

Universidad de las Ciencias Informáticas

Facultad 2



**Trabajo de Diploma para optar por el Título de
Ingeniero en Ciencias Informáticas.**

**Título: Portal web del paciente para el Sistema de
Información Hospitalaria del Centro de Informática Médica.**

Autor: Juan Yoel Rodríguez Matos

Tutor: Yasser Manuel Garbey Bermudes

Co-tutor: Lianny Hernández De la Paz

La Habana, Junio 2014

“Año 56 de la Revolución”

Declaración de autoría

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los 16 días del mes de Junio del año 2014.

Juan Yoel Rodríguez Matos

Ing. Yasser Manuel Garbey Bermudes

Firma del autor

Firma del tutor

Ing. Lianny Hernández De la Paz

Firma del co-tutor

DATOS DE CONTACTO

Ing. Yasser Manuel Garbey Bermudes

Graduado en la Universidad de las Ciencias Informáticas en el año 2009. Posee la categoría docente de Instructor Recién Graduado. Ha impartido asignaturas de Introducción a la Programación, Programación 2, Ingeniería de Software 1 e Ingeniería de Software 2. Ha participado en varios proyectos de desarrollo vinculados al perfil de salud. Actualmente labora en el Departamento de Sistemas de Gestión Hospitalaria del Centro de Informática Médica (CESIM), desempeñándose como desarrollador.

Correo electrónico: ymgarbey@uci.cu

Ing. Lianny Hernández De la Paz

Graduada en la Universidad de las Ciencias Informáticas en el año 2013. Actualmente labora en el Departamento de Sistemas de Gestión Hospitalaria del Centro de Informática Médica (CESIM), desempeñándose como analista.

Correo electrónico: ldelapaz@uci.cu

AGRADECIMIENTOS

A mis padres y mis hermanas por ser un elemento vital en los momentos más difíciles.

A mi novia por haber estado ahí para mí en todo momento.

A mis tutores por haber sabido guiarme y apoyarme en todo el proceso.

A mis amigos por haberme soportado a lo largo de toda la carrera.

DEDICATORIA

A mis padres y mis hermanas.

RESUMEN

Uno de los problemas más significativos al que se enfrentan los centros asistenciales de salud a la hora de elevar la calidad de su servicio, es el referido a mantener una comunicación estable con las personas que acuden a ellos. Muchos de estos centros no brindan un marco o espacio mediante el cual interactuar con el paciente una vez que se encuentre fuera de la institución, lo que limita la eficacia y eficiencia de los servicios brindados por el hospital, materializándose en la sobrecarga de los flujos básicos de los procesos que se llevan a cabo en el mismo.

Es por ello que el objetivo de la presente investigación es desarrollar el Portal web del paciente para el Sistema de Información Hospitalaria del Centro de Informática Médica (CESIM); solución informática para gestionar la información de los diferentes procesos de un hospital que actualmente no cuenta con un mecanismo que permita la comunicación entre pacientes y el personal médico encargado de su salud. El desarrollo del portal web estuvo guiado por el Proceso Unificado de Desarrollo de Software (RUP), apoyándose en el Lenguaje de Modelado Unificado (UML). Se utilizó el JBoss Developer Studio como entorno de desarrollo, Java como lenguaje de programación, PostgreSQL como Sistema Gestor de Base de Datos y Visual Paradigm como herramienta de modelado.

Con el desarrollo de la solución planteada se espera mejorar el flujo de comunicación entre pacientes e instituciones médicas que hacen uso del Sistema de Información Hospitalaria del CESIM.

Palabras claves: Centro de Informática Médica, Historia Clínica, Portal web para el paciente, Sistema de Información Hospitalaria.

TABLA DE CONTENIDO

Introducción	11
CAPÍTULO 1 Fundamentación Teórica del Portal web para pacientes.....	14
1.1 Conceptos básicos relacionados con el dominio del problema	14
1.2 Portales web para pacientes.....	15
1.3 Tendencias y tecnologías	16
1.4 Tecnologías horizontales	22
1.5 Metodologías de desarrollo de software.....	22
1.6 Herramientas.....	23
CAPÍTULO 2 Características del Portal web para pacientes.....	26
2.1 Modelo de Dominio	26
2.2 Especificación de los requerimientos del software	28
2.3 Modelo de Casos de Uso del Sistema	31
2.4 Características del sistema.....	41
CAPÍTULO 3 Diseño del Portal web para pacientes.....	43
3.1 Descripción de la arquitectura, fundamentación	43
3.2 Modelo de Diseño	44
3.2.1 <i>Diagrama de Paquetes</i>	46
3.2.2 <i>Diagramas de Clases del Diseño</i>	46
3.3 Modelo de Datos	53
CAPÍTULO 4 Implementación del Portal web para pacientes.....	57
4.1 Estrategia de integración	57
4.2 Modelo de Implementación.....	58

4.3	Tratamiento de errores.....	61
4.4	Seguridad	61
4.5	Estrategias de codificación: estándares y estilos a utilizar	61
	Conclusiones.....	64
	Recomendaciones.....	65
	Referencias bibliográficas.....	66
	Bibliografía	70
	Anexos	75

ÍNDICE DE TABLAS

Tabla 2.1 Definición de los actores del sistema.....	32
Tabla 2.2 Descripción textual del caso de uso: Crear noticia.....	37
Tabla 2.3 Descripción textual del caso de uso: Crear mensaje.....	39
Tabla 2.4 Descripción textual del caso de uso: Consultar citas pendientes	41
Tabla 3.1 Descripción de la clase controladora: MensajeController	51
Tabla 3.2 Descripción de la clase controladora: NoticiaController	52
Tabla 3.3 Descripción de la clase controladora: HcController	52
Tabla 3.4 Descripción de los atributos comunes entre todas las entidades	54
Tabla 3.5 Descripción de la tabla: Noticia_portal.....	55
Tabla 3.6 Descripción de la tabla: Cuestionario_portal	55
Tabla 3.7 Descripción de la tabla: TipoCuestionario_portal	55

ÍNDICE DE FIGURAS

Figura 2.1 Diagrama de Modelo de Dominio	27
Figura 2.2 Vista global de actores del sistema	32
Figura 2.3 Diagrama de Casos de Uso del Sistema	33
Figura 3.1 Diagrama de paquetes	46
Figura 3.2 Diagrama de Clases del Diseño: Consultar citas pendientes	47
Figura 3.3 Diagrama de Clases del Diseño: Crear noticia	48
Figura 3.4 Diagrama de Clases del Diseño: Crear mensaje	49
Figura 3.5 Modelo de Datos	54
Figura 4.1 Diagrama de Componentes	59
Figura 4.2 Diagrama de Despliegue	60

INTRODUCCIÓN

Se vive el siglo XXI el cual se presenta con el paradigma de una nueva sociedad, en la que el conocimiento acumulado es la base fundamental del desarrollo de todas las naciones y de la humanidad en general. Donde el avance de las tecnologías le ha permitido al ser humano hacer provecho de la información y el conocimiento en formas o maneras sin precedentes en la historia del mismo. Lo cual ha resultado en la posibilidad existente hoy en día de propiciar un intercambio científico y cultural a escala mundial, que sobrepasa barreras geográficas, religiosas, políticas e inclusive el tiempo.

Es un hecho que este nuevo paradigma de sociedad en el que se vive, es caracterizado por un rápido movimiento y crecimiento exponencial de la información existente en todas las áreas del conocimiento humano, en un proceso que se retroalimenta a sí mismo; por lo que vale destacar el importante papel que juegan las nuevas tecnologías, principalmente el uso de la informática como herramienta fundamental y necesaria que faculta a dicha sociedad con métodos, procesos y técnicas para el manejo de dicha información. (1)

Una de las áreas en las que se ha tornado vital el uso de las nuevas tecnologías en los últimos años es la salud, debido a la necesidad imperante de elevar la calidad de atención al paciente y de mejorar el desempeño del personal médico-administrativo en los centros asistenciales.

Por lo que en la actualidad existe en el mundo la tendencia generalizada hacia un mayor uso de la informática médica; desarrollándose diversas soluciones con el objetivo principal de prestar servicios a los profesionales de la salud. Estas soluciones se han diversificado para dar al traste con los innumerables problemas que existen en el área de la salud. (2)

Uno de dichos problemas radica en la comunicación entre los pacientes y los centros asistenciales de salud, materializándose en la ejecución de los procesos que se llevan a cabo en estas instituciones.

Cuba, al igual que muchos otros países ha hecho énfasis en el proceso de informatización hospitalaria. En aras de lograr un avance positivo en este sector, surge la Universidad de las Ciencias Informáticas. Dentro de la misma existen varios centros, uno de ellos es el Centro de Informática Médica (CESIM). Dicho centro ha sido el encargado del desarrollo de varias soluciones, como el Sistema de Información Hospitalaria del CESIM, el cual actualmente no cuenta con un mecanismo que permita la debida comunicación entre el paciente y el personal de la salud dentro y fuera de la institución.

Dicho problema se ha extendido hasta el área administrativa, lo que afecta el buen funcionamiento de los procesos que se realizan, originándose situaciones desfavorables que repercuten en el personal que acude y labora en dichas instituciones, tales como el aumento del volumen de información almacenada en formato duro, crecimiento de llamadas telefónicas para casos no urgentes y de pacientes en espera, presencia de personas en un centro asistencial con el objetivo de adquirir los resultados de sus exámenes y estos no encontrarse disponibles, así como la imposibilidad de informar cambios de horario o reprogramación de consultas a tiempo.

El presente trabajo surge para darle solución a la situación antes expuesta por lo que en este sentido se plantea como **problema científico** ¿Cómo **viabilizar** el flujo de información entre los pacientes y las instituciones hospitalarias encargadas de su salud?

El **objeto de estudio** lo constituye el proceso de comunicación entre las instituciones hospitalarias y sus pacientes por medio de sistemas y herramientas informáticas. Enmarcado en el **campo de acción** de los mecanismos de comunicación que sirven de base al proceso de comunicación entre las instituciones hospitalarias y sus pacientes, por medio de portales web.

Para darle solución al problema, se define como **objetivo general**: Desarrollar un portal web del paciente para el Sistema de Información Hospitalaria del CESIM que viabilice la comunicación entre los pacientes y las instituciones hospitalarias.

Para darle solución al objetivo anteriormente planteado se definen las siguientes **tareas de la investigación**:

1. Identificar las tendencias actuales de las soluciones informáticas relacionadas con los portales web para pacientes.
2. Asimilar la arquitectura definida para el Sistema de Información Hospitalaria del CESIM.
3. Identificar los escenarios en los que se ajusta la aplicación para la homologación con los datos en el Sistema de Información Hospitalaria del CESIM.
4. Obtener mediante el Proceso Unificado de Desarrollo, la documentación perteneciente al Portal web del paciente del Sistema de Información Hospitalaria del CESIM.
5. Desarrollar el Portal web del paciente para el Sistema de Información Hospitalaria del CESIM.

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

Capítulo 1: Fundamentación Teórica: se presentan los principales conceptos que constituyen las bases para el desarrollo del Portal web del paciente para el Sistema de Información Hospitalaria del CESIM. Se realiza un estudio preliminar de algunos de los portales web para pacientes más utilizados en el mundo, vinculados a las instituciones de salud. Se fundamentan las tecnologías, metodologías y herramientas de desarrollo a utilizar.

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

Capítulo 3: Análisis y diseño del sistema: se realiza una descripción y análisis de la estructura de la solución que se propone para dar respuesta a la problemática planteada. Se fundamenta la arquitectura empleada y se detalla el Modelo de Datos, en el que se visualiza la estructura donde se almacena toda la información requerida en el sistema.

Capítulo 4: Implementación: se introduce el flujo de trabajo de implementación, partiendo de los resultados obtenidos en el diseño. Se describen las estrategias de integración con el Sistema de Información Hospitalaria del CESIM. Se exponen aspectos referentes a la seguridad del sistema, las estrategias de codificación, así como la forma en que se tratarán los errores.

CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA DEL PORTAL WEB PARA PACIENTES

En el presente capítulo se realiza la descripción de los portales web de pacientes de instituciones hospitalarias y de los principales conceptos asociados a los mismos. Se exponen algunos de los sistemas más utilizados y populares de este tipo, así como sus características. Se presentan las tecnologías, metodologías y software utilizado para el desarrollo de la solución del problema científico de la investigación.

1.1 Conceptos básicos relacionados con el dominio del problema

Es posible encontrar varias descripciones del concepto de portal web, pero básicamente es una puerta de entrada a internet, donde se gestionan contenidos, se ayuda al usuario, brindándole servicios y productos, de forma tal que pueda hacer cuanto necesite realizar en la red. El objetivo fundamental es lograr una gran dependencia y fidelización del mismo, brindándole una expectativa de servicios que normalmente usarían en internet, información que le resulte interesante y de esta manera establecer un vínculo entre el usuario y el portal.

Las funciones de los portales web pueden variar en dependencia del tipo de usuario al que estén destinados, pero generalmente presentan servicios de valor añadido tales como información de diversos tipos, chats, e-mail gratuito, mensaje a móviles, software de libre distribución, grupos de discusión, comercio electrónico, entre otros. (3)

En la actualidad, médicos de todas las latitudes reconocen el valor que tienen los portales web para pacientes en instituciones hospitalarias, indudablemente debido a la eficiencia que añaden a las actividades prácticas realizadas en los hospitales, y a la mejora que existe en cuanto al acceso y satisfacción por parte de los pacientes a la hora de interactuar con la institución hospitalaria encargada de su salud.

Prácticamente un portal web para pacientes se ve como un sitio web, pero a diferencia de este, que ofrece una experiencia estática a los usuarios, los portales para pacientes basados en la web constituyen una puerta de entrada en el mundo de las prácticas médicas para los pacientes. Un portal web de este tipo provee de una comunicación segura entre los pacientes y sus proveedores de atención médica; la posibilidad del portal de ofrecer servicios las 24 horas del día, permite a los pacientes manejar sus interacciones clínicas a conveniencia con sus proveedores de salud, permitiendo también que el personal de la institución responda cuando estime conveniente. (4) (5)

1.2 Portales web para pacientes

En la actualidad se ha extendido el uso de los portales web para pacientes, como medio para gestionar procesos que se realizan en las instituciones hospitalarias, aprovechando las mejoras que brindan los mismos en la eficiencia de la atención médica y en los circuitos administrativos. Aunque estos portales tienen un objetivo común, que no es más que servir de puente entre los pacientes y las instituciones en las cuales se atienden, algunos tienen características que los hacen más útiles y completos. (5)

A continuación se muestran varios de los más utilizados, vinculados al área de la salud:

CureMD es un Sistema de Información Hospitalaria ampliamente utilizado que presenta diversas características y funcionalidades. Es altamente integrado, interoperable y modificable. Entre los servicios que brinda destaca su portal web para el paciente, el mismo brinda la posibilidad a los pacientes de acceder de una forma sencilla a un resumen de sus registros actualizados, en cualquier momento y lugar, con acceso seguro a gráficas, resultado de exámenes de laboratorio, informes de radiología, mensajes y declaraciones del personal encargado de su salud.

Entre los diferentes beneficios que brinda este portal a los pacientes, podemos mencionar que los mismos pueden realizar una autenticación completa en la web, así como la posibilidad de hacer solicitudes de citas en línea. Cuenta con un servicio de mensajería que permite una continua interacción y comunicación con los pacientes. Posibilita renovar las prescripciones de los pacientes con facilidad; además de ofrecer un servicio de clínica a tiempo completo en el que los pacientes pueden acceder a los resultados de sus exámenes de laboratorio, reportes de radiología y a un resumen de su historia clínica. Este portal tiene implementado un sistema de alertas automáticas para citas, prescripciones y resultados de exámenes. (6)

Athenahealth es un portal web para pacientes que representa la piedra angular en el servicio de comunicación con el paciente del Sistema de Información Hospitalaria del mismo nombre. Tiene como objetivo, inspirar seguridad al comprometerse con el paciente a extender la comunicación con el personal encargado de su salud más allá de la consulta.

Entre los principales beneficios que brinda, se encuentra la posibilidad de realizar una autenticación completa en la web, cuenta con un servicio de mensajería que ofrece un intercambio seguro de información entre el paciente y el personal encargado de su salud. Ofrece acceso a un resumen de la historia clínica en línea, además de implementar un sistema de alertas automáticas para citas, prescripciones y resultados de

exámenes. Una de las características más llamativas del sistema, es que cuenta con una aplicación desktop y otra móvil para la interacción con el mismo. (7)

Orion es un portal web para pacientes seguro y privado, diseñado con el objetivo de que los pacientes administren su propia historia clínica y se comuniquen con sus proveedores de salud. Permite el intercambio de documentación, imágenes, resultado de exámenes y mensajes entre pacientes y médicos. Ofreciendo beneficios al paciente, como una completa autenticación en línea. Tiene como objetivo principal, que sea el propio paciente quien tome responsabilidad por su atención médica, proporcionando un fácil acceso a información de referencia relevante y actual. (8)

El estudio realizado sobre dichos sistemas, permitió conocer las características específicas usadas para su construcción, las cuales posibilitaron la identificación de las principales funcionalidades con las que debe contar un portal web para pacientes; sirviendo de guía para el desarrollo de la solución que está en correspondencia con el objetivo trazado en la presente investigación, siendo capaz de extender el flujo de comunicación entre el paciente y las instituciones encargadas de su salud. También se arribó a la conclusión de que las soluciones antes descritas no responden a los problemas derivados de la relación médico paciente dentro y fuera de las instituciones que hacen uso del Sistema de Información Hospitalaria del CESIM.

1.3 Tendencias y tecnologías

En el desarrollo de un proyecto es de vital importancia realizar un estudio profundo de las tecnologías y herramientas a utilizar para la creación del mismo, y de esta manera lograr un producto final con las características y calidad requerida. Debido a que la solución propuesta debe integrarse al Sistema de Información Hospitalaria del CESIM se asimiló la arquitectura definida por el departamento de gestión hospitalaria de dicho centro.

1.3.1 *Arquitectura cliente-servidor*

El Sistema de Información Hospitalaria del CESIM utiliza la arquitectura cliente-servidor. Esta arquitectura consiste básicamente en un cliente que realiza peticiones a otro programa (el servidor) que le da respuesta. La interacción cliente-servidor es el soporte de la mayor parte de la comunicación por redes. Permite a los usuarios finales obtener acceso a la información de forma transparente, de diferentes lugares y aún en entornos multiplataforma. Este tipo de arquitectura presenta varias ventajas debido a la existencia de

plataformas de hardware cada vez más baratas, lo que constituye una de sus principales fortalezas, la posibilidad de utilizar máquinas mucho más baratas que las requeridas para una solución centralizada. Facilita la integración entre sistemas diferentes, favorece el uso de interfaces gráficas interactivas, la estructura inherentemente modular facilita además la integración de nuevas tecnologías y el crecimiento de la infraestructura computacional, favoreciendo así la escalabilidad de las soluciones. (9)

1.3.2 *Arquitectura en capas*

La arquitectura en capas es un estilo de diseño cuyo objetivo principal es la separación y agrupamiento de los componentes del software atendiendo a la función que cumplen en el mismo. Toda aplicación contiene código de presentación, código de procesamiento de datos y código de almacenamiento de datos, por lo que generalmente esta división se realiza en tres capas diferentes, la de presentación o interfaz de usuario, la capa de negocio y la capa de acceso a datos. (10)

1.3.3 *Patrones de Arquitectura y Diseño*

Los patrones de diseño expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software. (11) Los patrones de arquitectura expresan un esquema organizativo estructural fundamental para sistemas de software. (12) Para la realización del Portal web para el paciente del Sistema de Información Hospitalaria del CESIM, se utilizó el patrón de diseño de arquitectura Modelo-Vista-Controlador.

1.3.3.1 *Modelo-Vista-Controlador (MVC)*

El patrón de arquitectura, conocido por sus siglas en inglés Model View Controller, que significa Modelo Vista Controlador, permite realizar la programación multicapa, separando en tres componentes distintos los datos de una aplicación, la interfaz de usuario y la lógica de control. Este patrón está diseñado para reducir el esfuerzo de programación necesario en la implementación de sistemas múltiples y sincronizados de los mismos datos. En aplicaciones web, donde la vista es el HTML y el código que provee de datos dinámicos a la página, el modelo es el sistema de gestión de base de datos y el controlador representa la lógica del negocio. Además presenta la ventaja de que la conexión entre el Modelo y sus Vistas es dinámica, o sea se produce en tiempo de ejecución, no en tiempo de compilación. (13)

Modelo: representa la información con la que trabaja la aplicación, o sea, su lógica de negocio. El modelo no tiene conocimiento específico de los Controladores o las Vistas, ni siquiera contiene referencias a ellos.

Es el propio sistema el que tiene la responsabilidad de mantener las relaciones necesarias entre el Modelo y sus Vistas, y notificar a las vistas cuando cambia el modelo. (13)

Vista: convierte el modelo en una página web, que facilita al usuario interactuar con ella. Interactúa preferentemente con el Controlador, pero es posible que trate directamente con el Modelo a través de una referencia al propio modelo. (13)

Controlador: es el encargado de procesar las interacciones del usuario y ejecuta los cambios adecuados en el modelo o la vista. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el modelo a través de una referencia al propio modelo. (13)

1.3.4 Tecnologías utilizadas en el proceso de desarrollo

Debido a la necesidad de integrar un portal web para pacientes al Sistema de Información Hospitalaria del CESIM se propone seguir la línea de desarrollo del Departamento de Sistemas de Gestión Hospitalaria y utilizar tecnologías y herramientas que permitan su uso sin necesidad de pago de licencias. A continuación se presentan dichas herramientas y las tecnologías asumidas, según su ubicación dentro de la arquitectura en capas.

1.3.4.1 Java

Java es un lenguaje de programación multiplataforma y orientado a objetos. Su diseño, robustez, el respaldo de la industria y su fácil portabilidad han hecho de Java uno de los lenguajes con un mayor crecimiento y amplitud de uso en distintos ámbitos de la industria de la informática. Es altamente seguro e indiferente de la arquitectura, pues está diseñado para soportar aplicaciones que serán utilizadas en los más variados entornos de red. Es libre de patentes de software, por lo que cumple con uno de los principales objetivos planteados. (14)

1.3.4.2 Capa de presentación

La capa de presentación tiene como principal objetivo encargarse del formato en el que se va a mostrar la información, es decir lo que se le muestra al usuario, permite capturar las acciones realizadas por el mismo y solo se comunica con la capa de negocio. (15)

Java Server Faces (JSF)

Es el marco de trabajo para crear aplicaciones java J2EE basadas en el patrón MVC, permite desarrollar rápidamente aplicaciones de negocio dinámicas en la que toda la lógica del negocio se implementa en Java o es llamada desde Java, creando paginas para las vistas muy sencillas. Pretende normalizar y estandarizar el desarrollo de aplicaciones web. Los componentes JSF facilitan la construcción de interfaces web del lado del servidor, se conectan a fuentes de datos y relacionan de forma transparente eventos del cliente con manejadores en el servidor. (16)

RichFaces

RichFaces es una librería de código abierto basada en Java que permite crear aplicaciones web con Ajax. Incluye ciclo de vida, validaciones, conversiones y la gestión de recursos estáticos y dinámicos. Construye sobre el framework de Java Server Faces (JSF), sobre él implementa unos filtros para permitir peticiones Ajax en la página. (17)

Ajax4jsf

Ajax4jsf es una librería de código abierto, totalmente integrable a la arquitectura de JSF, extendiendo la funcionalidad de sus etiquetas al dotarlas con tecnologías Ajax sin necesidad de agregarle código JavaScript, este framework brinda la posibilidad de recargar determinados componentes de la página sin necesidad de refrescarla por completo, además que permite realizar peticiones automáticas al servidor y controlar cualquier evento que realice el usuario. (18)

Facelets

JavaServer Facelets es un framework para plantillas centrado en la tecnología JSF (JavaServer Faces), lo cual brinda la posibilidad a que JSP (JavaServer Pages) y JSF (JavaServer Faces) puedan funcionar conjuntamente en una misma aplicación web, debido a que JSP procesa los elementos de la página de arriba abajo, mientras que JSF realiza su propio re-rendering, siendo Facelets quien rellena el vacío entre estas dos tecnologías. Es de código abierto, distribuido bajo la licencia Apache. (19)

XHTML

XHTML (eXtensible Hypertext Markup Language) por sus siglas en inglés, es una versión más estricta y limpia del Lenguaje de Marcado de Hipertexto, que se crea precisamente con el objetivo de remplazarlo ante su limitación de uso con las cada vez más abundantes herramientas, basadas en el Lenguaje de Marcado Extensible (XML), un formato de datos universal altamente extensible, que define una manera estándar de estructurar el marcado de documentos. Es un lenguaje cuyo etiquetado, más estricto que HTML, permitirá una correcta interpretación de la información independiente del dispositivo desde el que se accede a ella. (20)

Seam UI

Seam UI son una serie de controles altamente integrables con JBoss Seam. Están dirigidos a complementar los controles JSF incorporados y los controles de otras bibliotecas externas. Las mejoras que este añade a JSF van desde validación, hasta la integración de la navegación en la interfaz de usuario basada en flujo de páginas o procesos de negocio. (21)

1.3.4.3 Capa de negocio

La capa de negocio es donde residen las funciones que se ejecutan, se reciben las peticiones del usuario, se procesa la información y se envían las respuestas tras el proceso. Se le llama capa de negocio o de lógica de negocio, porque es aquí donde se establecen todas las reglas que deben de cumplirse. Esta capa se comunica con la de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de acceso a datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él. (15)

Seam

JBoss Seam es un framework para el desarrollo de aplicaciones web en Java, que define un modelo de componentes uniforme para toda la lógica de negocio de las aplicaciones que sean desarrolladas mediante su utilización. Integra fácilmente tecnologías estándares como Java Server Faces (JSF), modelo de componentes para la capa de presentación; Enterprise JavaBeans (EJB3), modelo de componentes para la lógica de negocio y persistencia del lado del servidor; Java Persistence API (JPA), y de Business Process Management (BPM), integra además librerías de código abierto basadas en JSF como RichFaces e ICEFaces. (22)

1.3.4.4 Capa de acceso a datos

La capa de acceso a datos es la capa encargada de almacenar los datos del sistema y de los usuarios. Su función es almacenar y devolver datos a la capa de negocio. En una arquitectura de tres capas, esta capa es la única que puede acceder a los mismos. Está formada por uno o varios sistemas gestores de base de datos, localizado en un mismo servidor o en varios. (15)

Hibernate

Hibernate es una herramienta de mapeo objeto-relacional para la plataforma Java que facilita el mapeo de atributos entre una base de datos relacional y el modelo de objetos de una aplicación. Es una tecnología de software libre distribuido bajo los términos de licencia GNU LGPL. Hibernate tiene como objetivo solucionar el problema del problema 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 la base de datos (modelo relacional). Esta herramienta ofrece un lenguaje de consulta de datos llamada Hibernate Query Language (HQL), al tiempo que presenta una Interfaz de Programación de Aplicaciones (API), para construir las consultas. Logra además mantener la portabilidad entre todos los motores de bases de datos, con un ligero incremento en el tiempo de ejecución. (23)

Enterprise Java Beans (EJB3)

La tecnología Enterprise JavaBeans (EJB), es un componente de arquitectura del lado del servidor para la plataforma Java Enterprise Edition (Java EE). El objetivo de los EJB es dotar al programador de un modelo que le permita abstraerse de los problemas generales de una aplicación empresarial (conurrencia, transacciones, persistencia, seguridad, etc.), para centrarse en la lógica del negocio. La lógica del negocio reside en los Enterprise Beans y no en el lado del cliente, permitiendo que el desarrollo en el lado del cliente este desacoplado de la lógica del negocio. El hecho de estar basado en componentes permite que estos sean flexibles y sobre todo reutilizables. (24)

Java Persistence API (JPA)

Java Persistence API, conocida por sus siglas JPA, es la API de persistencia desarrollada para la plataforma Java EE y está incluida en el estándar EJB3. Tiene como objetivo no perder las ventajas de la orientación

a objetos al interactuar con una base de datos y permitir usar objetos regulares llamados POJOs (Plain Old Java Object). (25)

1.4 Tecnologías horizontales

Existen tecnologías que se extienden por todas las capas mencionadas y sirven de soporte a las tecnologías que se utilizan en cada una de ellas, las cuales se describen a continuación.

1.4.1 *Java Enterprise Edition 5 (JEE 5)*

Java Enterprise Edition es una plataforma de programación, para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java, con arquitectura de N capas distribuidas y que se apoya ampliamente en componentes de software modulares ejecutándose sobre un servidor de aplicaciones. JEE está enfocado en la creación de aplicaciones empresariales, las cuales se caracterizan, por resolver problemas corporativos, lo que supone, almacenamiento seguro, concurrencia, seguridad, escalabilidad, y procesamiento distribuido. (16)

1.4.2 *JBoss Application Server*

JBoss es un servidor de aplicaciones de código abierto, implementado en Java. Puede utilizarse en cualquier sistema operativo para el que esté disponible Java. Tiene alta calidad y eficiencia en sus funcionalidades. JBoss es ideal para aplicaciones Java y aplicaciones web. Soporta EJB 3.0, lo que permite que el desarrollo de aplicaciones sea mucho más simple. (26)

1.4.3 *Java Virtual Machine*

Una máquina virtual de java, es un programa nativo, es decir, ejecutable en una plataforma específica, capaz de interpretar y ejecutar instrucciones expresadas en un código binario especial (el Java Bytecode), el cual es generado por el compilador del lenguaje Java. Es la clave de muchas de las características principales de Java, como la portabilidad, la eficiencia y la seguridad (27)

1.5 Metodologías de desarrollo de software

Las metodologías para el desarrollo de software imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente. Una metodología de desarrollo de software tiene como principal objetivo aumentar la calidad de software que se produce en todas y cada una de sus fases

de desarrollo. No existe una metodología universal que se pueda aplicar a cualquier tipo de proyecto, por lo que su selección depende de las características de cada uno de ellos.

Para el desarrollo del Portal web para pacientes del Sistema de Información Hospitalaria del CESIM, se propone como metodología Proceso Unificado de Desarrollo (RUP), sugiriendo el uso del Lenguaje Unificado de Modelado (UML por sus siglas en inglés), debido a que por sus características permite englobar todo el proceso de desarrollo, generándose además los artefactos y documentos necesarios a lo largo del ciclo de vida del software.

1.5.1 Proceso Unificado de Desarrollo

RUP es una forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo). No es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización. Realiza el análisis y diseño, tan completo como sea posible. Se caracteriza por ser guiados por casos de uso, centrado en la arquitectura además de ser iterativo e incremental. (28)

1.5.2 Lenguaje Unificado de Modelado

UML por sus siglas en inglés, Unified Modeling Language: es el lenguaje de modelado de software más conocido y utilizado en la actualidad. UML ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales, tales como procesos de negocios y funciones del sistema, y aspectos concretos como lenguajes de programación, esquema de bases de datos y componentes de software reutilizable. Este lenguaje de modelado permite tener un mayor rigor en la especificación, realizar una verificación y validación del modelo de desarrollo, automatizar varios procesos, así como generar código a partir de los modelos y viceversa. (29)

1.6 Herramientas

1.6.1 JBoss Developer Studio

El JBoss Developer Studio proporciona un rendimiento superior para todo el ciclo de vida de desarrollo. Incluye un amplio conjunto de capacidades de las herramientas y soporte para múltiples modelos y marcos de programación, incluyendo Java Enterprise Edition 6, RichFaces, JavaServer Faces (JSF), Enterprise JabaBeans (EJB), Java Persistence API (JPA), Hibernate, HTML5 y muchas otras tecnologías populares. (30)

JBoss Tools

JBoss Tools son un conjunto de plug-ins que tienen como objetivo ayudar a los desarrolladores a crear aplicaciones web de forma rápida y sencilla. (31)

Está compuesto por los siguientes módulos:

- **RichFaces VE:** editor visual que incluye soporte para las librerías de componentes JSF incluyendo JBoss RichFaces. Brinda el apoyo para la edición visual de páginas HTML, JSF, JSP (Java Server Pages) y Facelets.
- **Seam Tools:** incluye soporte para la vinculación de los componentes del framework Seam.
- **Hibernate Tools:** soporta el mapeo de archivos, anotaciones y JPA con ingeniería inversa, completamiento de código, asistentes de proyecto, refactorización, ejecución interactiva de HQL/JPA-QL (JPA Query Language)/Criteria.
- **JBoss AS Tools:** contiene funciones para el despliegue eficiente de cualquier tipo de proyecto en el IDE.

1.6.2 Sistemas de Gestión de Base de Datos.

Los sistemas de gestión de base de datos, 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 el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante. (32)

1.6.2.1 PostgreSQL

PostgreSQL es un Sistema de Gestión de Base de Datos Objeto-Relacional (Object-Relational Database Management System (ORDBMS)), libre, por lo que no tiene costo asociado por lo que cualquiera puede disponer de su código fuente, modificarlo a voluntad y redistribuirlo libremente. Puede ser ejecutado sobre la mayoría de los sistemas operativos de existen en la actualidad. En cuanto a funciones, poseen bloques de código que se ejecutan en el servidor, los cuales pueden ser escritos en varios lenguajes, con la potencia que brinda cada uno de ellos.

Posee características sofisticadas como fiabilidad, replicación asíncrona, transacciones anidadas, realización de respaldo de datos en línea, optimizador o planificador de consultas y soporta internacionalización. Es altamente escalable en cuanto a la cantidad de información que puede manejar y al número de usuarios concurrentes que puede alojar. (33)

Por las características antes mencionadas, se propone la utilización de PostgreSQL en su versión 9.2 como sistema de gestión de base de datos en el desarrollo del Portal web para el paciente.

1.6.3 Visual Paradigm

Visual Paradigm es una herramienta CASE (Ingeniería de Software Asistida por Computación): propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y diseño, hasta la generación de código fuente de los programas y la documentación. Tienen como objetivo fundamental, soportar el ciclo de vida completo del proceso de desarrollo de software, a través de la representación de todo tipo de diagramas. (34)

Para el desarrollo del Portal web para el paciente se utiliza Visual Paradigm en su versión 8.0 por sus características y las ventajas que ofrece.

Conclusiones

Como resultado del análisis realizado en el presente capítulo, se determinó que los Sistemas de Información Hospitalaria existentes, que embeben portales web para pacientes, no manejan la información necesaria para la red de instituciones médicas que trabajan con el Sistema de Información Hospitalaria del CESIM, aunque cabe destacar que sirvieron como patrón en el diseño de la solución desarrollada. Además, se asimilaron las tecnologías y herramientas definidas para el Sistema de Información Hospitalaria del CESIM debido a que esto permite una total integración con dicho sistema.

CAPÍTULO 2 CARACTERÍSTICAS DEL PORTAL WEB PARA PACIENTES

En el presente capítulo, con el objetivo de una mejor comprensión de los procesos del negocio y del contexto en el cual se desarrolla el portal web, se muestra el Modelo de Dominio, donde quedan mostrados los conceptos relacionados con el portal web y las relaciones existentes entre los mismos. Se especifican los requerimientos funcionales y no funcionales y se agrupan los primeros en casos de uso, con el objetivo de estructurar el Diagrama de Casos de Uso del Sistema.

2.1 Modelo de Dominio

El Modelo de Dominio o Modelo Conceptual es una representación visual de los conceptos u objetos que se manejan en el dominio del sistema. Se utiliza para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema. Es un medio para comprender el sector de negocio al cual el sistema va a servir. Este modelo puede ser usado como punto de partida para el diseño del sistema. Los objetos o conceptos incluidos en el Modelo de Dominio no describen clases u objetos del software; sino entidades o conceptos del mundo real que están asociados al problema en cuestión. (35)

2.1.1 Conceptos fundamentales del dominio

A continuación se describen los principales conceptos asociados al ámbito del negocio donde se desarrolla el problema; con el objetivo de que se tenga una mejor comprensión del Diagrama de Modelo de Dominio. Dichos conceptos están profundamente relacionados con los conceptos existentes en los portales web para pacientes.

Usuario: persona que por medio de un ordenador o dispositivo móvil conectado a internet, puede acceder al portal web para el paciente, de acuerdo con los privilegios, permisos y roles asignados.

Administrador: usuario del portal web, encargado de realizar la configuración del mismo.

Paciente: sujeto que recibe los servicios de un médico u otro profesional de la salud y se somete a exámenes, a un tratamiento o una intervención.

Médico: profesional altamente cualificado en atención médica, capaz de dar respuestas generalmente acertadas y rápidas a problemas de salud.

Mensaje: contenido a transmitir entre un emisor (Portal web para el paciente) y un receptor (paciente, médico, administrador) en una situación comunicacional.

Noticia: texto informativo que se refiere a un hecho novedoso o no muy común, ocurrido en determinado ámbito específico.

Cita: encuentro previamente acordado entre dos o más personas en una fecha, hora y lugar determinado.

Historia Clínica: conjunto de documentos que contienen los datos, valoraciones e información de cualquier índole, sobre la situación y la evolución clínica de un paciente a lo largo del proceso asistencial.

Examen de laboratorio: exploración complementaria solicitada por un médico para confirmar o descartar un diagnóstico.

2.1.2 Diagrama del Modelo de Dominio

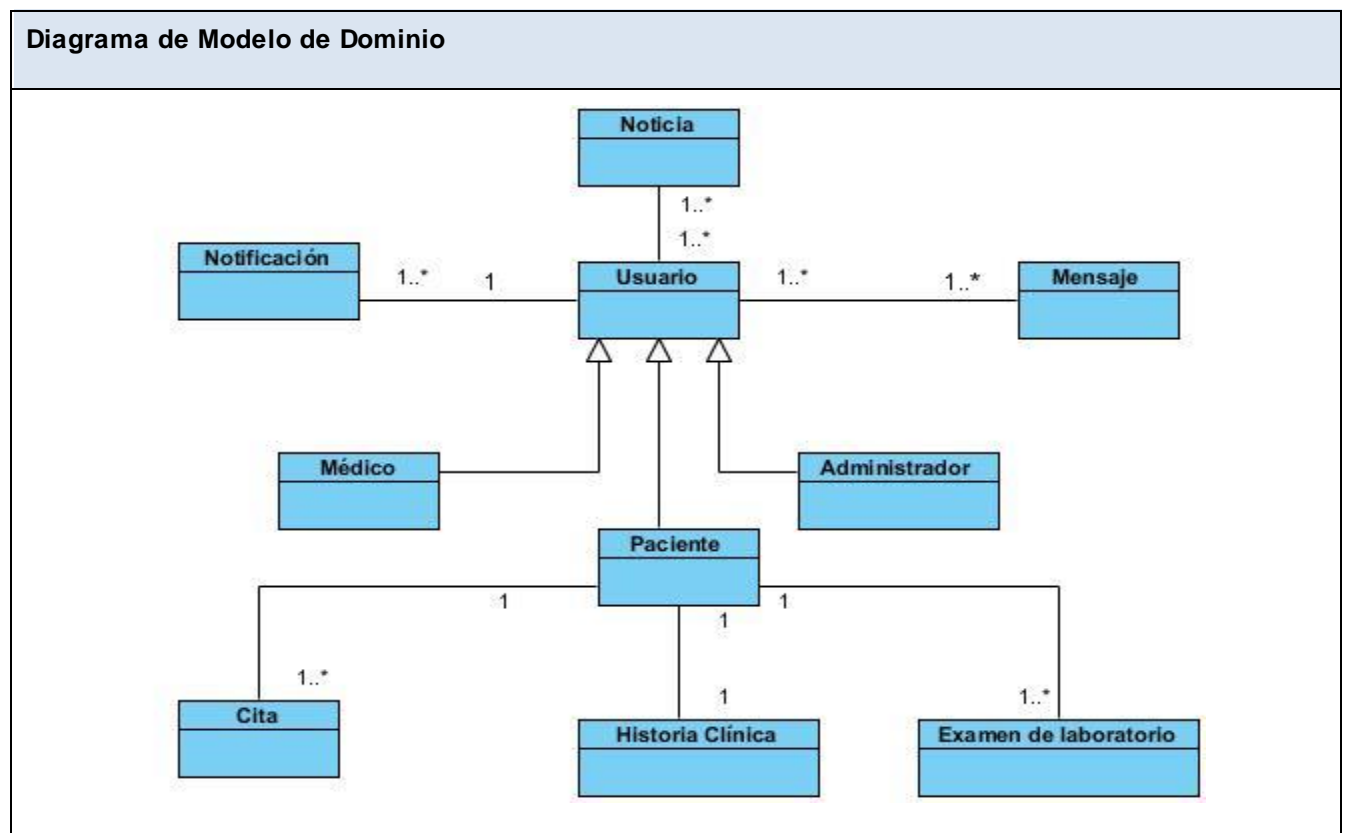


Figura 2.1 Diagrama de Modelo de Dominio

Como se evidencia en el Diagrama de Modelo de Dominio, un usuario del sistema puede ser un médico, paciente o un administrador, el usuario tiene la posibilidad de recibir noticias o recibir y enviar uno o varios

mensajes que son supervisados por el administrador. En caso de que el usuario sea un paciente, el mismo puede consultar los resultados de sus exámenes de laboratorio, consultar un resumen de su historia clínica o simplemente realizar una consulta de sus citas pendientes.

2.2 Especificación de los requerimientos del software

Requisitos funcionales del sistema

Los requerimientos funcionales son capacidades o condiciones que necesita el usuario para resolver un problema o lograr un objetivo. Son una definición detallada y formal de una función que debe realizar el sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. (36)

El portal web deberá permitir:

- RF1. Crear noticia.
- RF2. Ver detalles de noticia.
- RF3. Buscar noticia.
- RF4. Modificar noticia.
- RF5. Ver datos de noticia.
- RF6. Eliminar noticia.
- RF7. Listar mensajes.
- RF8. Crear mensaje.
- RF9. Ver detalles de mensaje.
- RF10. Modificar mensaje.
- RF11. Eliminar mensaje.
- RF12. Supervisar mensaje.
- RF13. Ver datos de mensaje.
- RF14. Buscar mensaje.
- RF15. Consultar Historia Clínica.

RF16. Consultar resultados exámenes de laboratorio.

RF17. Consultar citas pendientes.

RF18. Configurar foro.

Requisitos no funcionales del sistema

Los requerimientos no funcionales son requisitos que imponen restricciones en el diseño o implementación. Son propiedades o cualidades que el producto debe tener. (36)

Estos requerimientos se agrupan en varias categorías:

Requisitos de Usabilidad

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

Requisitos de Fiabilidad

El portal web para pacientes debe estar disponible de forma permanente y funcionar sin necesidad de intervención del administrador.

Requisitos de Seguridad

El portal web para pacientes debe mantener seguridad y control a nivel de usuarios, garantizando el acceso de los mismos sólo a los niveles establecidos de acuerdo a la función o rol que desempeñan. Las contraseñas podrán cambiarse únicamente por el propio usuario o por el administrador del portal web. Toda entrada de información estará validada, de forma que la introducción de datos incorrectos se mostrará al usuario especificándole el tipo de error. Todo tipo de mensaje, noticia o comunicación establecida a través del portal debe ser supervisado por el administrador del portal web.

Requisitos de Rendimiento

El portal web debe permitir la conexión de múltiples usuarios, garantizando un aprovechamiento óptimo de los recursos. Cada uno de las acciones que se ejecuten serán lo suficientemente óptimas en cuanto al uso de memoria física y necesidad de procesamiento.

Requisitos de Hardware

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

Estaciones de trabajo

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

Servidores

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

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

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

Requisitos de Software

El componente estará integrado al sistema alas HIS, dicha aplicación debe poder ser desplegada en sistemas operativos Windows y Linux, utilizando la plataforma Java (Máquina Virtual de Java - Java Enterprise Edition), el servidor de aplicaciones JBoss AS y PostgreSQL como sistema para la gestión de la

base de datos. Los usuarios deberán disponer de un navegador web, este puede ser Firefox 3.6, Google Chrome 14 o versiones superiores de ellos.

Requisitos de Interfaz de usuario

Las interfaces creadas como parte del portal web contendrán los datos legibles y bien estructurados, además de permitir la interpretación correcta de la información. Debe de contar con una interfaz sencilla y atractiva para el usuario. La entrada de datos incorrecta será detectada claramente y se mostrará al usuario.

Requisitos de portabilidad

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

2.3 Modelo de Casos de Uso del Sistema

2.3.1 Definición de los actores del sistema

Los actores de un sistema son agentes externos, roles que las personas (usuarios) o dispositivos juegan cuando interactúan con el software. (36)

Actor	Descripción
Usuario	Usuario global que se autentica en el portal, el sistema lo valida y le asigna un rol con la posibilidad de realizar funciones propias del rol que le fue asignado.
Administrador	Utiliza el portal web para pacientes, con el objetivo de gestionar noticias y supervisar los mensajes que se manejan en el foro del paciente.
Paciente	Utiliza el portal web para pacientes, con el fin de consultar información personal sobre su salud, como resultado de exámenes de laboratorios, citas pendientes e historia clínica.

Médico	Utiliza el portal web para pacientes, con el objetivo de responder y realizar preguntas en el foro del paciente.
--------	--

Tabla 2.1 Definición de los actores del sistema

2.3.2 Vista global de actores del sistema

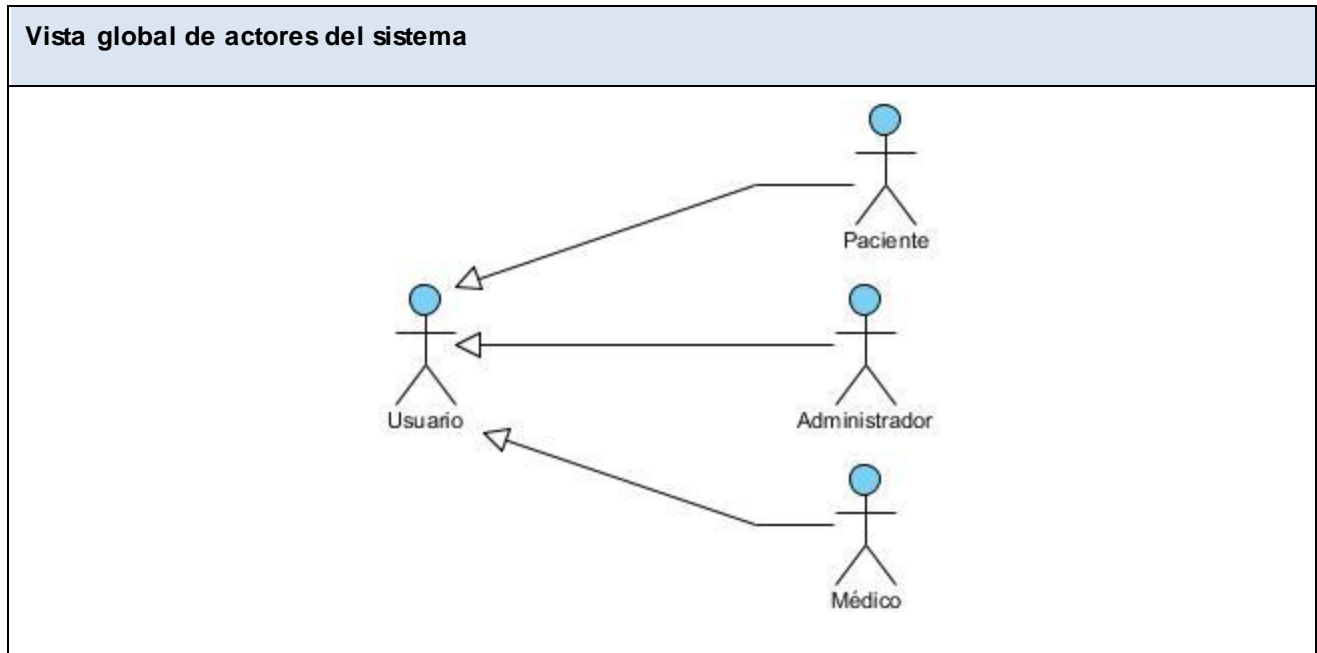


Figura 2.2 Vista global de actores del sistema

2.3.3 Diagrama de Casos de Uso del Sistema

Los Diagramas de Casos de Uso constituyen una representación gráfica de un conjunto de elementos como actores o usuarios que interactúan con el sistema y las actividades que realizan con el mismo llamadas casos de uso, así como las relaciones y dependencias que se establecen entre ellos. Dichos diagramas permiten que los desarrolladores y los clientes lleguen a un entendimiento sobre las condiciones y el alcance del sistema antes de emprender la fase de construcción del mismo. (36)

Diagrama de Casos de Uso del Sistema

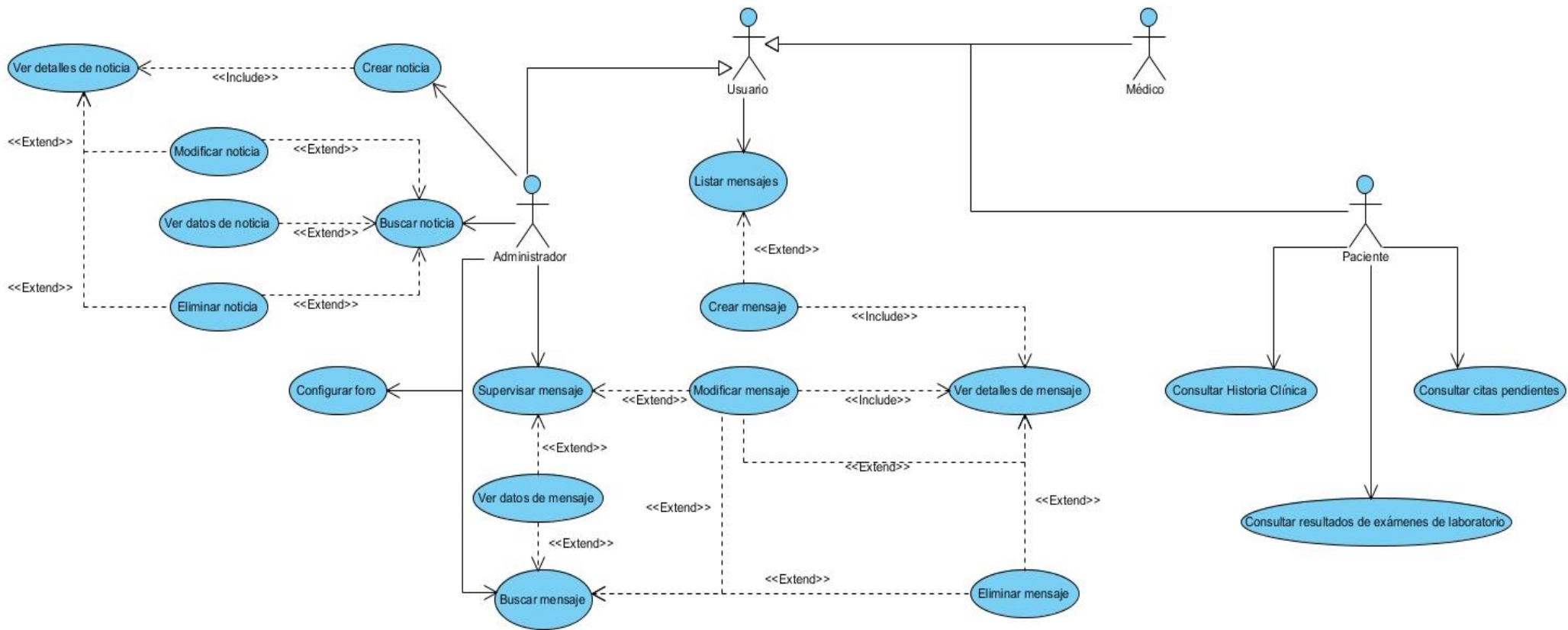


Figura 2.3 Diagrama de Casos de Uso del Sistema

El portal web para el paciente, cuenta con tres actores fundamentales que interactúan entre sí y con el propio portal en dependencia del rol que se le haya asignado.

Un usuario general tiene la posibilidad de interactuar en el foro del paciente mediante el intercambio de información con otros usuarios. También puede consultar noticias afines.

El administrador es encargado de gestionar las noticias, lo que le permite crear noticias; esta funcionalidad cuenta con las variantes de ver los detalles de las noticias, modificar, ver los datos, eliminar y buscar noticias. También es encargado de supervisar los mensajes que son manejados en el foro del paciente, lo que le permite modificar, ver detalles y eliminar un mensaje determinado.

El paciente tiene la posibilidad agregada de consultar un resumen de su historia clínica, los resultados de sus exámenes de laboratorio así como sus citas pendientes en cualquier momento y lugar a través del portal.

El médico tiene la posibilidad de establecer una comunicación directa a través del portal con el paciente, mediante mensajes que pueden ser preguntas o respuestas y que se visualizan en el foro del paciente.

Descripción textual de los casos de uso

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

Caso de uso	
CU-1	Crear noticia
Propósito	Permitir al actor la creación de una nueva noticia en el portal web.
Actores	Administrador
Resumen	El caso de uso inicia cuando el actor accede a la opción Crear noticia, el portal web brinda la posibilidad de introducir los datos para crear la Noticia, el sistema crea la Noticia, el caso de uso termina.
Precondiciones	

Flujo Normal de los Eventos	
Acción del actor	Respuesta del sistema
1. El caso de uso inicia cuando el actor accede a la opción Crear noticia.	
	<p>2. Muestra los campos predeterminados de Noticia:</p> <ul style="list-style-type: none">• Fecha de emisión• Fecha de caducidad• Título• Texto• Hora de emisión• Hora de caducidad <p>Brinda la posibilidad de introducir todos los datos de la Noticia</p> <ul style="list-style-type: none">• Fecha de emisión• Fecha de caducidad• Título• Texto• Hora de emisión• Hora de caducidad <p>y permite:</p> <ul style="list-style-type: none">• Aceptar Crear noticia• Cancelar operación. Ver Alternativa 1: "Cancelar"
3. Introduce todos los datos de Noticia: <ul style="list-style-type: none">• Título• Texto	

<ul style="list-style-type: none"> • Fecha de emisión • Fecha de caducidad • Hora de emisión • Hora de caducidad 	
4. Selecciona la opción de aceptar Crear noticia	
	5. Valida los datos. Si hay datos incompletos, ver Alternativa 2: “Existen datos incompletos”. Si hay datos incorrectos, ver Alternativa 3: “Existen datos incorrectos”.
	6. Crea la Noticia.
	7. El caso de uso termina.
Flujos alternos	
Alternativa 1. “Cancelar”	
Acción del actor	Respuesta del sistema
1. Selecciona la opción “Cancelar operación”.	
	2. Regresa a la vista anterior.
	3. El caso de uso termina.
Alternativa 2. “Existen datos incompletos”	
Acción del actor	Respuesta del sistema
	<ol style="list-style-type: none"> 1. Muestra un indicador sobre los campos incompletos. 2. Regresa al paso 3 del Flujo Normal de Eventos.

Alternativa 3. "Existen datos incorrectos"	
Acción del actor	Respuesta del sistema
	<ol style="list-style-type: none"> 1. Muestra indicador sobre los campos incorrectos. 2. Regresa al paso 3 del Flujo Normal de Eventos.
Poscondiciones	El administrador crea una noticia satisfactoriamente.

Tabla 2.2 Descripción textual del caso de uso: Crear noticia

Caso de uso	
CU-2	Crear mensaje
Propósito	Permitir al actor crear un nuevo mensaje en el portal web.
Actores	Usuario
Resumen	El caso de uso inicia luego de que se busca un mensaje en el sistema (ver Caso de Uso 4.8: Listar mensaje), el actor accede a la opción Crear mensaje, el portal web brinda la posibilidad de introducir los datos para crear el Mensaje, el actor introduce los datos del mensaje, el portal web crea el mensaje, el caso de uso termina.
Precondiciones	
Flujo Normal de los Eventos	
Acción del actor	Respuesta del sistema
1. El caso de uso inicia cuando el actor accede a la opción Crear mensaje.	

	<p>2. Muestra los campos predeterminados:</p> <ul style="list-style-type: none"> • Título • Texto <p>Brinda la posibilidad de introducir todos los datos del Mensaje:</p> <ul style="list-style-type: none"> • Título • Texto <p>y permite:</p> <ul style="list-style-type: none"> • Aceptar • Cancelar operación. Ver alternativa 1: "Cancelar"
<p>3. Introduce todos los datos del Mensaje:</p> <ul style="list-style-type: none"> • Título • Texto 	
<p>4. Selecciona la opción Aceptar</p>	
	<p>5. Valida los datos. Si hay datos incompletos, ver Alternativa 2: "Existen datos incompletos". Si hay datos incorrectos, ver Alternativa 3: "Existen datos incorrectos"</p>
	<p>6. Crea el Mensaje.</p>
	<p>7. El caso de uso termina.</p>
<p>Flujos alternos</p>	
<p>Alternativa 1. "Cancelar"</p>	
<p>Acción del actor</p>	<p>Respuesta del sistema</p>

1. Selecciona la opción “Cancelar”.	
	2. Regresa a la vista anterior.
	3. El caso de uso termina.
Alternativa 2. “Existen datos incompletos”	
Acción del actor	Respuesta del sistema
	<ol style="list-style-type: none"> Muestra un indicador sobre los campos incompletos. Regresa al paso 3 del Flujo Normal de Eventos.
Alternativa 3. “Existen datos incorrectos”	
Acción del actor	Respuesta del sistema
	1. Muestra un indicador sobre los campos incorrectos.
	2. Regresa al paso 3 del Flujo Normal de Eventos .
Poscondiciones	Se creó un mensaje en el sistema.

Tabla 2.3 Descripción textual del caso de uso: Crear mensaje

Caso de uso	
CU-3	Consultar citas pendientes
Propósito	Permitir al actor consultar las citas pendientes.
Actores	Paciente

Resumen	El caso de uso inicia cuando el actor accede a la opción “Consultar citas pendientes”. El portal web muestra una lista de las citas que el usuario tiene pendientes. El portal brinda la posibilidad de introducir criterios de búsqueda para localizar citas determinadas, el actor introduce los datos que considera como criterios para realizar una búsqueda, el portal busca y muestra las citas que cumplen con los criterios de búsqueda
Precondiciones	
Flujo Normal de los Eventos	
Acción del actor	Respuesta del sistema
1. El caso de uso inicia cuando el actor accede a la opción “Consultar citas pendientes”.	
	2. Muestra una lista de las citas pendientes. 3. Brinda la posibilidad de introducir los criterios elementales de búsqueda: <ul style="list-style-type: none"> • Fecha de inicio. • Fecha de fin. • Cancelar la operación. Ver Alternativa 1: “Cancelar”
4. Introduce o selecciona los datos que considera como criterios para realizar una búsqueda y selecciona la opción de Buscar.	
	5. Busca la cita que cumple con los criterios de búsqueda.
	6. Muestra un listado de las citas que cumplen con los criterios de búsqueda.

	7. El caso de uso termina.
Flujos alternos	
Alternativa 1. “Cancelar”	
Acción del actor	Respuesta del sistema
1. Selecciona la opción “Cancelar”.	
	2. Regresa a la pantalla de bienvenida del portal web.

Tabla 2.4 Descripción textual del caso de uso: Consultar citas pendientes

2.4 Características del sistema

Como resultado de la investigación desarrollada y teniendo en cuenta que los usuarios que van a acceder al portal son las personas atendidas en instituciones médicas, así como el personal que labora en las mismas, se llegó a la conclusión de darle solución al problema planteado mediante la realización de una aplicación web, específicamente un portal web para pacientes.

Este tipo de aplicación:

- Constituye una puerta de entrada dentro de las prácticas médicas para los pacientes.
- Provee una comunicación segura de dos vías, entre los pacientes y sus proveedores de atención médica.
- Ofrece convenientemente servicios las 24 horas del día, a diferencia de los consultorios de salud, lo que posibilita que los pacientes puedan interactuar con el personal responsable de su atención a conveniencia.
- Agiliza el acceso a los servicios que brinda la institución así como a parte de su información que es lo que más valoran los pacientes.

La solución desarrollada permite a los usuarios ver noticias de interés tanto para la institución como para los pacientes; realizar consultas tanto a un resumen de la Historia Clínica como a los resultados de los exámenes de laboratorio indicados y a las citas pendientes en la institución, siempre y cuando lo necesite. Brinda además un marco que posibilita el libre intercambio de información con el personal médico. Dicho

intercambio se puede realizar en ambas vías, propiciando así la aclaración y discusión de temas de interés para todos los participantes.

Conclusiones

En el presente capítulo se obtuvo el Modelo de Dominio que permitió una mejor comprensión del problema y una identificación positiva de las principales funcionalidades, desglosadas detalladamente mediante los requisitos funcionales y no funcionales. Esto propició la realización del Modelo de Casos de Uso del Sistema en correspondencia con los requisitos previamente identificados, sentando las bases para el diseño e implementación del portal.

CAPÍTULO 3 DISEÑO DEL PORTAL WEB PARA PACIENTES

En el presente capítulo se realiza una descripción detallada del Portal web del paciente para el Sistema de Información Hospitalaria del CESIM, describiéndose su arquitectura así como los patrones de diseño que se ponen en práctica en su desarrollo. También, se presentan los Diagramas de Clases del Diseño que forman parte del Modelo de Diseño y se ofrece una breve descripción de las clases a utilizar para el almacenamiento de la información del portal mediante el Modelo de Datos.

3.1 Descripción de la arquitectura, fundamentación

La arquitectura de software se define como el concepto de más alto nivel de un sistema en su entorno. Incluyendo el ajuste con la integridad del sistema, con las restricciones económicas, las preocupaciones estéticas y el estilo. No se limita a un enfoque interior, sino que tiene en cuenta el sistema en su totalidad dentro del entorno de usuario y el de desarrollo, un enfoque exterior. La misma tiene que ver con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requerimientos no funcionales, como la confiabilidad, escalabilidad, portabilidad y disponibilidad. (37)

Para el desarrollo del Portal web del paciente y teniendo en cuenta que se asume la arquitectura definida para el Sistema de Información Hospitalaria del CESIM, se define como línea base, la implementación del patrón de diseño arquitectónico Modelo Vista Controlador (MVC), debido a que el mismo permite separar en tres componentes distintos los datos de la aplicación, la interfaz de usuario y la lógica de control.

La utilización de este patrón permite además desacoplar la vista del modelo y la reutilización de los componentes. Lo que brinda la posibilidad de realizar modificaciones en la vista con un impacto mínimo en la lógica de negocio o en los datos de la aplicación. En general el mismo posibilita que elementos de una capa determinada sean modificados o sustituidos completamente causando el mínimo de alteraciones en otros elementos que lo utilicen.

La capa de presentación está formada principalmente por las páginas XHTML, compuestas por formularios que mediante controles JSF, Seam UI, Facelets y RichFaces obtienen y validan los datos provistos por el usuario en las operaciones que realiza el mismo. Con el uso de los componentes de cada uno de los controles antes mencionados se logra una interfaz con un diseño más rico y natural para el usuario. La

transferencia de datos se realiza con componentes Ajax4jsf, logrando una mejor interacción con el Portal web del paciente.

La capa de negocio está formada por las clases controladoras, encargadas de definir la lógica del negocio del portal. Dichas clases realizan el manejo de los datos capturados en la capa de presentación.

La capa de acceso a datos está formada por los componentes de Hibernate que permiten cargar, modificar, eliminar y persistir la información en la base de datos una vez que la misma sea validada. El uso de Hibernate brinda la posibilidad al desarrollador abstraerse del gestor de base de datos utilizando el mapeo de las tablas, que le posibilita realizar consultas en un lenguaje de objetos.

3.2 Modelo de Diseño

El modelo de Diseño es un modelo de objetos, con el objetivo de describir el flujo de realización de los casos de uso, el mismo está compuesto por artefactos que engloban todas las clases del diseño, subsistemas, paquetes, colaboraciones y las relaciones entre ellos. Principalmente el Modelo de Diseño se centra en el impacto que puede tener sobre el sistema en desarrollo los requisitos funcionales y no funcionales, junto a otras restricciones relacionadas con el entorno de implementación. (36)

Generalmente durante la elaboración del diseño se utilizan patrones, que expresan esquemas para definir las relaciones entre las clases y los objetos con los que se construyen los sistemas de software. Dichos patrones 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. Un patrón de diseño resulta ser una solución a un problema de diseño que pueda presentarse durante el desarrollo del software. Esto permite al diseñador decidir si es necesario o no emplear un patrón específico. (11)

Para la definición del diseño del Portal web del paciente se tuvo en cuenta una serie de patrones, entre los mismos se encuentran los Patrones de Software para la Asignación General de Responsabilidad (GRASP, por sus siglas en inglés), que son patrones generales de software para la asignación de responsabilidades.

Se utilizó el patrón Experto o Experto en información con el objetivo básico de asignarle a las clases las tareas que podían realizar según la información que poseían, es decir, la responsabilidad de la creación de un objeto o la implementación de un método debe recaer sobre la clase que conoce toda la información necesaria para crearlo, para lo cual se utilizó el patrón Creador debido a que el mismo permite identificar quién debe ser el responsable de la creación de nuevos objetos o clases.

También fue utilizado el patrón Alta cohesión que determina que la información que almacena una clase debe de ser coherente y debe de estar, en la medida de lo posible, relacionado con la clase. Además se hace uso del patrón Bajo acoplamiento, para mantener las clases lo menos ligadas entre sí posible, lo que posibilita que en caso de producirse una modificación en una de ellas, tenga la mínima repercusión posible en el resto de las clases, potenciando la reutilización y disminuyendo la dependencia entre las clases.

En esta fase del desarrollo del Portal web del paciente se elaboran los Diagramas de Clases de Diseño, donde se presentan las interfaces y controladores, así como las relaciones existentes entre ellos. La presentación de dichos diagramas es de gran relevancia, debido que permiten la visualización, especificación y documentación de modelos estructurales.

También en esta fase se definen los Diagramas de Interacción, los mismos son artefactos que muestran gráficamente las relaciones entre los objetos. Este diagrama se usa para visualizar, especificar, construir y documentar o modelar un flujo de control particular de un caso de uso.

Con un uso más específico, dentro de los diagramas anteriormente mencionados se encuentran los diagramas de Secuencia y los Diagramas de Colaboración. Los primeros en este caso son usados en el diseño y los segundos en el análisis.

Para el diseño del Portal web del paciente del Sistema de Información Hospitalaria del CESIM se define una estructura de paquetes que favorece la división del Portal web en fragmentos manejables para la implementación. Empleándose el criterio de empaquetamiento por procesos y por clases, siguiendo la estructura de procesos definida en el sistema. Los paquetes que hacen referencias a procesos están compuestos por subcarpetas que responden directamente a la realización de los casos de uso y contienen un Diagrama de Clases y los respectivos Diagramas de Secuencia.

Se crea un paquete Repositorio de clases que contiene tres subpaquetes: el de las Vistas, el de las Sesiones y el de las Entidades. En el caso del paquete de las Vistas, este incluye los contenidos web referentes a las páginas clientes y los formularios que componen las mismas. En el de las Sesiones se agrupan las clases controladoras autogeneradas por el entorno de desarrollo y las clases controladoras personalizadas. Por último en el subpaquete de las Entidades se encuentran las Entidades personalizadas. En el caso de las Entidades autogeneradas, estas se obtienen del proceso de mapear la base de datos, y las personalizadas son aquellas que son modificadas para obtener un mejor funcionamiento a la hora de cumplir con una función u objetivo específico del sistema.

3.2.1 Diagrama de Paquetes

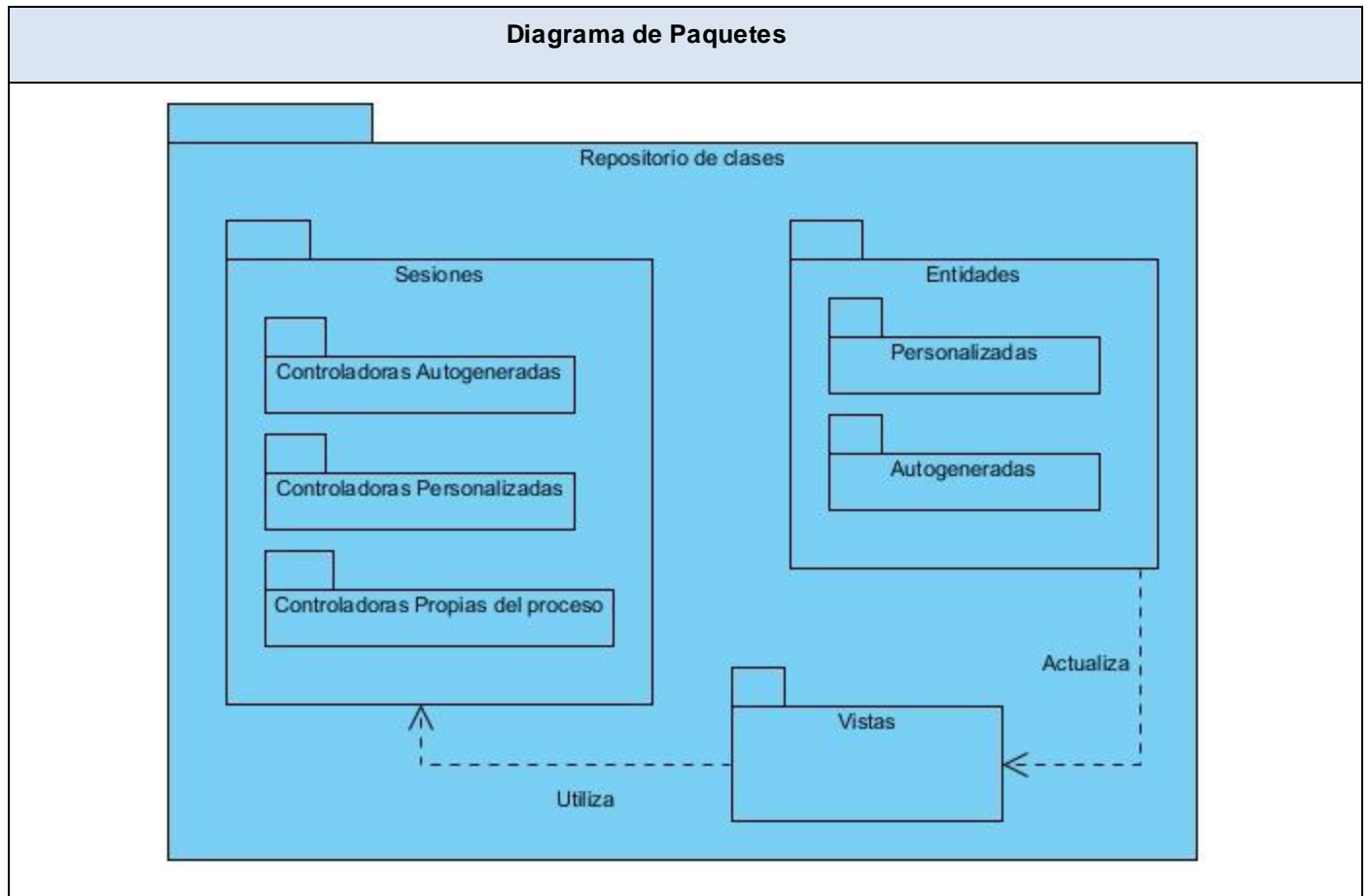


Figura 3.1 Diagrama de paquetes

3.2.2 Diagramas de Clases del Diseño

Para la realización de los Diagramas de Clases de Diseño se utilizaron los estereotipos web, que no son más que una representación gráfica de los componentes a los cuales hacen referencia. En estos diagramas quedan reflejados las relaciones entre todos los componentes del sistema. (29) (38)

Dichos diagramas muestran el flujo que se realiza cuando una página servidora construye una página cliente, la cual contiene los formularios que pueden actualizar directamente a las entidades o enviar peticiones a la página servidora, estas últimas son las encargadas de invocar los métodos de las clases controladoras, las que pueden consultar o modificar entidades.

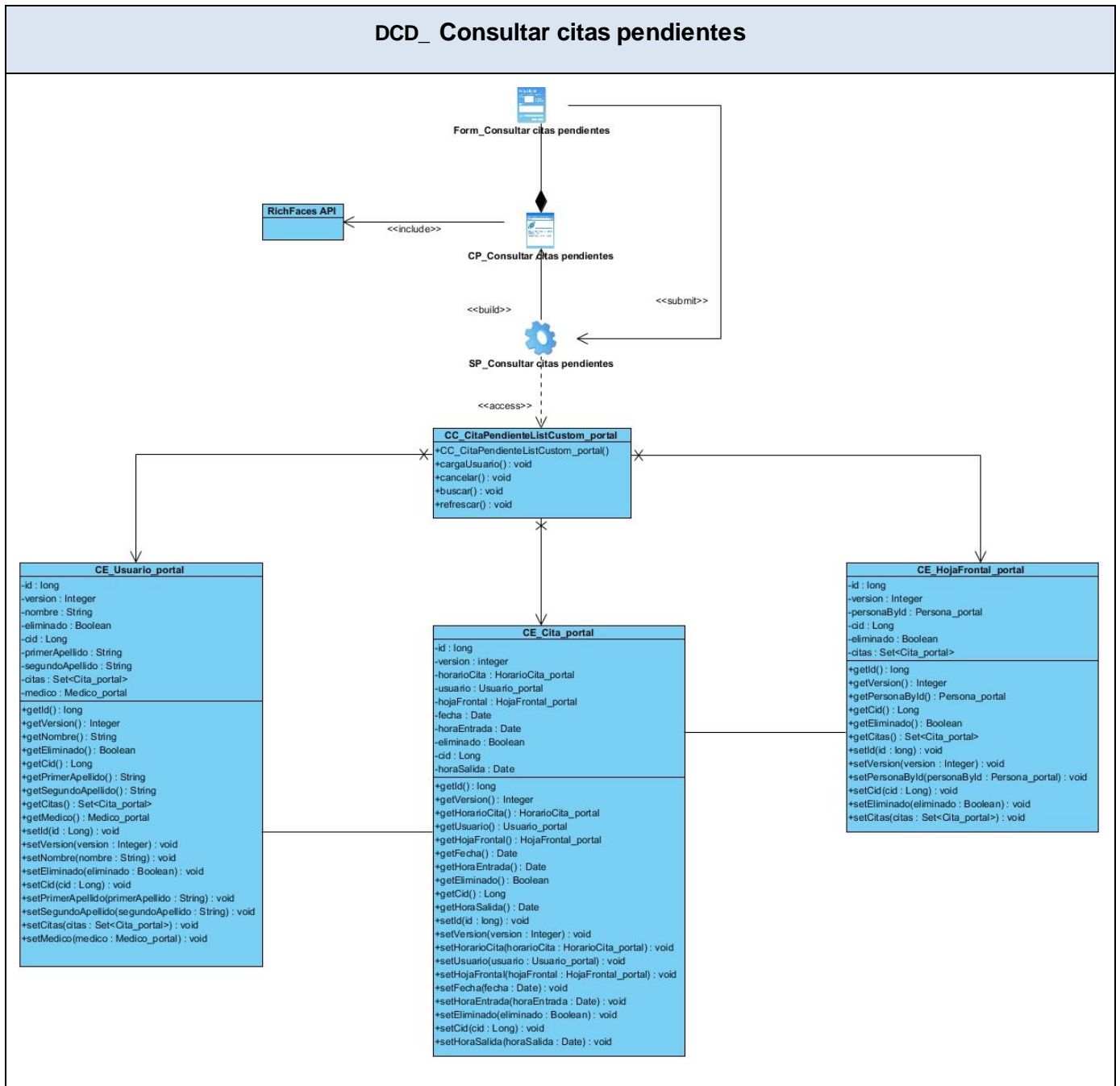


Figura 3.2 Diagrama de Clases del Diseño: Consultar citas pendientes

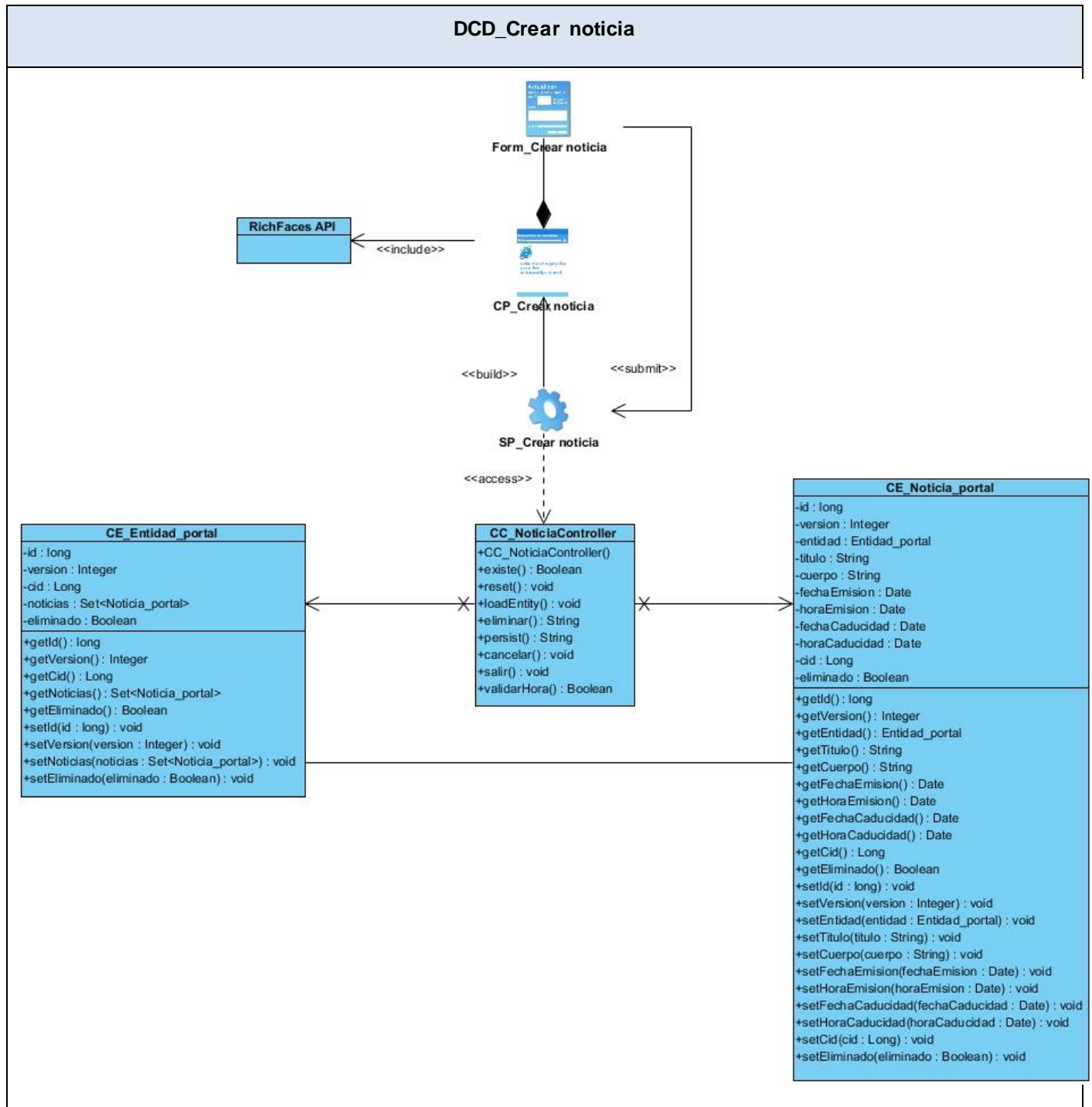


Figura 3.3 Diagrama de Clases del Diseño: Crear noticia

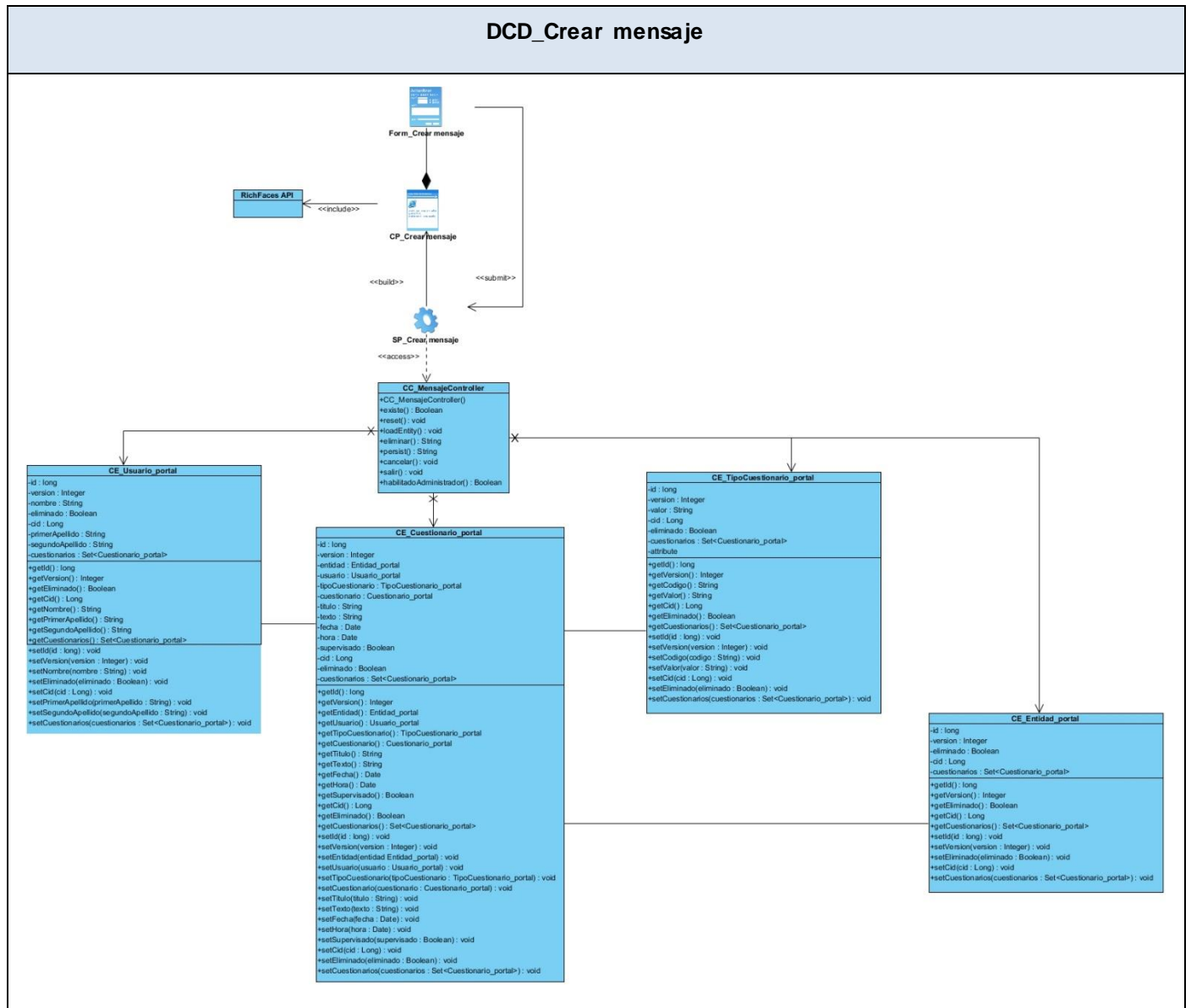


Figura 3.4 Diagrama de Clases del Diseño: Crear mensaje

Las clases del diseño se agrupan en:

Páginas servidoras: están compuestas por componentes Facelets, RichFaces, JSF, Seam UI, así como por el código HTML que con todos los elementos antes mencionados se ejecutan en el servidor web, permitiendo a su vez que se generen las páginas clientes que son representadas por el navegador web.

Páginas clientes: están compuestas por código HTML, CSS y JavaScript. Dichas páginas son interpretadas por los navegadores web, brindándole al usuario una interfaz con la cual puede interactuar con el sistema.

Formulario: es una sección de código HTML en el que se agregan los diferentes campos de entrada o de confirmación, así como los botones, que como mínimo ha de haber uno, el de envío. Los usuarios interactúan con estos controles a menudo introduciendo texto, seleccionando objetos de un menú, entre otras opciones. Esta información es enviada al servidor, donde es procesada para su posterior utilización.

Controladoras: en estas clases radica la implementación de un caso de uso o de un proceso en dependencia de la complejidad de los mismos.

3.2.3 Descripción de las clases y sus atributos

A continuación se muestra la descripción de dos de las principales clases que han sido identificadas, con el objetivo de brindar una mejor comprensión de los diagramas anteriores y del sistema en cuestión.

Nombre: MensajeController	
Tipo de clase: controladora.	
Atributo	Tipo
titulo	String
texto	String
tipo	String
fecha	Date
userPortal	Usuario_portal
pregunta	Cuestionario_portal
supervisado	Boolean
inicie	Boolean
preguntald	Long
Para cada responsabilidad:	
Nombre:	loadEntity(): void
Descripción:	Permite cargar un mensaje determinado que se encuentra persistido en la base de datos.
Nombre:	eliminar(): String
Descripción:	Permite eliminar un mensaje determinado, cambiando el estado de dicho atributo en la base de datos.

Nombre:	persist(): String
Descripción:	Permite persistir un mensaje en la base de datos.
Nombre:	cancelar(): void
Descripción:	Permite cancelar la operación que se está realizando, restableciendo los valores por defecto de los diferentes atributos de un mensaje.
Nombre:	habilitadoAdministrador(): boolean
Descripción:	Permite conocer si un usuario determinado es un administrador del sistema. Retorna True en caso afirmativo, falso en caso contrario.

Tabla 3.1 Descripción de la clase controladora: MensajeController

Nombre: NoticiaController	
Tipo de clase: controladora.	
Atributo	Tipo
inicie	Boolean
titulo	String
cuerpo	String
fechaEmision	Date
horaEmision	Date
fechaCaducidad	Date
horaCaducidad	Date
horaEmisionM	String
horaEmisionH	String
amEmisionHora	String
horaCaducidadM	String
horaCaducidadH	String
amCaducidadHora	String
Para cada responsabilidad:	
Nombre:	loadEntity(): void
Descripción:	Permite cargar una noticia determinada que se encuentra persistida en la base de datos.
Nombre:	eliminar(): String

Descripción:	Permite eliminar una noticia determinada, cambiando el estado de dicho atributo en la base de datos.
Nombre:	persist(): String
Descripción:	Permite persistir una noticia en la base de datos.
Nombre:	cancelar(): void
Descripción:	Permite cancelar la operación que se está realizando, restableciendo los valores por defecto de los diferentes atributos de una noticia.
Nombre:	validaHora(): boolean
Descripción:	Permite validar que la fecha de emisión de la noticia no sea mayor que la fecha de caducidad.

Tabla 3.2 Descripción de la clase controladora: *NoticiaController*

Nombre: HcController	
Tipo de clase: controladora.	
Atributo	Tipo
usuarioPortal	Usuario_portal
hojaf	HojaFrontal_portal
iniciado	boolean
listHojasConsulta	List<HojaConsulta_portal>
listHojasHospitalizacion	List<HojaHospitalizacion_portal>
Para cada responsabilidad:	
Nombre:	cargarHojasConsulta(): void
Descripción:	Carga las hojas de consultas.
Nombre:	cargarHojasHospitalizacion (): void
Descripción:	Carga las hojas de Hospitalización.
Nombre:	cargarUsuario(): void
Descripción:	Determina el usuario que se encuentra autenticado en el sistema.
Nombre:	salir(): void
Descripción:	Permite salir de la interfaz en la que se encuentra el usuario.

Tabla 3.3 Descripción de la clase controladora: *HcController*

3.3 Modelo de Datos

El Modelo de Datos permite describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan los mismos entre sí. Es usado para definir el mapeo entre las clases del diseño y las estructuras de datos. Está compuesto por entidades, atributos y sus relaciones:

Las entidades representan objetos del mundo real con existencia independiente, es decir, se diferencian unívocamente de otras, incluso siendo del mismo tipo.

Los atributos son las características que definen o identifican a una entidad.

Las relaciones constituyen dependencias existentes entre entidades, lo que permite la asociación de las mismas. (39)

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

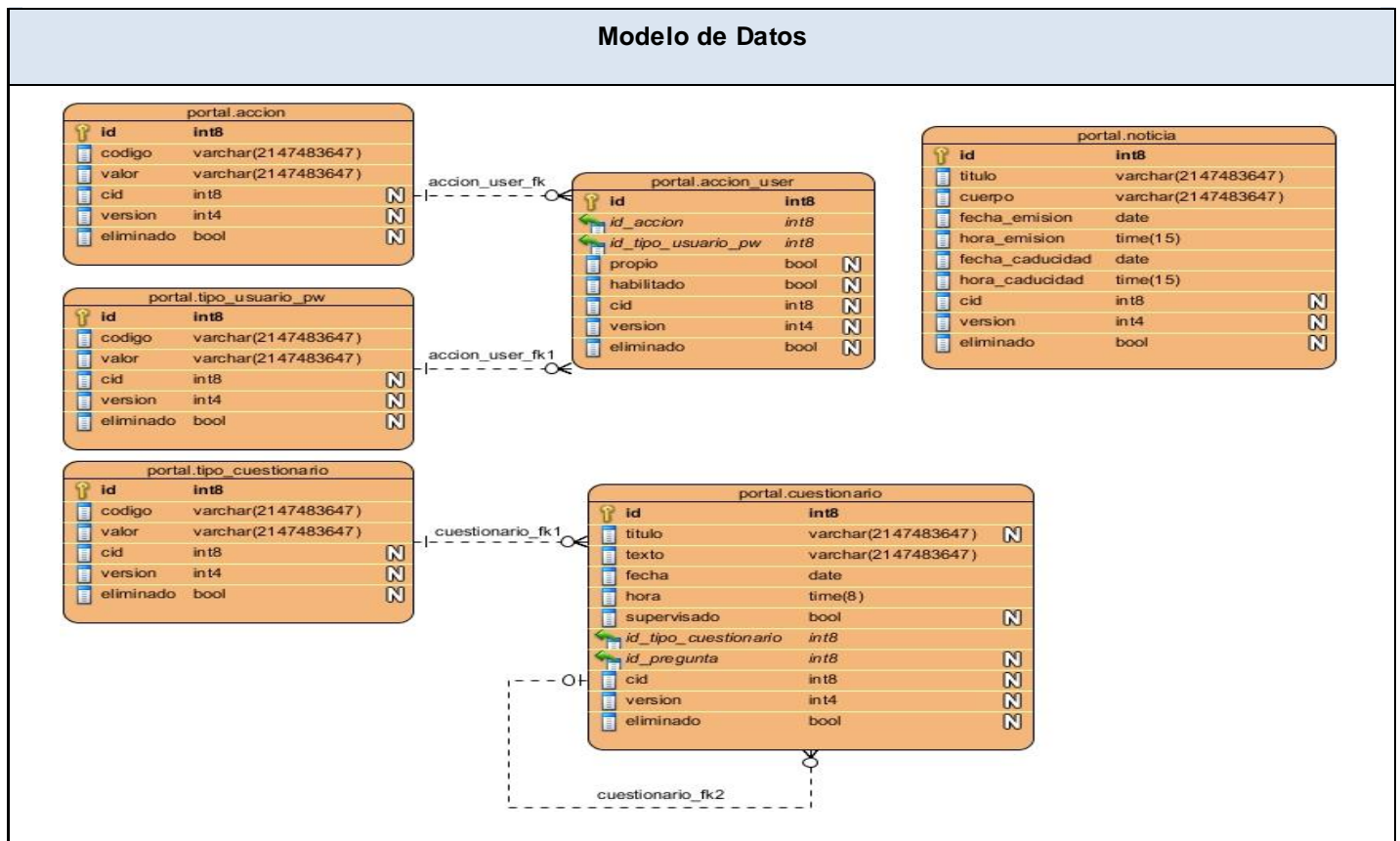


Figura 3.5 Modelo de Datos

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

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

Atributo	Tipo	Descripción
Id	long	Identificador necesario en cada entidad para las referencias en las relaciones entre tablas.
version	integer	Indica con qué versión de la entidad se está trabajando. Es usado para garantizar que se está trabajando con la versión de la entidad más actualizada que existe en la base de datos.
cid	long	Permite identificar quién realiza alguna acción sobre la entidad.

Tabla 3.4 Descripción de los atributos comunes entre todas las entidades

Noticia_portal		
La tabla contiene los datos necesarios para la gestión de noticias.		
Atributo	Tipo	Descripción
titulo	varchar	Título de la noticia
cuerpo	varchar	Texto que conforma la noticia
fecha_emision	date	Fecha en que se emitirá la noticia
hora_emision	date	Hora en la que se emitirá la noticia

fecha_caducidad	date	Fecha en la que caduca la noticia
hora_caducidad	date	Hora en la que caduca la noticia

Tabla 3.5 Descripción de la tabla: Noticia_portal

Cuestionario_portal		
La tabla contiene información que permite la gestión de mensajes.		
Atributo	Tipo	Descripción
titulo	varchar	Título del mensaje
texto	varchar	Texto del mensaje
fecha	date	Fecha en la que fue creado el mensaje
hora	date	Hora en la que fue creado el mensaje
supervisado	boolean	Atributo que permite determinar si un mensaje ha sido supervisado

Tabla 3.6 Descripción de la tabla: Cuestionario_portal

TipoCuestionario_portal		
La tabla contiene información referente a las citas de un paciente determinado.		
Atributo	Tipo	Descripción
código	varchar	Código
valor	varchar	Valor

Tabla 3.7 Descripción de la tabla: TipoCuestionario_portal

Conclusiones

Con la elaboración de los diagramas de clases del diseño correspondientes a los casos de usos del sistema con los que cuenta el portal y la definición del modelo de datos, se dio paso a la implementación del portal.

CAPÍTULO 4 IMPLEMENTACIÓN DEL PORTAL WEB PARA PACIENTES

En el presente capítulo se introduce el flujo de trabajo de implementación, a partir de los resultados obtenidos en el diseño. Inicialmente se presenta la estrategia de integración definida entre el portal y el Sistema de Información Hospitalaria, en el que se detalla la estructura de las entidades utilizadas por el portal web. Luego se muestra el Modelo de Implementación, que está compuesto por el Diagrama de Despliegue y el Diagrama de Componentes. También se exponen aspectos referentes a la seguridad del sistema, las estrategias de codificación, así como la forma en que se tratarán los errores.

4.1 Estrategia de integración

El Portal web del paciente no responde a un área específica de las instituciones hospitalarias; este constituye la vía que permite mantener un flujo de comunicación estable entre los pacientes y la institución hospitalaria encargada de su salud. Por tal motivo y teniendo en cuenta la necesidad por la cual fue diseñado, el mismo se integra como un módulo más del Sistema de Información Hospitalaria del CESIM.

Con el objetivo de lograr uniformidad con respecto a los demás componentes y módulos del Sistema de Información Hospitalaria del CESIM, se hace uso de la clase Usuario, la cual permite conocer qué usuario se encuentra conectado al portal web y obtener los datos del mismo. También se hace uso de la interfaz IActiveModule con el objetivo de conocer el módulo y entidad donde se encuentra el usuario que está haciendo uso del sistema. La interfaz IBitacora se utiliza para llevar un seguimiento de las acciones que realizan los usuarios en el sistema.

Para lograr la integración del portal con el Sistema de Información Hospitalaria se siguen los siguientes pasos:

1. Integrar la base de datos: hacer una salva de los cambios realizados localmente en la estructura de la base de datos del portal e integrarlos al servidor central de base de datos del sistema.
2. Extraer todos los recursos (mensajes, páginas, reglas y procesos de negocio, compilados, reportes, íconos y recursos): se integran al servidor de aplicaciones.

Con esta estrategia se garantiza rapidez y simplicidad en el proceso de integración, por lo que en caso de generarse algún problema durante este proceso, se realizaría la restauración de forma inversa al proceso anteriormente descrito.

4.2 Modelo de Implementación

El Modelo de Implementación consiste en obtener una visión general de lo que tiene que ser implementado, permite planificar las integraciones de sistemas necesarias en cada iteración e implementar las clases y subsistemas definidos durante el diseño. (36) (12)

4.2.1 *Diagrama de Componentes*

Un Diagrama de Componentes representa cómo un software es dividido en componentes y muestra las dependencias entre los mismos. Son usados para modelar y documentar cualquier arquitectura. En él se sitúan librerías, tablas, archivos, ejecutables y documentos que formen parte del sistema. (29)

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

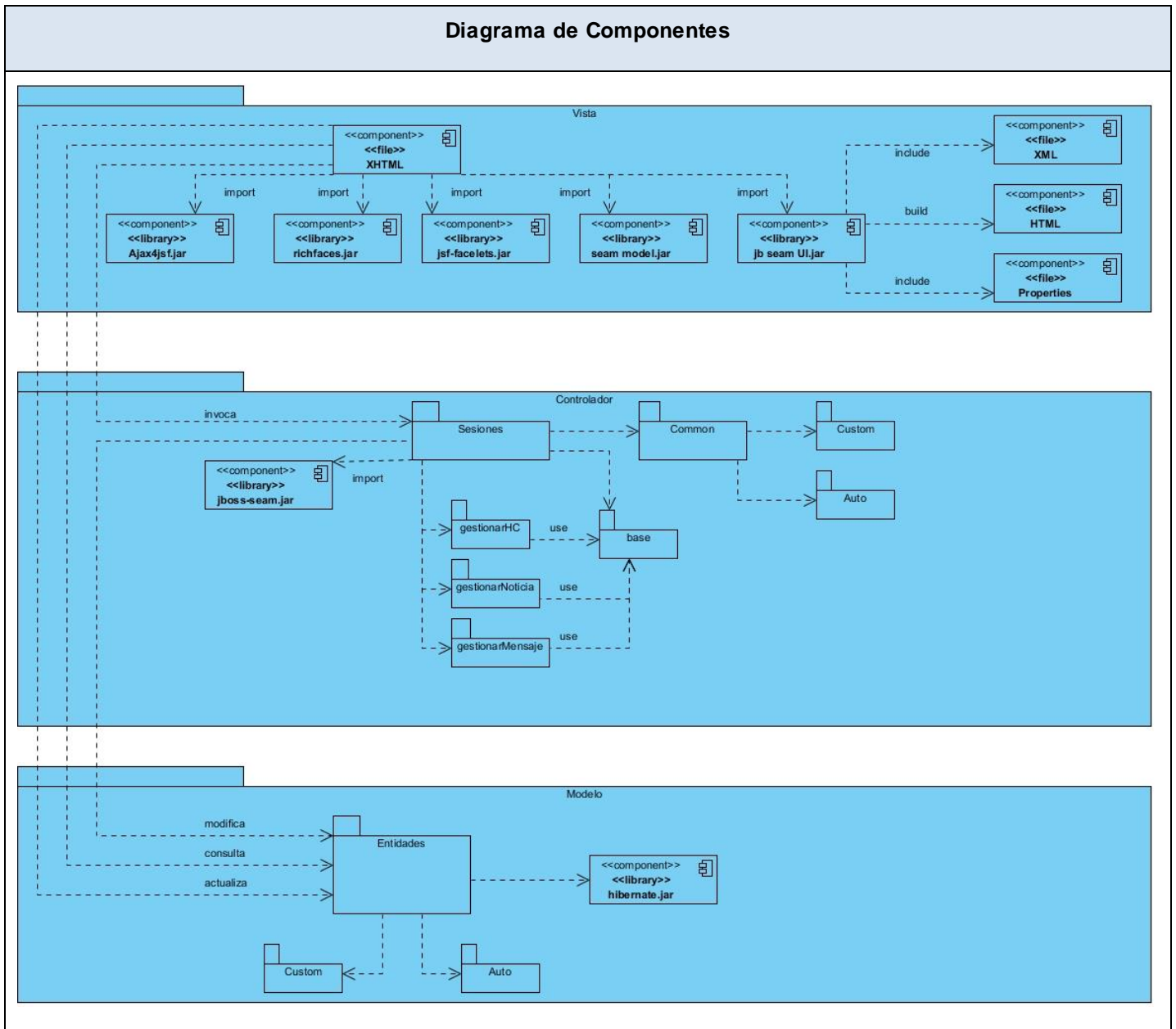


Figura 4.1 Diagrama de Componentes

El diagrama anterior muestra cómo se encuentra estructurado el sistema en componentes. La Vista está formada por las páginas XHTML que importan diversas librerías para su construcción. El paquete Sesiones hace uso de varias librerías y contiene las controladoras autogeneradas, las personalizadas y las controladoras propias del proceso, que son aquellas que permiten llevar a cabo los requisitos funcionales

del sistema. En el Modelo están presentes las entidades autogeneradas y las personalizadas, contenidas todas en el paquete Entidades, el cual utiliza la librería hibernate.jar. Estos paquetes se relacionan entre sí, las vistas consultan y actualizan las entidades e invocan a las controladoras y estas a su vez modifican las entidades.

4.2.2 Diagrama de Despliegue

El Diagrama de Despliegue muestra las relaciones físicas entre los componentes de hardware y software en el sistema final. Los elementos utilizados por este tipo de diagrama son nodos, componentes y asociaciones. (40)

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

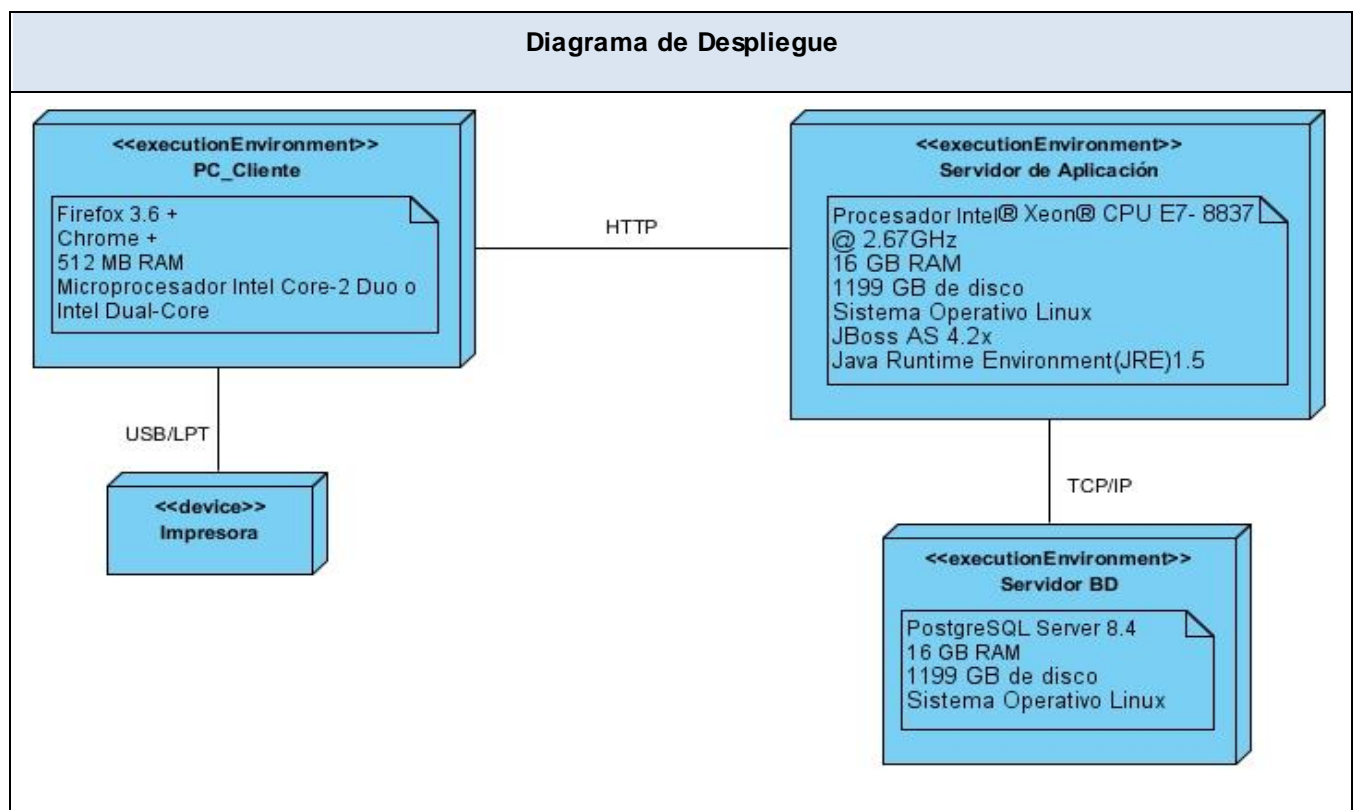


Figura 4.2 Diagrama de Despliegue

Para la implantación y utilización del portal web en un hospital, el usuario debe conectarse al mismo mediante una computadora cliente, haciendo uso de un navegador web, la cual puede tener conectada una

impresora mediante el puerto USB / LPT. Las peticiones por el protocolo HTTP serán procesadas por el servidor de aplicaciones, el cual enviará respuesta al cliente y emitirá peticiones por el protocolo TCP / IP (Transfer Control Protocol / Internet Protocol) hacia el servidor de base de datos.

4.3 Tratamiento de errores

El tratamiento de errores es uno de los aspectos más importantes a tener en cuenta durante el desarrollo del sistema. La validación de la información garantiza la corrección y precisión de todos los valores introducidos en la aplicación. Con el objetivo de elevar la calidad de la misma se realiza este proceso desde la fase de captación de los datos registrados por los usuarios. Las excepciones obtenidas como resultado del proceso de validación son tratadas, mostrándose finalmente al usuario.

4.4 Seguridad

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

4.5 Estrategias de codificación: estándares y estilos a utilizar

Un estándar de codificación completo comprende todos los aspectos de la generación de código, donde la legibilidad del código fuente repercute directamente en lo bien que se comprende un sistema de software y la mantenibilidad del código es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento.

El mejor método para asegurarse de que se mantenga un código de calidad es establecer un estándar de codificación sobre el que se efectuarán luego revisiones del código de rutinas.

4.5.1 Identación

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

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

4.5.2 Variables y constantes

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

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

Ejemplo: `mensajeController`.

Las constantes deben declararse con todas sus letras en mayúsculas.

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

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

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

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

4.5.4 Clases y objetos

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

Los nombres de las clases deben comenzar con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación PascalCasing. Ejemplo: `NoticiaController` ().

Para el caso de las instancias se comenzará con un prefijo que identificará el tipo de dato, este se escribirá en minúscula.

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

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

Conclusiones

En el presente capítulo se realizaron una serie de tareas que permitieron una mejor comprensión del sistema, haciéndolo más sencillo y rápido de implementar. Se realizaron los modelos de despliegue y de componentes. Así mismo se definió la forma en la que el sistema maneja los errores y cómo es tratada su seguridad. También se especificaron las restricciones del código, las cuales trazan normas a seguir en lo referente a la implementación. Se concluyó el desarrollo del Portal web para el paciente del Sistema de Información Hospitalaria del CESIM.

CONCLUSIONES

Con el desarrollo de la investigación:

1. El estudio realizado de los portales web existentes evidenció que los mismos no permiten la integración al Sistema de Información Hospitalaria del CESIM, debido a que no responden a los problemas derivados de la relación médico paciente dentro y fuera de las instituciones que hacen uso de dicho sistema; dicho estudio además propició valorar y definir qué información debe manejar un portal web para pacientes.
2. Para la realización del portal web, se asimiló la arquitectura propuesta por el departamento Sistemas de Gestión Hospitalaria del CESIM, por lo que las tecnologías y herramientas empleadas están basadas en dicha arquitectura; siendo todo el proceso guiado por la metodología de desarrollo RUP.
3. Se definieron los artefactos necesarios correspondientes a las fases del proceso de desarrollo, que enriquecido por el uso de los patrones de diseño, sentaron la bases para la implementación del portal web.
4. Se le dio cumplimiento al objetivo general de la investigación mediante el desarrollo del Portal web del paciente para el Sistema de Información Hospitalaria del CESIM.

RECOMENDACIONES

Se recomienda:

- Realizar un estudio para determinar los elementos necesarios en el Resumen de Historia Clínica del paciente.
- Migrar la solución desarrollada a un tipo de aplicación que no sea modular, que no esté embebida dentro del Sistema de Información Hospitalaria del CESIM, para lograr una mayor interacción con otros sistemas del CESIM que compartan la misma fuente de datos.
- Implementar una funcionalidad que permita realizar el registro de pacientes como usuarios del portal, directamente desde el mismo.

REFERENCIAS BIBLIOGRÁFICAS

1. **Waldez, Javier Quiroz.** *Sociedad del conocimiento y la información.* s.l. : Instituto Nacional de Estadísticas, Geografía e Informática .
2. **Martínez, Dr. José Negrete.** Historia de la Informática Médica. [En línea] 2006. [Citado el: 14 de Marzo de 2014.] <http://www.facmed.unam.mx/emc/computo/infomedic/historia.htm>.
3. **García Gómez, Juan Carlos.** *Portales de internet: concepto, tipología básica y desarrollo.* En: *El profesional de la información, 2001, julio-agosto, v. 10, n. 7-8, pp. 4-13.* 2001.
4. **Kenneth G Adler.** Web Portals in Primary Care: An Evaluation of Patient Readiness and Willingness to Pay for Online Services. *NCBI.* [En línea] Octubre - Diciembre de 2006. [Citado el: 14 de Marzo de 2013.] www.ncbi.nlm.nih.gov/pubmed/17213045.
5. —. **Who Uses the Patient Internet Portal? The Patient Site Experience.** *NCBI.* [En línea] Enero-Febrero de 2006. [Citado el: 14 de Marzo de 2014.] www.ncbi.nlm.nih.gov/pubmed/16221943.
6. **CureMD Healthcare.** CureMD. [En línea] 1997. [Citado el: 20 de enero de 2014.] <http://www.curemd.com/>.
7. **Athenahealth.** Athenahealth. [En línea] 2014. [Citado el: 20 de enero de 2014.] <http://www.athenahealth.com/knowledge-hub/patient-engagement/patient-portal.php>.
8. **Orion Health.** Orion. [En línea] 2013. [Citado el: 20 de enero de 2014.] <http://www.orionhealth.com/options/patient-portal>.
9. **Capítulo 5.** Cliente - Servidor. pág. 9, PDF.
10. **Sánchez, Carlos González.** Proyecto de fin de carrera. Aplicaciones en capas. [En línea] 28 de septiembre de 2004. [Citado el: 17 de enero de 2014.] <http://oness.sourceforge.net/proyecto/html/ch03s02.html>.
11. **Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides.** Design Patterns. Elements of Reusable Object-Oriented Software. s.l. : Addison Wesley, 2005.

12. **Pressman, Roger S.** Software Engineering, a practitioner's approach.(7th edición). s.l. : McGraw-Hill, 2011. ISBN 9780071267823.
13. **Díaz González, Yanette y Fernández Romero, Yenisleidy.** Revista Telem@tica. Patrón Modelo-Vista-Controlador. [En línea] 2012. [Citado el: 25 de enero de 2014.] <http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/15>.
14. **Ecured. EcuRed.** Lenguaje de programación Java. [En línea] [Citado el: 26 de enero de 2014.] <http://www.ecured.cu/index.php/Java>.
15. **Arquitectura de capas en sistemas de información. Ecured.** [En línea] [Citado el: 24 de Marzo de 2014.] http://www.ecured.cu/index.php/Arquitectura_de_capas_en_sistemas_de_informaci%C3%B3n#Tres_capas.
16. **Ghia, Dustin. Programación Práctica. JEE5 - Fundamentos.** [En línea] 31 de marzo de 2011. [Citado el: 15 de enero de 2014.] <http://programmabilis.blogspot.com/2011/03/i1-fundamentos-de-jee5.html>.
17. **Newton, Mark, y otros. WildFly. RichFaces.** [En línea] 2008. [Citado el: 3 de febrero de 2013.] <http://www.jboss.org/richfaces>.
18. **JBoss Community. Jboss ajax4jsf.** [En línea] 2007. [Citado el: 5 de febrero de 2013.] <http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html>.
19. **9 Tecnologías.** pág. 23, PDF.
20. **LIBROSWEB. HTML y XHTML.** [En línea] 2010. [Citado el: 10 de enero de 2014.] http://www.librosweb.es/xhtml/capitulo_1/html_y_xhtml.html.
21. **Orshalick, Jacob. Refcardz. SeamUI.** [En línea] [Citado el: 30 de noviembre de 2013.] <http://refcardz.dzone.com/refcardz/seam-ui>.
22. **Red Hat. SeamFramework.org.** [En línea] 2009. [Citado el: 25 de enero de 2014.] <http://www.seamframework.org/>.
23. **JBossCommunity. HIBERNATETools. Hibernate Tools for Eclipse and Ant.** [En línea] [Citado el: 15 de febrero de 2013.] <http://www.hibernate.org/subprojects/tools.html>.

24. **Universidad de los Andes, Departamento de Sistemas. Enterprise Java Bean 3. Características.** [En línea] [Citado el: 15 de febrero de 2013.] <http://sistemas.uniandes.edu.co/~isis3702/dokuwiki/lib/exe/fetch.php?media=principal:isis3702-ejb3.pdf>.
25. **Rondon Grados, Luis . JAVA J2EE. JPA - Java Persistence API .** [En línea] 28 de agosto de 2009. [Citado el: 15 de febrero de 2013.] <http://luchorondon.blogspot.com/2009/04/jpa-java-persistence-api.html>.
26. **Ecured. Ecured. JBoss.** [En línea] [Citado el: 25 de enero de 2014.] <http://www.ecured.cu/index.php/JBoss>.
27. —. **Ecured. Máquina Virtual de java.** [En línea] [Citado el: 26 de enero de 2014.] http://www.ecured.cu/index.php/M%C3%A1quina_virtual_de_java.
28. **Ivar Jacobson, Grady Booch, James Rumbaugh.** El proceso unificado de desarrollo de software. ISBN: 84-7829-036-2.
29. **Popkin Software and Systems.** Modelado de Sistemas con UML. pág. 20, PDF.
30. **Red Hat. Red Hat JBoss Developer Studio. JBoss Community.** [En línea] [Citado el: 20 de Marzo de 2014.] <https://www.jboss.org/products/devstudio.html>.
31. —. **JBoos Tools - Documentation. JBoos Tools.** [En línea] [Citado el: 20 de Marzo de 2014.] <http://tools.jboss.org/documentation/>.
32. **Abraham Silberschatz, Henry F. Korth, S. Sudarshan.** FUNDAMENTOS DE BASES DE DATOS (Cuarta Edición). s.l. : McGraw-Hill Inc. ISBN: 0-07-228363-7.
33. **PostgreSQL. What is PostgreSQL?** [En línea] 20 de mayo de 2009. [Citado el: 16 de febrero de 2013.] http://wiki.postgresql.org/wiki/FAQ#What_is_PostgreSQL.3F_How_is_it_pronounced.3F_What_is_Postgres.3F.
34. **Free Download Manager. Visual Paradigm para UML.** [En línea] marzo de 2007. [Citado el: 17 de febrero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

35. **Larman, Craig.** UML y Patrones. 2ª Edición. s.l. : Prentice Hall, 2003.
36. **Pressman, Roger.** Ingeniería del Software, un enfoque práctico. s.l. : Mc-Graw Hill, 2002.
37. **Aramayo Fernández, David Ricardo.** Arquitectura de Software. s.l. : Universidad Tecmilenio, ITESM.
38. **Perdita Stevens, Rob Pooley.** Utilización de UML en Ingeniería del Software con Objetos y Componentes. s.l. : Addison Wesley, 2002.
39. **Pantaleón, Marta E. Zorrilla.** Modelos de datos. 2010.
40. **Sparx Systems Pty Ltd.** Diagrama de Despliegue. Enterprise Architect. [En línea] 2000-2013. [Citado el: 16 de Marzo de 2014.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.

BIBLIOGRAFÍA

- Waldez, Javier Quiroz. *Sociedad del conocimiento y la información*. s.l. : Instituto Nacional de Estadísticas, Geografía e Informática .
- Martínez, Dr. José Negrete. Historia de la Informática Médica. [En línea] 2006. [Citado el: 14 de Marzo de 2014.] <http://www.facmed.unam.mx/emc/computo/infomedic/historia.htm>.
- García Gómez, Juan Carlos. *Portales de internet: concepto, tipología básica y desarrollo*. En: *El profesional de la información, 2001, julio-agosto, v. 10, n. 7-8, pp. 4-13*. 2001.
- Kenneth G Adler. Web Portals in Primary Care: An Evaluation of Patient Readiness and Willingness to Pay for Online Services. *NCBI*. [En línea] Octubre - Diciembre de 2006. [Citado el: 14 de Marzo de 2013.] www.ncbi.nlm.nih.gov/pubmed/17213045.
- Who Uses the Patient Internet Portal? The Patient Site Experience. *NCBI*. [En línea] Enero-Febrero de 2006. [Citado el: 14 de Marzo de 2014.] www.ncbi.nlm.nih.gov/pubmed/16221943.
- CureMD Healthcare. CureMD. [En línea] 1997. [Citado el: 20 de enero de 2014.] <http://www.curemd.com/>.
- Athenahealth. Athenahealth. [En línea] 2014. [Citado el: 20 de enero de 2014.] <http://www.athenahealth.com/knowledge-hub/patient-engagement/patient-portal.php>.
- Orion Health. Orion. [En línea] 2013. [Citado el: 20 de enero de 2014.] <http://www.orionhealth.com/options/patient-portal>.
- *Capítulo 5. Cliente - Servidor*. pág. 9, PDF.
- Sánchez, Carlos González. Proyecto de fin de carrera. *Aplicaciones en capas*. [En línea] 28 de septiembre de 2004. [Citado el: 17 de enero de 2014.] <http://oness.sourceforge.net/proyecto/html/ch03s02.html>.
- Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides. *Design Patterns. Elements of Reusable Object-Oriented Software*. s.l. : Addison Wesley, 2005.

- Pressman, Roger S. *Software Engineering, a practitioner's approach.* (7th edición). s.l. : McGraw-Hill, 2011. ISBN 9780071267823.
- Díaz González, Yanette y Fernández Romero, Yenisleidy. *Revista Telemática. Patrón Modelo-Vista-Controlador.* [En línea] 2012. [Citado el: 25 de enero de 2014.] <http://revistatelematica.cujae.edu.cu/index.php/tele/article/view/15>.
- Ecured. EcuRed. *Lenguaje de programación Java.* [En línea] [Citado el: 26 de enero de 2014.] <http://www.ecured.cu/index.php/Java>.
- Arquitectura de capas en sistemas de información. *Ecured.* [En línea] [Citado el: 24 de Marzo de 2014.] http://www.ecured.cu/index.php/Arquitectura_de_capas_en_sistemas_de_informacion%C3%B3n#Tres_capas.
- Ghia, Dustin. *Programación Práctica. JEE5 - Fundamentos.* [En línea] 31 de marzo de 2011. [Citado el: 15 de enero de 2014.] <http://programmabilis.blogspot.com/2011/03/i1-fundamentos-de-jee5.html>.
- Newton, Mark, y otros. WildFly. *RichFaces.* [En línea] 2008. [Citado el: 3 de febrero de 2013.] <http://www.jboss.org/richfaces>.
- JBoss Community. *Jboss ajax4jsf.* [En línea] 2007. [Citado el: 5 de febrero de 2013.] <http://www.jboss.org/jbossajax4jsf/docs/devguide/en/html/Introduction.html>.
- *9 Tecnologías.* pág. 23, PDF.
- LIBROSWEB. *HTML y XHTML.* [En línea] 2010. [Citado el: 10 de enero de 2014.] http://www.librosweb.es/xhtml/capitulo_1/html_y_xhtml.html.
- Orshalick , Jacob . Refcardz. *SeamUI.* [En línea] [Citado el: 30 de noviembre de 2013.] <http://refcardz.dzone.com/refcardz/seam-ui>.
- Red Hat. *SeamFramework.org.* [En línea] 2009. [Citado el: 25 de enero de 2014.] <http://www.seamframework.org/>.
- JBossCommunity. *HIBERNATETools. Hibernate Tools for Eclipse and Ant.* [En línea] [Citado el: 15 de febrero de 2013.] <http://www.hibernate.org/subprojects/tools.html>.

- Universidad de los Andes, Departamento de Sistemas. Enterprise Java Bean 3. *Características*. [En línea] [Citado el: 15 de febrero de 2013.] <http://sistemas.uniandes.edu.co/~isis3702/dokuwiki/lib/exe/fetch.php?media=principal:isis3702-ejb3.pdf>.
- Rondon Grados, Luis . JAVA J2EE. *JPA - Java Persistence API* . [En línea] 28 de agosto de 2009. [Citado el: 15 de febrero de 2013.] <http://luchorondon.blogspot.com/2009/04/jpa-java-persistence-api.html>.
- Ecured. Ecured. *JBoss*. [En línea] [Citado el: 25 de enero de 2014.] <http://www.ecured.cu/index.php/JBoss>.
- Ecured. *Máquina Virtual de java*. [En línea] [Citado el: 26 de enero de 2014.] http://www.ecured.cu/index.php/M%C3%A1quina_virtual_de_java.
- Ivar Jacobson, Grady Booch, James Rumbaugh. *El proceso unificado de desarrollo de software*. ISBN: 84-7829-036-2.
- Popkin Software and Systems. *Modelado de Sistemas con UML*. pág. 20, PDF.
- Red Hat. Red Hat JBoss Developer Studio. *JBoss Community*. [En línea] [Citado el: 20 de Marzo de 2014.] <https://www.jboss.org/products/devstudio.html>.
- JBoos Tools - Documentation. *JBoos Tools*. [En línea] [Citado el: 20 de Marzo de 2014.] <http://tools.jboss.org/documentation/>.
- Abraham Silberschatz, Henry F. Korth, S. Sudarshan. *FUNDAMENTOS DE BASES DE DATOS (Cuarta Edición)*. s.l. : McGraw-Hill Inc. ISBN: 0-07-228363-7.
- PostgreSQL. *What is PostgreSQL?* [En línea] 20 de mayo de 2009. [Citado el: 16 de febrero de 2013.] http://wiki.postgresql.org/wiki/FAQ#What_is_PostgreSQL.3F_How_is_it_pronounced.3F_What_is_Postgres.3F.
- Free Download Manager. *Visual Paradigm para UML*. [En línea] marzo de 2007. [Citado el: 17 de febrero de 2013.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.
- Larman, Craig. *UML y Patrones. 2ª Edición*. s.l. : Prentice Hall, 2003.

- Pressman, Roger. *Ingeniería del Software, un enfoque práctico*. s.l. : Mc-Graw Hill, 2002.
- Aramayo Fernández, David Ricardo. *Arquitectura de Software*. s.l. : Universidad Tecmilenio, ITESM.
- Perdita Stevens, Rob Pooley. *Utilización de UML en Ingeniería del Software con Objetos y Componentes*. s.l. : Addison Wesley, 2002.
- Pantaleón, Marta E. Zorrilla. *Modelos de datos*. 2010.
- Sparx Systems Pty Ltd. . Diagrama de Despliegue. *Enterprise Architect*. [En línea] 2000-2013. [Citado el: 16 de Marzo de 2014.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.
- Puelles , Lizana Esther . El Rinconcito Informático. *Modelado de Sistemas con UML*. [En línea] 2000. [Citado el: 15 de enero de 2013.] <http://www.elrinconcito.com/articulos/modeladoUML/modeladoUML.html>.
- Orshalick, Jacob. Seam UI. *DZone*. [En línea] DZone, Inc., 1997-2014. <http://refcardz.dzone.com/refcardz/seam-ui>.
- Maldonado, Daniel M. El CoDiGo K. *Arquitectura de programación en 3 capas*. [En línea] 2007. [Citado el: 10 de diciembre de 2012.] <http://www.elcodigok.com.ar/2007/09/arquitectura-de-programacion-en-3-capas/>.
- Jaramillo M, Wilmer. Fedora People. *JBoss Application Server*. [En línea] 2006. [Citado el: 10 de febrero de 2013.] <http://wilmer.fedorapeople.org/files/presentations/JBoss.pdf>.
- Jamae, David y Johnson, Peter. *JBoss in Action*. 2009. pág. 10, PDF.
- García Carmona, Juan. Patrones. *GRASP: Alta cohesión y bajo acoplamiento*. [En línea] [Citado el: 12 de marzo de 2013.] <http://www.grasp-alta-cohesion-y-bajo-acoplamiento.html>.
- MASTERMAGAZINE. *Definición de Arquitectura de Software*. [En línea] 2007. [Citado el: 5 de enero de 2013.] <http://www.Definición de Arquitectura Software - Significado y definición de Arquitectura Software.html>.

- Kioskea.net. *Entorno cliente/servidor*. [En línea] [Citado el: 24 de enero de 2013.] <http://es.kioskea.net/contents/148-entorno-cliente-servidor>.
- Definición.DE. *Definición de notificación*. [En línea] 2008. [Citado el: 27 de noviembre de 2012.] <http://definicion.de/notificacion/>.
- Instituto de Física Aplicada del CSIC. *¿Qué es Java? Características del lenguaje Java*. [En línea] 1997. [Citado el: 20 de enero de 2013.] <http://www.iec.csic.es/criptonomicon/java/quesjava.html>.
- Conallen, Jim. *Building Web Applications with UML*. s.l. : Addison Wesley, 2000. ISBN: 0201615770.
- Who Uses the Patient Internet Portal The PatientSite Experience. *NCBI*. [En línea] [Citado el: 20 de Marzo de 2014.] www.ncbi.nlm.nih.gov/pubmed/16221943.

ANEXOS

Anexo 1: Interfaz de usuario para crear una noticia.

Anexo 1 Crear noticia

Anexo 2: Interfaz de usuario para crear un mensaje.

Anexo 2 Crear mensaje

Anexo 3: Interfaz de usuario para buscar una noticia

Titulo	Fecha de emisión	Fecha de caducidad
News	04/04/2014	04/04/2014
Relevante	04/04/2014	04/04/2014

Anexo 3 Buscar noticia

Anexo 4: Interfaz de usuario para buscar un mensaje

Anexo 4 Buscar mensaje

Anexo 5: Interfaz de usuario para listar mensajes

Anexo 5 Listar mensaje

Anexo 6: Interfaz de usuario para consultar citas pendientes


Anexo 6 Consultar citas pendientes

Anexo 7: Interfaz de usuario para consultar resultados de exámenes de laboratorio

Consultar resultados de exámenes de laboratorio Buscar...

Datos generales del paciente | **Datos laborales del paciente**

Datos generales del paciente No.H.C: 58972


Nombre: Administrador **Cédula:** 326541 **Tipo de paciente:** Titular
Primer apellido: Administrador **Fecha de nacimiento:** 03/02/2011 **Edad:** 3 años
Segundo apellido: - **Sexo:** - **ABO/Rh:** -

Seleccionar examen

- 10/06/2013
 - Hematología completa
 - Proteína-C-reactiva
- 11/06/2013
- 14/06/2013

Resultados

Datos del examen

Nombre: Proteína-C-reactiva **Sección:** Inmunología y Serología

Datos de la solicitud

Fecha: 10/06/2013 **Médico:** María Daniela Rosario Albornoz

Listado de resultados

Grupo	Aspecto	Descripción	Resultado
-	-	0 - 0.9 mg/dl	5.42 mg/dl

Microorganismos aislados

No existe información a mostrar.

Salir

© Universidad de las Ciencias Informáticas, 2010.


Anexo 7 Consultar resultados de exámenes de laboratorio

Anexo 8: Interfaz de usuario para consultar resumen de Historia Clínica

Resumen de historia clínica Buscar...

Datos generales del paciente | **Datos laborales del paciente**

Datos generales del paciente No.H.C: 58972


Nombre: Administrador **Cédula:** 326541 **Tipo de paciente:** Titular
Primer apellido: Administrador **Fecha de nacimiento:** 03/02/2011 **Edad:** 3 años
Segundo apellido: - **Sexo:** - **ABO/Rh:** -

Listado de hojas de consultas

Fecha: 27/06/2013

Hora de inicio: 07:29 PM **Hora de fin:** 07:33 PM **Médico:** ELPIDIO GARCIA

Especialidad: Medicina interna **Tipo:** General

Diagnóstico

Código	Descripción	Tipo
I10.Xa	Hipertensión (arterial) (benigna) (esencial) (maligna) (primaria) (sistémica)	Control

Listado de hojas de hospitalización

No existe información a mostrar.

Salir

© Universidad de las Ciencias Informáticas, 2010.

Anexo 8 Consultar Historia Clínica