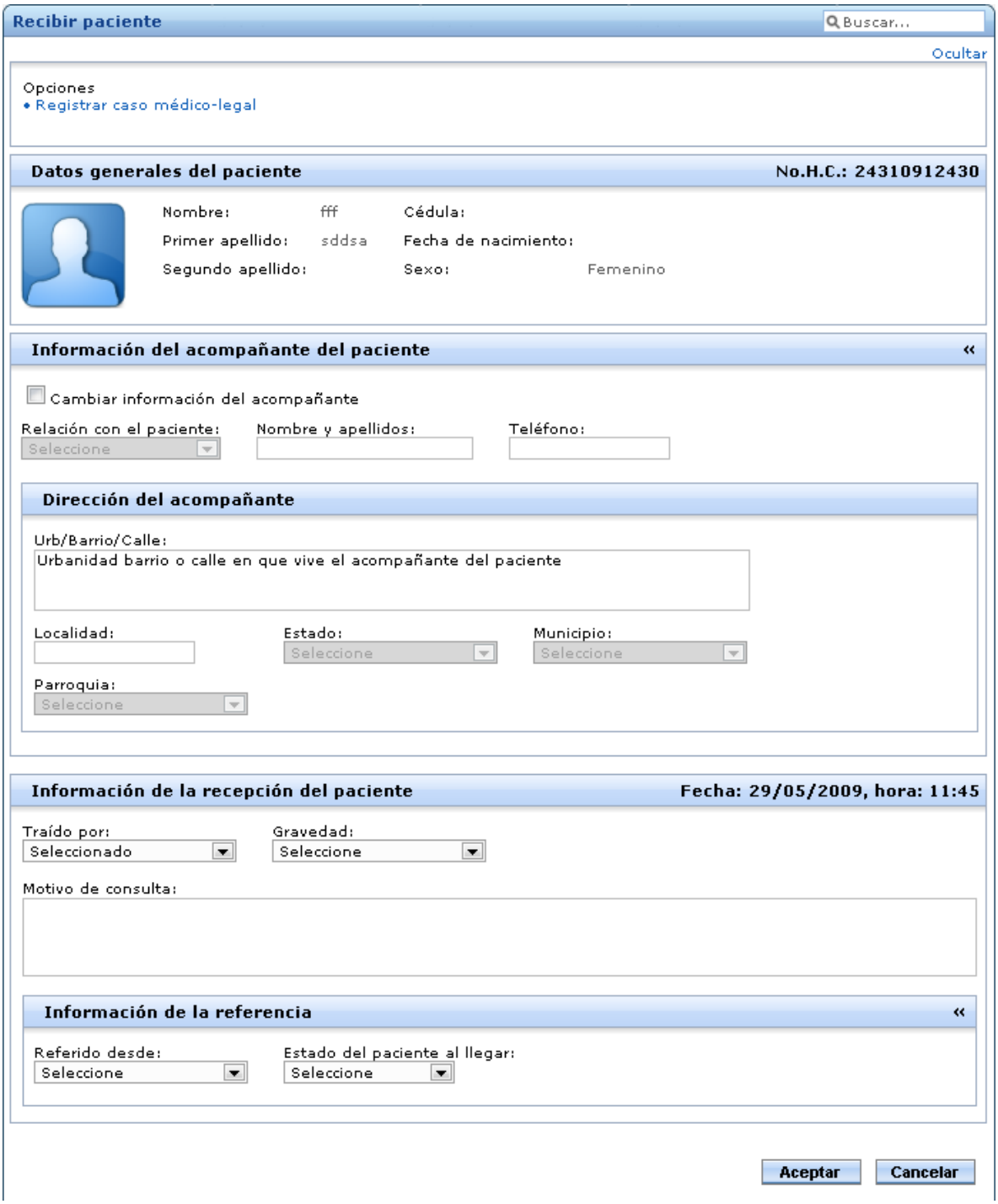
Anexo1 Página principal.



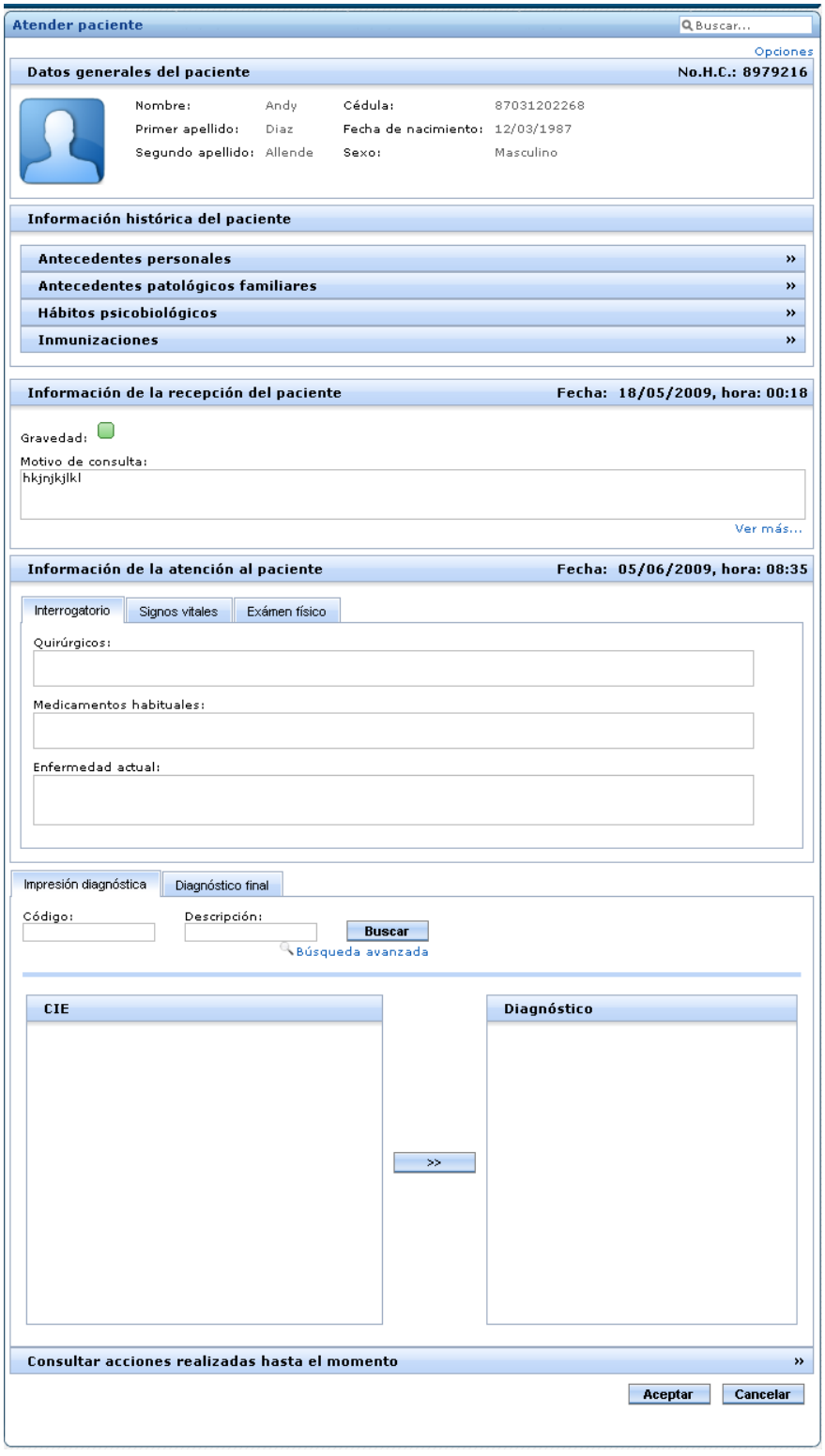
Anexo 2 Buscar paciente.



Anexo 3 Búsqueda avanzada de paciente.



Anexo 4 Recepción del paciente.



Anexo 5 Atender paciente.

Anexo 6 Atender paciente desconocido.