
**Universidad de las Ciencias Informáticas**

**Facultad 7**

# Título: Implementación de los módulos Gestión de Reportes y Registro de Trasplantes del Sistema Alas NefroRed

**Trabajo de Diploma para optar por el título de**

**Ingeniero en Ciencias Informáticas**

**Autores: Beatriz Ricardo Velázquez**

**Iduanys Caballero Márquez**

**Tutores: Yanersy Díaz Colomé**

**Renier Ricardo Figueredo**

**Ciudad de La Habana, Junio 2009**

**"Año del 50 Aniversario del triunfo de la Revolución"**

## Declaración de Autoría

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

Para que así conste firmamos la presente a los 12 días del mes de junio del año 2009.

**Beatriz Ricardo Velázquez** **Ing. Renier Ricardo Figueredo**

_____ _____

**Firma de la Autora** **Firma del Tutor**

**Iduanys Caballero Márquez** **Ing. Yanersy Díaz Colomé**

_____ _____

**Firma del Autor** **Firma del Tutor**

*"La muerte no es verdad cuando se ha cumplido bien la obra de la vida."*

*José Martí*

**Agradecimientos**

*A la Revolución, que nos permitió hacer realidad nuestros sueños.*

*Agradecemos a todas las personas que nos han brindado su mano en los momentos difíciles.*

*A nuestros amigos, Leosdan, Osmany, Yarel, Alexis, por ayudarnos siempre, nunca los olvidaremos.*

*A nuestras amigas, Yanet, Irays, Mabel, Yanitza, Iselda, Lizandra y Aimé por brindarnos su apoyo en los momentos difíciles.*

*A nuestros tutores, Renier y Yanersy, por haber depositado todo su confianza en nosotros y siempre estuvieron a nuestro lado.*

*A todos nuestros profesores, que a lo largo de estos cinco años contribuyeron a nuestra formación profesional.*

*A nuestras familias, que han entregado todo por nuestro futuro, su cariño y dedicación que nunca se terminó.*

*A todos ¡Muchas Gracias!*

**De Beatriz**

*A mi papá, por su ayuda incondicional en todos estos años.*

*A mi mamá, por ayudarme siempre en todo lo que me hizo falta y por darme siempre ese cariño tan especial.*

*A mis abuelos queridos, son las personas más importantes de mi vida y a ellos los debo lo que soy.*

*A mis tías, por su ayuda incondicional en todo.*

*A Iduanys, por ser la persona más especial en todos estos años.*

**De Iduanys**

*A mis padres, por brindarme su apoyo siempre que los necesité y por esa dedicación tan infinita a mi formación como profesional.*

*A mis tíos y mi hermana, por su ayuda en los momentos más difíciles de la carrera.*

*A Beatriz, por ayudarme siempre y haber sido tan especial.*

***Dedicatoria***

**De Beatriz**

*A mis abuelos, este título es y será por siempre de ustedes, son las personas más importantes de mi vida.*

*A mis padres, que no solo me trajeron al mundo, sino que me guiaron por los mejores caminos, me brindaron su mano en los momentos difíciles y tristes, gracias por hacerme ver la vida más clara y por demostrarme que podía ser capaz de alcanzar nuevas metas para el futuro.*

A mis tías, gracias por ser mis segundas madres, esa dedicación que siempre recibí de *ustedes es una de las mejores cosas que he tenido en mi vida.*

*A mi única prima, que la quiero como una hermana, gracias por todos esos momentos de alegría que me das cuando más los necesito.*

*A mis amigos, por estar a mi lado cuando los necesité.*

*A mi novio y compañero de tesis, eres lo mejor que me ha pasado en estos años, gracias por ser mi brazo derecho en lo momentos difíciles, gracias por hacerme ver la vida de la forma correcta, gracias por todos los momentos lindos que me has permitido vivir.*

*A todos los que de una forma u otra me ayudaron a hacer mi sueño realidad.*

*A todos ¡Muchas Gracias!*

**De Iduanys**

*A mis padres, por todo el amor y la comprensión que me han dado, por haber sido mi guía en la vida, por haberme hecho el hombre que hoy soy, por su amor y dedicación hacia mí.*

*A mi abuelo, es para mí una gran alegría poder graduarme y que puedas estar presente, gracias abuelo por ser tan especial.*

*A mi hermana, por todo su cariño, por ser tan especial y por brindarme siempre todo su apoyo.*

*A mis amigos, por estar a mi lado cuando los necesité.*

*A mi novia y compañera de tesis, por todo el amor y la confianza que me ha dado. Gracias mi amor.*

*A todos los demás que siempre me ayudaron, mis tíos, mis primos y mis amigos lejanos.*

## Resumen:

El presente trabajo está enfocado a la implementación de los módulos Gestión de Reportes y Registro de Trasplantes del Sistema Alas NefroRed, para establecer un control estadístico de los datos de los pacientes y que la gestión de la información de los trasplantes que se realicen en cada uno de los 47 servicios nefrológicos, sea la necesaria.

Para la realización del trabajo se tuvo en cuenta la arquitectura definida por el Ministerio de Salud Pública (MINSAP) y el Ministerio de informática y comunicaciones (MIC) para el desarrollo las aplicaciones van a desplegadas en Infomed. Como estilos arquitectónicos de la aplicación se siguen el patrón Modelo Vista Controlador y Modelo en Tres Capas, además la aplicación se basa en Componentes y se realiza un consumo de servicios del Sistema de Información para la Salud (SISalud). Las Tecnologías y Herramientas que se utilizaron son como: servidor Web Apache 2.0, como lenguaje de programación PHP en su versión 5; como gestor de bases de datos MySQL en la versión 5. Para la elaboración de la aplicación, se utiliza el IDE de programación Zend Studio y los Frameworks de clases de PHP CodeIgniter y de Javascript YUI.

La aplicación facilita el proceso de Gestión de Reportes y Registro de Trasplante, obtendrá el control total de la información sobre los pacientes, y contribuirá a que el resultado final sea preciso y con la calidad requerida. Aumenta la capacidad organizativa de los Servicios de Diálisis del país y ayuda al trabajo de los médicos.

**Palabras Claves:** Aplicación, Sistema Informático, Gestión de Información, Nefrología, Enfermedad Renal Crónica, Diálisis, Gestión de Reportes, Trasplantes.

# Tabla de contenido

## LISTA DE TABLAS

## LISTA DE FIGURAS

## Introducción

En la actualidad se enfrentan grandes desafíos para lograr un mayor desarrollo de las Tecnologías de la Informática y las Comunicaciones (TIC), es importante destacar que el uso de las TIC favorece el trabajo individual y grupal. La experiencia demuestra que los medios informáticos permiten intercambiar ideas, discutir y decidir, en común, situaciones complejas; en el país están dirigidas fundamentalmente a la educación y la salud, dos de las esferas más importantes de nuestra sociedad.

Para lograr un desarrollo tecnológico avanzado en estas esferas, las estrategias de informatización deben centrarse en los problemas que día a día afectan a la sociedad en el mundo, de grandes cambios y transformaciones, logrando así elevar la calidad de vida de los ciudadanos.

Desde el triunfo de la Revolución, se han dado grandes pasos para mejorar la calidad de vida de la población cubana. En sus inicios se aplicaron técnicas de computación en el área de salud, pero no fue hasta finales de 1960 y principios de 1970 cuando se logró aumentar su uso y extenderlas a casi todo nuestro país.

Los amplios servicios de salud pública que se brindan gratuitamente a toda la población cubana, se consolidan, encaminados a vencer los nuevos retos de calidad y asistencia, con la introducción de modernas tecnologías.

Cuba cuenta con diversos centros nacionales especializados en distintas ramas de la salud, a través de la red INFOMED podemos acceder a cada unos ellos, con el objetivo de informarse sobre enfoques médicos novedosos y lo que es mejor, interactuar directamente con los documentos normativos del Sistema Nacional de Salud y posibilitar el acceso a los directorios de instituciones de salud de Cuba y el de Unidades de la Red de Información de Ciencias Médicas, así se logra una reducción de costos y aumenta la calidad del cuidado de los pacientes. En INFOMED además se trabaja para el establecimiento de redes nacionales de Bancos de Sangre, Nefrología, e Imágenes Médicas. (1)

Con el uso de la informática, el médico podrá ser liberado de tareas repetitivas y tediosas, pasará a ser el eje sobre el que se apoye todo el soporte informático, al actuar como selector y controlador de los datos de entrada, creador de los protocolos y algoritmos que éste seguirá, interpretando, además, sus resultados y por lo tanto siendo el máximo responsable del sistema.

# INTRODUCCION

Una rama de la salud priorizada en Cuba es la Nefrología, debido al incremento que ha tenido en los últimos años el número de personas que padecen Enfermedad Renal Crónica. La atención a los pacientes con esta enfermedad se hace cada vez más importante, se han creado hasta estos momentos, nuevas unidades de diálisis, y se han adquirido modernas tecnologías, que pueden ayudar a mejorar la vida de estos pacientes.

Desde el comienzo de la Revolución, se han venido desarrollando una serie de acciones que demuestran el deseo de un progreso continuo en esta rama. En 1961, se realizó la primera hemodiálisis en un enfermo portador de un fallo renal agudo. En 1966, se creó el Instituto de Nefrología, cuna de la especialidad, centro de referencia, coordinador de la actividad y asesor para el MINSAP actualmente. En 1968, se inició el tratamiento por diálisis, un año después se extendía a 4 territorios del país. En 1970, se generó el primer Programa Nacional de Nefrología para integrar el manejo de los pacientes. La práctica de la diálisis, el trasplante renal y el papel rector del Instituto, requirió desde sus inicios, el desarrollo de una comunicación permanente de los Centros que se fueron creando, comenzando a principios de los 80, de modo esporádico, la distribución nacional de los órganos para el trasplante renal.

Estos trasplantes están basados en los principios de la compatibilidad biológica a partir del Sistema Histocompatibilidad Humano (HLA); teniendo como presupuesto “el mejor riñón donado para el mejor receptor del país”. En 1989 se fundó el Centro Nacional Coordinador de Trasplante con un programa de selección automatizado para la selección de la pareja donante-receptor, el cual ha permitido hasta hoy el intercambio de órganos entre todos los territorios y la posibilidad de acceder a la diálisis y el trasplante renal con máxima justicia y equidad a los miles de enfermos dializados y trasplantados. (2)

Durante estos años, el trabajo se ha fortalecido para lograr grandes transformaciones y mejoras continúas en el proceso de atención al paciente, además se han abierto nuevos servicios con actividades territoriales, que siempre están vinculados con el Instituto. A pesar de todos los problemas por lo que ha tenido que pasar y está pasando el pueblo cubano, se optó por priorizar esta especialidad, construyéndose nuevos centros de Nefrología y rehabilitando otros.

# INTRODUCCION

Diariamente en estos centros bases se recogen de forma manual, estadísticas de las diálisis realizadas a los pacientes que son enviadas al INEF mensualmente por medio de correo electrónico o vía telefónica, lo que provoca errores humanos o algunos inconvenientes al no obtenerse la información de estos pacientes en tiempo real, ni con la exactitud requerida.

En ocasiones no se tiene el control de algunos pacientes que son trasplantados en el país, no se conoce el número real de trasplantes que se realizan, ni las estadísticas exactas de los trasplantes exitosos y los que no lo son. Todo esto sucede básicamente porque no existe un único mecanismo de control de esta información en los servicios nefrológicos del país y cuando se lleva algún control se tiene archivado en papel y esto trae consigo que el proceso de búsqueda de esta información sea engorroso e incluso que dicha información pueda perderse por deterioro de los documentos.

Además no se lleva el control sistemático del estado de aptitud de los pacientes que van a ser trasplantados lo que puede provocar que un órgano listo para ser utilizado se pierda porque el paciente al que le corresponde el órgano según la lista de espera de trasplantes no esté apto en ese momento.

Debido a la situación expuesta anteriormente se plantea el siguiente problema a resolver**:** ¿Cómo obtener un sistema funcional que facilite el proceso de gestión de estadísticas y trasplantes de los servicios nefrológicos del país? Este problema se enmarca en el Objeto de Estudio**:** El proceso de gestión de las estadísticas y trasplantes. De aquí se deriva que el campo de acción**:** Proceso de desarrollo de software para la gestión de las estadísticas y trasplantes de los servicios nefrológicos de Cuba.

Para dar solución a la problemática antes mencionada se ha definido como objetivo general**:** Implementar un sistema informático que facilite el proceso de gestión de estadísticas y trasplantes que se realizan en los servicios nefrológicos del país.

Este sistema permitirá una mayor organización y eficiencia a la hora de atender a los pacientes en los centros de diálisis del país. Asimismo, los médicos podrán trabajar de forma rápida con los datos e informaciones necesarias para cada consulta y tener un control total de la descripción de cada trasplante realizado en sus pacientes, y dichos trasplantes se realizarán de forma más rápida y ordenada. Con el uso del sistema la calidad de la gestión de la información estadística será efectiva, permitiendo realizar una comparación a nivel municipal, provincial y nacional.

# INTRODUCCION

Dentro de las tareas de la investigación que se proponen para dar solución a los objetivos planteados están:

1. Valorar las técnicas de programación, lenguaje, plataforma y librerías propuestas por el cliente y la universidad.
2. Valorar la arquitectura de los sistemas que están en los servidores de Infomed.
3. Analizar los componentes del sistema Alas NefroRed que fueron desarrollados en el curso 2007-2008.
4. Valorar la arquitectura propuesta en el Análisis y Diseño del sistema.
5. Realizar la implementación de los componentes utilizando los patrones de diseño establecidos.
6. Obtener Modelo de Implementación y los artefactos necesarios que describan la Base de Datos.

El presente trabajo consta de tres capítulos que son descritos a continuación:

**Capítulo I Fundamentos teóricos:** En este capítulo se analiza el estado del arte de las diferentes técnicas de programación que existen a nivel internacional, nacional y de la Universidad. Las tendencias, tecnologías y metodologías relacionadas con dichas técnicas, así como las plataformas de desarrollo que la soportan. Además se realiza un estudio crítico y valorativo de las técnicas de programación, plataformas, librerías usadas en el desarrollo del proyecto y se explican los patrones de arquitectura utilizados.

**Capítulo II Elementos de la arquitectura:** En esta sección se explican y seleccionan los requerimientos no funcionales, se describe la arquitectura propuesta, conjuntamente se analizan las posibles implementaciones, componentes o módulos ya existentes que puedan ser rehusados. También se puede encontrar cómo se controla la seguridad en la aplicación, el Diagrama de Despliegue y las estrategias de codificación, estándares y estilos a utilizar.

**Capítulo III Descripción y análisis de la solución propuesta:** En este apartado se hará una descripción y análisis de las nuevas clases u operaciones desarrolladas en la aplicación y de las tablas que conforman el modelo de datos, de los módulo Gestión de Reportes y Registro de trasplantes se valorará el diseño propuesto por el analista y se mostrará la vista de implementación con sus respectivos diagramas de componentes.

## Capitulo I: Fundamentación Teórica

En el presente capitulo se hace un análisis para valorar el estado del arte de las diferentes técnicas de programación que existen a nivel internacional, nacional y de la Universidad. Las tendencias, técnicas, tecnologías, así como las plataformas de desarrollo que la soportan. Conjuntamente se realiza un estudio crítico y valorativo de dichas técnicas de programación, plataformas, librerías usadas en el desarrollo del proyecto y se explican los patrones de arquitectura utilizados.

### 1.1 Objeto de estudio. Descripción general

El proceso de gestión de las estadísticas y trasplantes de los servicios nefrológicos del país, constituye el objeto de estudio de este trabajo, a través del cual se desea desarrollar una solución para los problemas existentes en el control de las estadísticas y trasplantes en el país. Desde el comienzo de la Revolución, se han venido desarrollando una serie de acciones que demuestran el deseo de una mejora continua para el pueblo de Cuba.

En 1966, se creó el Instituto de Nefrología "Dr. Abelardo Buch López", cuna de la especialidad, centro de referencia, coordinador de la actividad y asesor para el MINSAP. El centro está adjunto al Hospital General Universitario "Joaquín Albarrán". En el Instituto de Nefrología se encuentra el Centro Coordinador Nacional de Enfermedad Renal Crónica, Diálisis y Trasplante Renal. En el país, existen actualmente 47 unidades de diálisis de las cuales 9 son además Centros de Trasplantes, en Santa Clara, Camagüey, Holguín, Santiago de Cuba y 5 en Ciudad Habana, estas últimas incluyen Laboratorios de Tipaje Tisular, así como otros de apoyo a esta actividad.

La Enfermedad Renal Crónica es uno de los problemas de salud mas importantes del siglo XXI, que puede ser provocada por diabetes, bajo peso al nacer, consumo de drogas e historia familiar, los pacientes con ERC en su etapa terminal requieren tratamiento de sustitución renal por diálisis o trasplante renal. El trasplante renal puede ser una de las mejores soluciones, pero no siempre sus resultados son positivos: los pacientes que inician el tratamiento para realizarse el trasplante en muchos casos pueden rechazar el trasplante, estando obligados a volver al programa de diálisis, o pueden morir. Para los especialistas en Nefrología, es muy significativo evitar que el paciente llegue a la Enfermedad Renal Crónica.

## 1.2 Tendencias y tecnologías actuales

### 1.2.1 ¿Qué es Internet?

Internet surgió en 1969, en el departamento de defensa de los EEUU como un experimento para unir en red a los ordenadores de los centros de investigación tanto a nivel militar como educacional. Se dio origen entonces a una red informática llamada ARPANET la cual no se debilitaría y permitiría que no se perdiera toda la información en el caso que hubiera un ataque nuclear. Más adelante se utilizó para el gobierno, investigación académica y comercial y para comunicaciones

Para ello se creó un "idioma" que hablaran los ordenadores para comunicarse entre sí. Este fue el origen del protocolo TCP/IP, actualmente empleado en Internet. Posteriormente se fueron incorporando las diferentes universidades norteamericanas y la red comenzó a crecer dando lugar en 1983 a Internet, la red que posiblemente haya alcanzado mayor popularidad a nivel mundial. (3)

En la actualidad la red Internet ofrece un gran número de servicios fundamentales a los internautas. Entre ellos destacan el correo electrónico, que permite mandar mensajes instantáneos a cualquier parte del mundo; la Web, la parte más amplia de la red en la que se alojan un gran número de páginas de múltiples contenidos; el FTP, un sistema de transferencia de ficheros y los grupos de noticias, que se componen de foros en los que se puede participar y recibir mucha información. (4)

### 1.2.2 Aplicaciones Web. Ventajas y Desventajas.

En la ingeniería de software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador. En otras palabras, es una aplicación software que se codifica en un lenguaje soportado por los navegadores web (HTML, JavaScript, Java, etc.) en la que se confía la ejecución al navegador.

## Ventajas:

- No requieren instalación, pues usan tecnología Web, lo cual permite el aprovechamiento de todas las características del Internet.
- Son fáciles de usar (No se requieren complicadas combinaciones de Hardware/Software para utilizar estas aplicaciones). Solo un computador con un buen navegador web.
- Alta disponibilidad, ya que puede realizar consultas en cualquier parte del mundo donde tenga acceso a Internet y a cualquier hora.
- Una empresa puede migrar de sistema operativo o cambiar el Hardware libremente sin afectar el funcionamiento de las aplicaciones de servidor.
- Se facilita el trabajo a distancia. Se puede trabajar desde cualquier PC o computador portátil con conexión Internet.
- Actualizar o hacer cambios en el Software es sencillo y sin riesgos de incompatibilidades. Existe solo una versión en el servidor lo que implica que no hay que distribuirla entre los demás computadores. El proceso es rápido y limpio. Al funcionar en un navegador, se requiere un conocimiento básico de informática para utilizar una aplicación web. (5)

## Desventajas:

- Acceso limitado, la necesidad de conexión permanente y rápida o intranet hacen que el acceso a estas aplicaciones no esté al alcance de todos.

Diferencias de presentación entre plataformas y navegadores. La falta de estándares ampliamente soportados dificulta el desarrollo de las aplicaciones. (6)

### 1.2.3 Lenguajes de programación Web

#### PHP

Es uno de los lenguajes Open Source multiplataforma y libre que logra trabajar con la mayoría de las bases de datos actuales (especialmente MySQL) y es compatible con todos los navegadores actuales. Es un lenguaje de programación muy potente que, junto con HTML, permite crear sitios web dinámicos. PHP se instala en el servidor y funciona con versiones de Apache, Microsoft IIS, Netscape Enterprise Server y otros. La forma de usar PHP es insertando código PHP dentro del código HTML de un sitio web. Permite la conexión a numerosas bases de datos, incluyendo MySQL, Oracle, ODBC, etc. Y puede ser ejecutado en la mayoría de los sistemas operativos (Windows, Mac OS, Linux, Unix).

A diferencia de otros lenguajes de programación que se ejecutan en el navegador. Su ejecución, es en el servidor. Los desarrolladores PHP hoy tienen un futuro muy prometedor y un mercado de trabajo internacional. (7)

#### Ventajas:

- PHP corre en (casi) cualquier plataforma utilizando el mismo código fuente, pudiendo ser compilado y ejecutado en algo así como 25 plataformas, incluyendo diferentes versiones de Unix, Windows y Macs. Como en todos los sistemas se utiliza el mismo código base, los scripts pueden ser ejecutados de manera independiente al sistema operativo.
- La sintaxis de PHP es similar a la del C, por esto cualquiera con experiencia en lenguajes del estilo C podrá entender rápidamente PHP.
- PHP es completamente expandible. Está compuesto de un sistema principal (escrito por Zend), un conjunto de módulos y una variedad de extensiones de código.
- Muchas interfaces distintas para cada tipo de servidor. PHP actualmente se puede ejecutar bajo Apache, IIS, AOLServer, Roxen y THTTPD. Otra alternativa es configurarlo como modulo CGI.

- Puede interactuar con muchos motores de bases de datos tales como MySQL, MS SQL, Oracle, Informix, PostgreSQL, y otros muchos. Siempre podrás disponer de ODBC para situaciones que lo requieran.
- Una gran variedad de módulos, cuando un programador PHP necesite una interfaz para una librería en particular, fácilmente podrá crear una API para esta. Algunas de las que ya vienen implementadas permiten manejo de gráficos, archivos PDF, Flash, Cybercash, calendarios, XML, IMAP, POP, etc.
- Rapidez. PHP generalmente es utilizado como módulo de Apache, lo que lo hace extremadamente veloz. Esta completamente escrito en C, así que se ejecuta rápidamente utilizando poca memoria.
- PHP es Open Source y libre, lo cual significa que el usuario no depende de una compañía específica para arreglar cosas que no funcionan, además no está forzado a pagar actualizaciones anuales para tener una versión que funcione. (8)

## Desventajas:

- Los programadores trabajan de formas diferentes, esto hace que sea muy difícil mantener el código de terceras personas.
- Existen muchas versiones de PHP con incompatibilidades entre sí.
- En PHP 4 es difícil depurar los errores.
- El manejo de errores no es tan sofisticado como Cold Fusion o ASP. (9)

## HTML

Lenguaje para la creación de páginas web, utilizado para el formato de documentos de hipertextos. Los documentos HTML no son documentos de texto normal, sino documentos de hipertextos ya que en el propio documento aparecen enlaces a otros. Es un estándar reconocido a nivel mundial, una página HTML se visualiza de forma muy similar en cualquier navegador en diferentes sistemas operativos. Ha pasado de ser un lenguaje utilizado exclusivamente para crear documentos electrónicos a ser un lenguaje

que se utiliza en muchas aplicaciones electrónicas como: buscadores, tiendas online y bancas electrónicas. (10)

### Características de HTML

Cada elemento de un documento HTML consta de una etiqueta de comienzo, un bloque de texto y una etiqueta de fin con el siguiente formato:<etiqueta>bloque de texto</etiqueta>.Estos elementos se denominan contenedores, porque contienen un bloque de texto entre las etiquetas de comienzo y fin. También existen elementos vacíos, que no tienen contenido y únicamente tienen una etiqueta de inicio <etiqueta>.Muchos elementos tienen atributos que definen propiedades del elemento:
<etiqueta atributo="valor"> bloque de texto </etiqueta>

HTML no distingue entre mayúsculas y minúsculas. Cuando es importante hacerlo, como al poner un título o un atributo, hay que ponerlo entre comillas dobles como en <a href=”index.aspx” title=”Hooping”>.Los comentarios se escriben en HTML empezando con <!-- y terminando con -- > .Los espacios, tabulaciones, líneas en blanco y retornos del documento HTML se ignoran, tomándose como un único espacio en blanco. Esto permite añadir espacios para aumentar la claridad del documento.

Los caracteres con acentos o la ñ se escriben poniendo el carácter ampersand (&) seguido del nombre asociado al carácter y un punto y coma (;).

HTML tiene unas reglas estructurales que indican dónde pueden y no pueden ir los elementos. Se tiene el ejemplo de los titulares, que son independientes entre ellos, no pudiendo contenerse unos a otros. Las etiquetas tienen que seguir un orden piramidal, las primeras que se abren son las últimas que se cierran. (11)

## Javascript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Técnicamente, es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se

pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java. Legalmente, JavaScript es una marca registrada de la empresa Sun Microsystems. Es un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado. Todos los navegadores interpretan el código Javascript integrado dentro de las páginas web. La ventaja que presenta sobre el HTML es que permite crear páginas Web más dinámicas, lo que las hace más atractivas para el usuario. (12)

## JSP

JSP es un acrónimo de Java Server Pages. Es una tecnología orientada a introducir páginas web con programación en Java. Con JSP se pueden implantar aplicaciones web que se ejecuten en variados servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintaxis Java. Por tanto, las JSP se escriben con un editor HTML/XML habitual. El motor de las páginas JSP está basado en los servlets de Java ,programas en Java destinados a ejecutarse en el servidor, aunque el número de desarrolladores que pueden afrontar la programación de JSP es mucho mayor, dado que resulta mucho más sencillo aprender que los servlets.

En JSP se establecen páginas de manera parecida a como se crean en ASP o PHP otras dos tecnologías de servidor. Se crean archivos con extensión .jsp que incluyen, dentro de la estructura de etiquetas HTML, las sentencias Java a ejecutar en el servidor. Antes de que sean funcionales los archivos, el motor JSP lleva a cabo una fase de traducción de esa página en un servlet, implementado en un archivo class (Byte codes de Java). Esta fase de traducción se lleva a cabo habitualmente cuando se adopta la primera solicitud de la página .jsp, aunque existe la opción de precompilar en código para evitar ese tiempo de espera la primera vez que un cliente solicita la página. (13)

## Lenguaje de programación escogido para desarrollar el trabajo

Luego de hacer el análisis entre las distintas técnicas de programación, se decide utilizar el PHP por las siguientes características:

- Está soportado en la mayoría de las plataformas de Sistemas Operativos, mientras que por ejemplo ASP por ser propiedad de Microsoft no es multiplataforma.
- El PHP no tiene costo oculto, o sea que cuando se adquiere incluye un sin número de bibliotecas que proporcionan el soporte para la mayoría de las aplicaciones Web, por ejemplo e-mail, generación de ficheros PDF y otros. En caso de que no se tengan las bibliotecas estas se pueden encontrar gratis en Internet. En el caso de ASP forma parte del Internet Information Server que viene integrado en Windows NT-2000 Server con su elevado costo de adquisición.
- Algunos son parecidos en cuanto a la forma de utilización, pero PHP es más rápido, gratuito y multiplataforma.

Por todas estas características y ventajas previamente descritas, por políticas del Área Temática donde el trabajo está inmerso, y además porque el sistema se desplegará en los servidores de INFOMED, para la cual existe una arquitectura definida que incluye la utilización del lenguaje programación PHP para el desarrollo de las aplicaciones web, son las razones por las cuales se escoge el lenguaje de programación en cuestión.

### 1.2.4 Sistemas Gestores de Bases de Datos (SGBD)

Un Sistema Gestor de base de datos (SGBD) es un conjunto de programas que permiten crear y mantener una Base de datos, asegurando su integridad, confidencialidad y seguridad.

Por tanto debe permitir:

Especificar tipos, estructuras y restricciones de datos, construir la base de datos: guardar los datos en algún medio controlado por el mismo SGBD y manipular la base de datos: realizar consultas, actualizarla, generar informes. Ejemplo de SGBD son Oracle y SQL Server de Microsoft. (14)

Todo SGBD debe cumplir con los siguientes objetivos:

- Abstracción de la información.
- Redundancia mínima.
- Consistencia.
- Seguridad.
- Integridad.
- Tiempo de respuesta mínimo. (15)

### PostgreSQL

El Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos conocido como PostgreSQL está derivado del paquete Postgres escrito en Berkeley. Con cerca de una década de desarrollo tras él, PostgreSQL es el gestor de bases de datos de código abierto más avanzado en la actualidad, ofreciendo control de concurrencia multi-versión, soportando casi toda la sintaxis SQL (incluyendo subconsultas, transacciones, y tipos y funciones definidas por el usuario), contando también con un amplio conjunto de enlaces con lenguajes de programación (incluyendo C, C++, Java, Perl, tcl y Python).

Postgres ofrece una potencia adicional sustancial al incorporar los siguientes cuatro conceptos adicionales básicos, es una vía en la que los usuarios pueden extender fácilmente el sistema: clases, herencia, tipos y funciones. (16)

Otras características aportan potencia y flexibilidad adicional:

- Restricciones (Constraints)
- Disparadores (triggers)
- Reglas (rules)
- Integridad transaccional

Estas características colocan a Postgres en la categoría de las Bases de Datos identificadas como objeto-relacionales. (17)

Caracteristicas:

- Implementación del estándar SQL92/SQL99.
- Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes (MAC, IP...), cadenas de bits, etc. También permite la creación de tipos propios.
- Incorpora una estructura de datos array.
- Incorpora funciones de diversa índole: manejo de fechas, geométricas, orientadas a operaciones con redes, etc.
- Permite la declaración de funciones propias, así como la definición de disparadores.
- Soporta el uso de índices, reglas y vistas.
- Incluye herencia entre tablas (aunque no entre objetos, ya que no existen), por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.(18)

## SQL

Lenguaje de consulta estructurado (SQL), es un lenguaje de base de datos normalizado, utilizado por el motor de base de datos de Microsoft Jet. Se utiliza para crear objetos QueryDef, como el argumento de origen del método OpenRecordSet y como la propiedad RecordSource del control de datos. También se puede utilizar con el método Execute para crear y manipular directamente las bases de datos Jet y crear consultas de paso a través para manipular bases de datos remotas cliente - servidor. Está compuesto por comandos, cláusulas, operadores y funciones de agregado. Estos elementos se combinan en las instrucciones para crear, actualizar y manipular las bases de datos.

Existen dos tipos de comandos **SQL**:

- Los DLL que permiten crear y definir nuevas bases de datos, campos e índices.
- Los DML que permiten generar consultas para ordenar, filtrar y extraer datos de la base de datos.

Algunas de las características del SQL:

- Es una forma estándar de consulta de datos específicos.
- Es una forma de extraer y manipular datos de una base de datos.
- Usado para todas las funciones de bases de datos, incluyendo administración.
- Creación de esquemas y datos recuperables.
- Puede ser usado de forma implícita dentro de una aplicación. (19)

## MySQL

Sistema de gestión de bases de datos relacional, licenciado bajo la GPL de la GNU. Su diseño multihilo le permite soportar una gran carga de forma muy eficiente. MySQL fue creada por la empresa sueca MySQL AB, que mantiene el copyright del código fuente del servidor SQL, así como también de la marca. Aunque MySQL es software libre, MySQL AB distribuye una versión comercial de MySQL, que no se diferencia de la versión libre más que en el soporte técnico que se ofrece, y la posibilidad de integrar este gestor en un software propietario, ya que de no ser así, se vulneraría la licencia GPL.Este gestor de bases de datos es, probablemente, el gestor más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

Entre las características disponibles en las últimas versiones se puede destacar:

- Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multihilo.
- Soporta gran cantidad de tipos de datos para las columnas.
- Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, etc).
- Gran portabilidad entre sistemas.
- Soporta hasta 32 índices por tabla.
- Gestión de usuarios y passwords, manteniendo un muy buen nivel de seguridad en los datos.(20)

## Características de la versión 5.0:

- Tipo de datos Binarios: Soporte para almacenamiento de números de notación binaria, mucho más eficiente que el conocido tipo Boolean.
- Manejo y administración de cursores dentro de procedimientos almacenados.
- Introducción de INFORMATION_SCHEMA: un medio estándar para el acceso a la metadata del Servidor MySQL.
- Administrador en Instancia: Se puede iniciar y parar el servidor MySQL desde un cliente remoto.
- Precisión Matemática: Se añade una biblioteca para dar mayor precisión a las operaciones matemáticas con punto flotante.
- Procedimientos Almacenados (store procedures): Se han añadido soporte para procedimientos almacenados.
- Disparadores (Triggers): Se ha añadido soporte limitado para disparadores.
- Vistas (views): Se ha añadido soporte para vistas.
- Tamaño de Varchar: Se ha aumentado el tamaño del tipo varchar a 65,532 bytes, posee también mayor eficiencia que en anteriores versiones.
- Transacciones XA: Se ha añadido soporte para transacciones XA.
- Introduce un nuevo optimizador que puede reducir considerablemente el tiempo requerido para llegar un plan de la ejecución de la pregunta.

- Uso del método de la fusión, que permite el uso de múltiples índices en la recuperación de datos de la consulta.
- Optimizador de condiciones en las consultas.
- Sistema de integración referencial.
- Nueva forma de almacenamiento que disminuye el espacio de almacenamiento en 20%. (21)

## Sistema Gestor de Base de Datos escogido para desarrollar el trabajo

Luego de analizadas las características y facilidades de los SGBD presentados, y las de la herramienta a desarrollar se decide usar el MySQL en su versión 5.0 como SGBD, por todas las características mostradas anteriormente; además por las siguientes razones: PHP lo maneja más fácil que a cualquier otro sistema debido a la gran cantidad de funciones que tiene explícitas. Es multiplataforma y hasta su versión 5.0 no tiene precio en el mercado, se adquiere libremente. Igualmente de todas las ventajas que trae implícita como gestor de Base de Datos en sí, se definió por políticas del Área Temática, el uso de MySQL, debido que el sistema será desplegado en los servidores de INFOMED para la cual existe una arquitectura definida que rige la utilización de este Gestor.

## 1.2.5 IDE de programación

('IDE') Un entorno de desarrollo integrado, es un programa compuesto por un conjunto de herramientas para un programador. Los IDEs proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación. En algunos lenguajes, un IDE puede funcionar como un sistema en tiempo de ejecución, en donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto, como es el caso de Smalltalk, Zend Studio u Objective-C.Es posible que un mismo IDE pueda funcionar con varios lenguajes de programación. Este es el caso de Eclipse, que mediante pluggins se le puede añadir soporte de lenguajes adicionales.

Los componentes que deben tener los IDEs son:

- Un editor de texto.
- Un compilador.
- Un intérprete.
- Herramientas de automatización.
- Un depurador.
- Posibilidad de ofrecer un sistema de control de versiones.
- Factibilidad para ayudar en la construcción de interfaces gráficas de usuarios. (22)

## Zend Studio como IDE de programación

Se trata de un programa de la casa Zend, impulsores de la tecnología de servidor PHP, orientada a desarrollar aplicaciones web, en lenguaje PHP. El programa, además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código. El programa entero está escrito en Java, lo que a veces supone que no funcione tan rápido como otras aplicaciones de uso diario. Sin embargo, esto ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para Windows, Linux y MacOS, aunque el desarrollo de las versiones de este último sistema se retrase un poco más. Zend Studio consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Las dos partes se instalan por separado, la del cliente contiene el interfaz de edición y la ayuda.

Permite también hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración habrá que disponer de la parte del servidor, que instala Apache y el módulo PHP, en caso de que estén instalados, los configura para trabajar juntos en depuración. (23)

## 1.2.6 XML / WebServices

## Web Services

Los Web Services permiten la comunicación entre aplicaciones o componentes de aplicaciones de forma estándar a través de protocolos comunes (como http) y de manera independiente al lenguaje de programación, plataforma de implantación, formato de presentación o sistema operativo. Un Web Service es un contenedor que encapsula funciones específicas y hace que estas funciones puedan ser utilizadas en otros servidores. Estas son algunas ventajas que presentan los Web Services:

- Son programables.
- Están basados en XML, que es un lenguaje abierto.
- Son autodescriptivos.
- Pueden buscar registros de otros Web Services.

## Beneficios de Web Services sobre otras tecnologías

Los Web Services presentan algunas diferencias sobre sistemas distribuidos tradicionales, tales como EJB, COM/DCOM, CORBA, SOM/DSOM o DCE. Algunas de las diferencias son las siguientes:

- Escaso acoplamiento. El cliente no necesita conocer nada acerca de la implementación del servicio al que está accediendo, salvo la definición WSDL.
- Independencia del lenguaje de programación. El servidor y el cliente no necesitan estar escritos en el mismo lenguaje.
- Independencia del modo de transporte. SOAP puede funcionar sobre múltiples protocolos de transporte, como por ejemplo HTTP, HTTPS, HTTP-R, BEEP, JABBER, IIOP, SMTP o FTP.
- Múltiples modos de invocación. Los servicios web soportan tanto invocación estática como invocación dinámica.
- Múltiples estilos de comunicación. Los servicios web soportan tanto comunicación síncrona (RPC) como comunicación asíncrona (mensajería). (24)

Entonces puede decir que un Web Service es una comunicación por medio de mensajes SOAP entre diferentes equipos a través de una red. Para conocer cómo se realiza el intercambio de mensajes en los Web Services primero se debe saber cuales son los elementos fundamentales que lo componen. Estos son el XML, SOAP, WSDL, y UDDI.

Web Services Protocol Stack, se denomina así al conjunto de servicios y protocolos, de los servicios Web:

**XML** (Extensible Markup Language): Es el formato estándar para los datos que se vayan a intercambiar.

**SOAP** (Simple Object Access Protocol): Protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

**WSDL** (Web Services Description Languages): Es el lenguaje de la interfaz pública para los servicios Web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios Web.

**UDDI** (Universal Description, Discovery and Integration): Protocolo para publicar la información de los servicios Web. Permite comprobar qué servicios web están disponibles.

**TCP/IP:** Es uno de los protocolos fundamentales para la comunicación, está diseñado para enrutar y tiene un grado muy elevado de fiabilidad, es adecuado para redes grandes y medianas, así como en redes empresariales. Se utiliza a nivel mundial para conectarse a Internet y a los servidores web. Es compatible con las herramientas estándar para analizar el funcionamiento de la red.

**HTTP:** Es el protocolo usado en cada transacción de la Web, define la sintaxis y la semántica que utilizan los elementos software de la arquitectura web (clientes, servidores, proxies) para comunicarse. Es orientado a transacciones y sigue el esquema petición-respuesta entre un cliente y un servidor. Además es un protocolo sin estado, es decir, que no guarda ninguna información sobre conexiones anteriores.

**WS-Security** (Web Services Security): Protocolo de seguridad aceptado como estándar por OASIS (Organization for the Advancement of Structured Information Standards). Garantiza la autenticación de los actores y la confidencialidad de los mensajes enviados Otros protocolos: los datos en XML también pueden enviarse de una aplicación a otra mediante protocolos normales como HTTP (Hypertext Transfer Protocol), FTP (File Transfer Protocol), o SMTP (Simple Mail Transfer Protocol). (25)

## Apache

Servidor web flexible, rápido y eficiente, continuamente actualizado y adaptado a los nuevos protocolos (HTTP 1.1).

Entre sus características destacan:

- Multiplataforma
- Es un servidor de web conforme al protocolo HTTP/1.1
- Modular: Puede ser adaptado a diferentes entornos y necesidades, con los diferentes módulos de apoyo que proporciona, y con la API de programación de módulos, para el desarrollo de módulos específicos.
- Basado en hebras en la versión 2.0

- Incentiva la realimentación de los usuarios, obteniendo nuevas ideas, informes de fallos y parches para la solución de los mismos.
- Se desarrolla de forma abierta
- Extensible: gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor.

Apache presenta, entre otras características, mensajes de error altamente configurables, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración. Apache tiene amplia aceptación en la red, desde 1996, es el servidor HTTP más usado. Alcanzó su máxima cuota de mercado en 2005 siendo el servidor empleado en el 70% de los sitios web en el mundo, sin embargo ha sufrido un descenso en los últimos años. La mayoría de las vulnerabilidades de la seguridad descubiertas y resueltas tan sólo pueden ser aprovechadas por usuarios locales y no remotamente. (26)

### Servidor escogido para el desarrollo del trabajo

El servidor utilizado es el Apache pues está establecido en la línea base de la arquitectura de la facultad 7. Este tiene gran popularidad, es reconocido en muchos ámbitos empresariales, tecnológicos y también por las siguientes razones: corre en una multitud de Sistemas Operativos, es una tecnología gratuita de código fuente abierto, es un servidor altamente configurable de diseño modular, trabaja con gran cantidad de Perl, PHP y otros lenguajes de script.

### 1.2.7 Framework

Conjunto de librerías que obligan a trabajar en un patrón de diseño determinado. Sirven para crear aplicaciones, ya que por sí mismas no tienen ninguna funcionalidad hasta que se desarrolla una aplicación. Los Frameworks simplifican el desarrollo de las aplicaciones mediante la automatización de muchas de las tareas comunes. Además proporciona estructura al código fuente, forzando al programador a crear código más legible y más fácil de mantener. (27)

## Framework Symfony

Symfony es un completo Frameworks diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Symfony es un enorme conjunto de herramientas y utilidades que simplifican el desarrollo de las aplicaciones web. Este emplea el tradicional patrón de diseño MVC (modelo-vista-controlador) para separar las distintas partes que forman una aplicación web. El modelo representa la información con la que trabaja la aplicación y se encarga de acceder a los datos. (28)

Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio electrónico de primer nivel. (29)

## Características de Symfony

Symfony se diseñó para que se ajustara a los siguientes requisitos:

- Fácil de instalar y configurar en la mayoría de plataformas (y con la garantía de que funciona correctamente en los sistemas Windows y *nix estándares).
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.

- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros .(30)

## Framework CodeIgniter

Poderoso Framework para PHP que facilita la escritura de código repetitivo, y en comparación con otros Frameworks como CakePHP, Symphony o Zend Framework, CodeIgniter es más rápido y más fácil. Es totalmente extensible y altamente compatible con gran variedad de versiones y configuraciones de PHP.Entre sus características se puede encontrar su compatibilidad con PHP 4 y PHP 5, está basado en el Modelo Vista Controlador, posee soporte para múltiples bases de datos entre las que se pueden mencionar PostgreSQL, MySQL, MSSQL; se le pueden incluir drivers para otros gestores de bases de datos, incluye también plantillas, validaciones, no requiere instalación y es una librería con un gran número de clases.

Una de las características más interesantes de CodeIgniter es el elevado número de clases que incluye para trabajar con distintos objetos: calendario, bases de datos, correo electrónico, manipulación de imágenes, FTP, lenguaje, tablas, sesiones, compresión ZIP, etcétera. En contraste de otros Frameworks, CodeIgniter cuenta con una documentación excelente que permite conocer todos los secretos de este entorno de trabajo. Está liberado bajo la licencia de código abierto Apache-BSD, lo que significa que es totalmente libre y puede ser usado. (31)

## Yahoo user Interface (YUI)

Yahoo user Interface (YUI) Library es un conjunto de utilidades y controles, escrito en Java Script para desarrollar aplicaciones Web interactivas usando técnicas DOM, DHTML y AJAX.YUI incluye también un conjunto de fuentes CSS .Todos los componentes YUI han sido desarrollados bajo Licencia BSD Open Source y están disponibles para todo tipo de uso.

**Características:**

Están disponibles dos tipos de componentes diferentes: Utilidades y Controles: Las YUI Utilidades simplifican el desarrollo para la compatibilidad entre Navegadores basados en técnicas DOM, DHTML y AJAX.Los controles de YUI proporcionan elementos visuales altamente interactivos del diseño para sus aplicaciones Web. Estos elementos se crean y se manejan íntegramente del lado del cliente (usuario) y nunca requieren de una recarga de página.

Utilidades disponibles:

- **Animación:** Para crear "efectos cinemáticos", animando la posición, el tamaño, la opacidad u otras características de los elementos de la página. Estos efectos se pueden utilizar para reforzar la comprensión del usuario en los cambios que suceden en la página.
- **Administrador de Historial**: Los desarrolladores de aplicaciones Web interactivas requerían que los marcadores (bookmarks) no solo apunten a las páginas sino también al estado de las mismas, así como también el botón de volver atrás del navegador opere de acuerdo a la aplicación. El Administrador de Historial provee la funcionalidad para la administración de marcadores (bookmarks) y del botón de volver atrás del navegador. Esta utilidad se encuentra en etapa experimental.
- **Administrador de Conexiones:** Esta librería ayuda a manejar conexiones XMLHttpRequest (Comúnmente conocido como AJAX) en el navegador incluyendo soporte para uploads, envío de formularios por método POST y gestión de errores.
- **Utilidad de Fuente de datos:** DataSource proporciona una interfaz para recuperar datos desde arrays, servicios de XHR (XMLHttpRequest), funciones personalizadas con cache integrado y soporte de Administrador de Conexiones.
- **Utilidades DOM**: La utilidad DOM contiene una variedad de métodos basados en la estructura DOM para permitir compatibilidad entre navegadores, incluyendo posicionamiento de elementos y administración de CSS
- **Arrastrar y Soltar:** Crea objetos que se pueden ser tomados y soltados en cualquier parte de la página. Escribes el código para los "momentos interesantes" que son disparados a lo largo de la interacción, por ejemplo cuando un objeto pasa por encima un área determinada de la página; esta utilidad se encarga de mantener funcionando suavemente en todos los navegadores.

- **Eventos**: Esta utilidad facilita el manejo de eventos.

Controles disponibles:

- **Autocompletado**: El control de Autocompletado permite crear una vía de interacción con el usuario donde estén involucradas las entras de texto (formularios); este control provee listas de sugerencias y funcionalidades de type-ahead (tipear más rápido) basadas en una variedad de datos y formatos de diferentes fuentes. (ej. XMLHttpRequest).
- **Botón de control Calendario**: Este control es un calendario grafico con selección dinámica de fechas.
- **Contenedor**: Este control soporte diferentes patrones DHTML Windowing (diferentes maneras de mostrar información en pantalla) con ToolTip, Panel, Dialog y SimpleDialog. El modulo proporciona una plataforma para implementaciones de patrones DHTML adicionales y configurables
- **Registros de Eventos**: El YUI Logger proporciona una manera rápida y fácil de generar los registros de eventos para ser mostrados en pantalla o consola. (Ej. Firebug para Firefox). Con este control podrás detectar y corregir errores en tus aplicaciones Web.
- **Menú**: Para crear minués en el instante con pocas líneas de código.(32)

## Framework escogido para desarrollar el trabajo

Luego de analizadas las características de los Framework presentados se decide hacer uso del Framework CodeIgniter, ya que es un poderoso Framework para PHP que facilita la escritura de código repetitivo, y en comparación de otros Frameworks cómo CakePHP, Symphony o Zend Framework, CodeIgniter es más rápido y mas fácil, es totalmente extensible y altamente compatible con gran variedad de versiones y configuraciones de PHP.Además es compatible con PHP 5 y está basado en el Modelo Vista Controlador, posee soporte para múltiples bases de datos entre las que se pueden mencionar PostgreSQL, MySQL, MSSQL.

Se utiliza además Yahoo User Interface, las Utilidades de YUI simplifican el desarrollo para la compatibilidad entre Navegadores, y los controles proporcionan elementos visuales altamente interactivos del diseño para sus aplicaciones Web, utiliza Prototype es un Frameworks basado en JavaScript que se orienta al desarrollo sencillo y dinámico de aplicaciones Web.

## 1.3 Tecnología y herramientas a utilizar

El sistema basa su arquitectura en una ya elaborada por el grupo de arquitectura MINSAP-MIC para el desarrollo de todas las aplicaciones para Infomed. Esta arquitectura está basada en componentes siguiendo el patrón de aplicaciones en tres capas, además se utiliza el patrón Modelo-Vista-Controlador y haciendo un consumo de servicios de SISALUD. De la misma forma se definen varias de las herramientas que se deben utilizar, servidor Web Apache 2.0, el cual es libre de código abierto y bajo la licencia Apache/GPL cuya descripción y aplicaciones ya fue dada anteriormente.

Como lenguaje de programación esta arquitectura define PHP en su versión 5.0, como gestor de bases de datos se usará el ya conocido y libre MySQL en la versión 5.0. Para la elaboración de la aplicación, se usará el IDE de programación Zend Studio por las grandes prestaciones que este brinda. El sistema además está implementado usando el Frameworks de clases de PHP CodeIgniter, y de Javascript YUI, se hace rehúso de todas las librerías y clases que estos dos potentes Frameworks ofrecen.

## 1.4 Patrones o estilos arquitectónicos presentes en la solución. Justificación de su uso y características

Los patrones del diseño tratan los problemas que se repiten y se presentan en situaciones particulares a la hora de diseñar la aplicación, con el fin de proponer soluciones a ellas. Por tanto estos son soluciones exitosas a problemas comunes. Existen muchas ideas para implementarlos, los detalles son llamados estrategias. Hay patrones que abarcan las distintas etapas del desarrollo, desde el análisis hasta el diseño y desde la arquitectura hasta la implementación.

## Modelo Tres Capas


La arquitectura en capas es la generalización de la arquitectura Cliente-Servidor, donde la carga se divide en tres partes con un reparto claro de funciones: una Capa de Presentación, otra parte para las reglas lógicas del negocio, denominada Capa de Negocio y la Capa de Datos para el almacenamiento de los mismos, pudiéndose crear nuevas capas intermedias de ser necesario. Es decir, está soportado sobre un nivel de abstracción creciente, lo cual posibilita a los desarrolladores la fragmentación de un problema complejo en una secuencia de pasos incrementales. También proporciona una amplia reutilización, ya que los datos abstractos pueden ser utilizados por diferentes implementaciones o versiones de una misma capa en la medida que soporten las mismas interfaces.

La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y en caso de que ocurra algún cambio, sólo se analiza al nivel requerido sin tener que revisar entre código

mezclado. Además, permite distribuir el trabajo de creación de una aplicación por niveles, de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles.

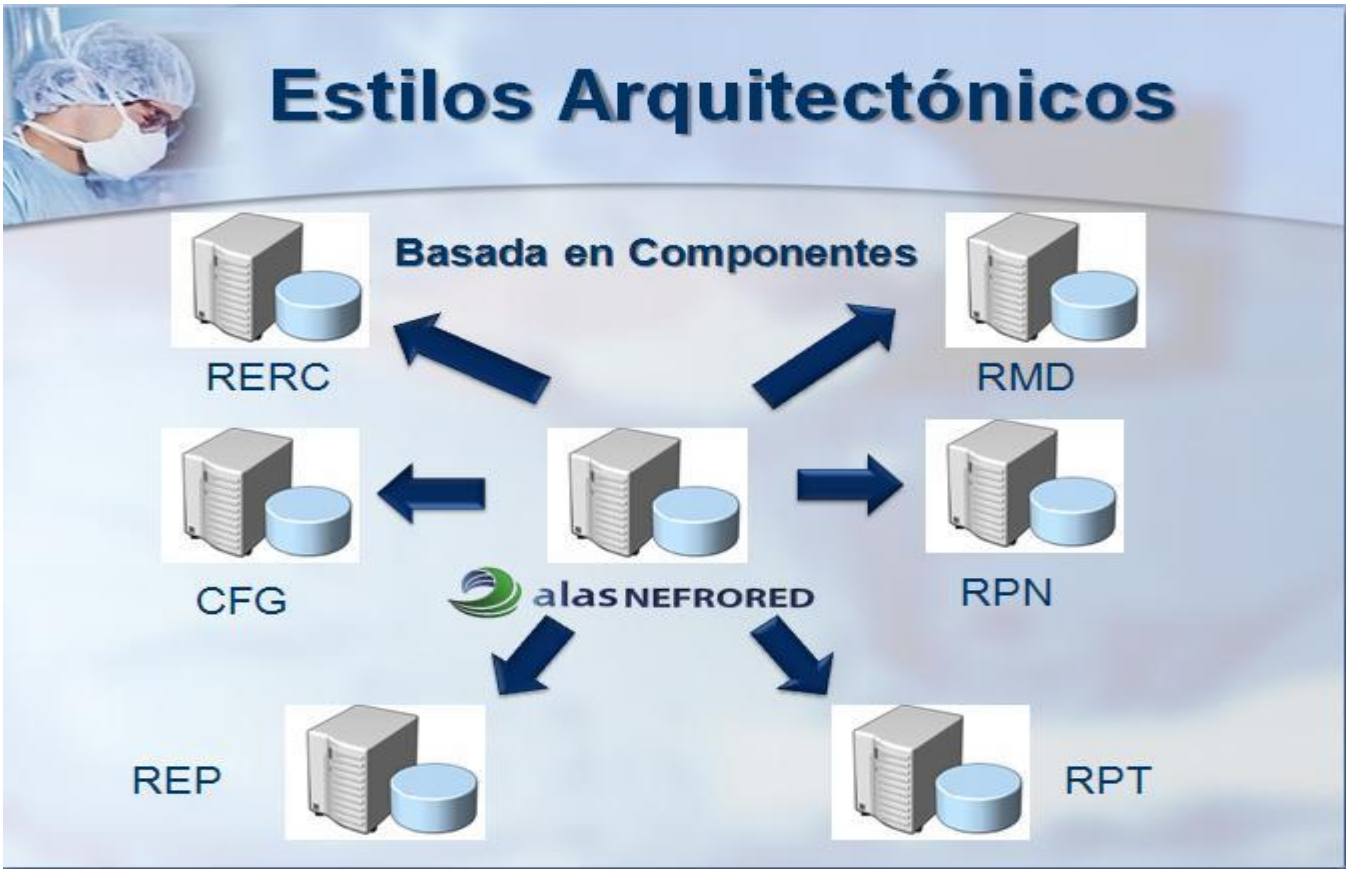
Diseño en tres niveles (o en tres capas).

**1.- Capa de presentación**: Es la que ve el usuario (denominada también "capa de usuario"), que presenta el sistema al usuario, le comunica la información y captura los datos pasados por el usuario en un mínimo de procesos (realiza un filtrado previo para comprobar que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio. También es conocida como interfaz grafica y debe tener la característica de ser "amigable" (entendible y fácil de usar) para el usuario.

**2.- Capa de negocio**: Es donde residen los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) porque es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos para almacenar o recuperar datos de él. También se consideran aquí los programas de aplicación.

**3.- Capa de datos:** Es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. Todas estas capas pueden residir en un único ordenador, si bien lo más usual es que haya una multitud de ordenadores, si el crecimiento de las necesidades lo aconseja se pueden separar en dos o más ordenadores. (33)

## Arquitectura Basada en Componentes



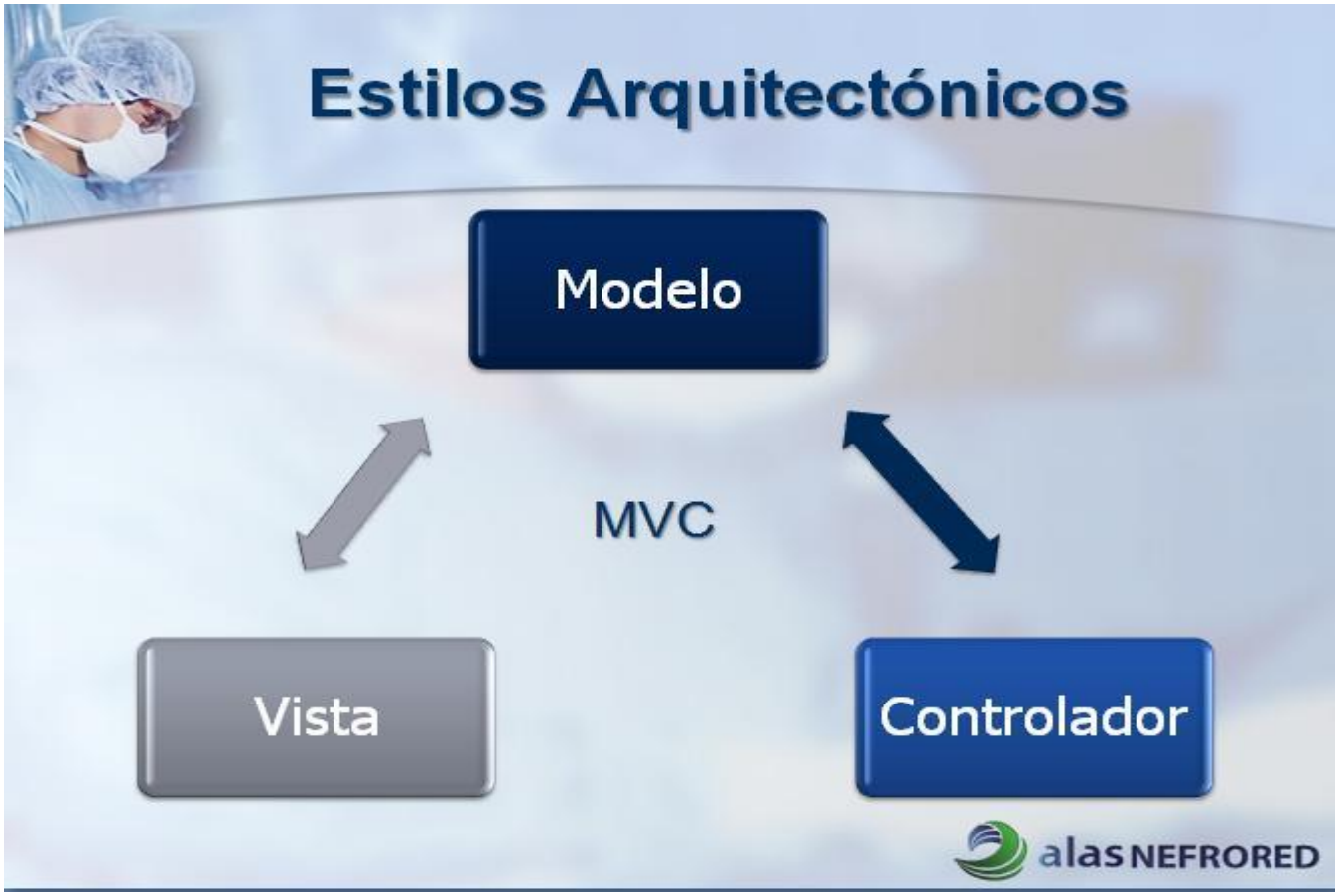

El objetivo de la tecnología de componentes software es construir aplicaciones complejas mediante ensamblado de módulos (componentes) que han sido previamente diseñados por otras personas a fin de ser rehusados en múltiples aplicaciones. La arquitectura software de una aplicación basada en componentes consiste en uno o un número pequeño de componentes específicos de la aplicación (que se diseñan específicamente para ella), que hacen uso de otros muchos componentes prefabricados que se ensamblan entre sí para proporcionar los servicios que se necesitan en la aplicación. Es decir radica en mapear los componentes funcionales en la arquitectura lógica de la aplicación. Para que una arquitectura de componentes pueda operar es necesario disponer de un entorno normalizado que proporcione soporte a los mecanismos con que se comunican las interfaces.

Se pueden hallar dentro de sus características:

**Reusabilidad de Servicios:** Reducción considerable de tiempos y costos de desarrollo de aplicaciones al utilizar servicios disponibles ya desarrollados, para resolver problemáticas comunes a otras aplicaciones. Aumentado por esta razón la robustez del nuevo sistema, al utilizarse software ya probado.

**Interoperatibilidad de aplicaciones:** Disminución de la complejidad en el proceso de integración, pues se interactúa con elementos que se abstraen de la tecnología y ubicación de los servicios. Entonces se puede decir que las características principales son la modularidad, la reusabilidad y compatibilidad, en la tecnología basada en componentes también se requiere robustez ya que los componentes han de operar en entornos mucho más heterogéneos y diversos. Su premisa es que los componentes cumplan con alta cohesión y bajo acoplamiento. (34)

### Modelo-Vista-Controlador




Para el diseño de aplicaciones con sofisticadas interfaces se utiliza el patrón de diseño Modelo-Vista-Controlador. Es muy importante pues simplifica la comprensión y la organización del desarrollo de sistemas complejos, aminorando las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Encima, ayuda a identificar qué puede reutilizarse.

Este patrón de arquitectura de software permite separar los datos de una aplicación, la interfaz de usuario y la lógica de negocio en tres componentes distintos, esto proporciona múltiples vistas sobre un mismo modelo de datos. El patrón MVC se usa frecuentemente en aplicaciones Web donde se utilicen diferentes interfaces de usuario y el código que provee los datos a la página es dinámico.

Elementos del patrón:

*El modelo*: Accede a la capa de almacenamiento de datos o acceso a datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento. Lleva un registro de las vistas y controladores del sistema. Ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

*El controlador*: Responde a las peticiones de los usuarios, recibe los eventos de entrada. Contiene reglas de gestión de eventos. Estas acciones pueden suponer peticiones al modelo.

*Las vistas*: Existen distintas formas de implementar el patrón, en la vista se reciben los datos del controlador y se los muestra al usuario. Define la interfaz de usuario.

Tanto la vista como el controlador dependen del modelo, el cual no depende de otros conceptos o clases. Esta separación permite construir y probar el modelo independientemente de la representación visual. En aplicaciones Web por ejemplo, la separación entre la vista (navegador) y el controlador (componentes del lado del servidor que manejan los requerimientos a través de HTTP) está muy claramente definida. Los requerimientos no funcionales de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas del negocio. Los clientes pueden preferir distintas opciones de representación pero dado que el modelo no depende de la vista, agregar nuevas opciones o modificar las ya existentes generalmente no afecta al modelo.

### Consumo de Servicios


En la aplicación Alasnefrored la arquitectura orientada a servicios se utiliza consumiendo servicios de diferentes sistemas como son: Sistema de Autentificación, Autorización y Auditoría (SAAA). Es el sistema que gestiona el nivel de acceso a los distintos componentes del SISalud, permitiendo que los usuarios tengan un solo usuario y una contraseña para acceder a todos los sistemas. También brinda a través de Web Services una serie de métodos que facilitan todos los servicios necesarios para asegurar el sistema. El Registro de Unidades de Salud, se utiliza para obtener la información que identifica a las unidades donde está ubicada los centros de diálisis. Además el Registro de Ciudadanos brinda la información primaria de los pacientes como es: su nombre, apellidos, CI, dirección y el Registro de Ubicación Geográfica, se encarga de gestionar la información de las provincias, municipios del país. Esto permite que el trabajo sea más fácil y con mayor rapidez, al poder trabajar con datos necesarios que solo se encuentran en estos servicios.

## Patrón Observador


El patrón Observador es uno de los más utilizados en la programación de sistemas, hay dos roles básicos en este patrón: el componente observable, que lanza eventos para notificar cambios en su estado, y el observador que espera y recibe esos eventos. El objeto observado no necesita saber nada acerca de los observadores. Son los observadores quienes deben registrarse como 'oyentes' para poder recibir eventos.

El bajo acoplamiento y que los oyentes reciben todos los eventos lanzados, sin distinción, se considera una ventaja ya que simplifica la posterior reutilización de componentes. Los oyentes reciben notificación de todos los eventos generados en los objetos observados. Pueden registrarse en cualquier momento durante el ciclo de vida del componente. El patrón es bastante sencillo, y puede encontrarse como parte de otros patrones más complejos tales como el patrón Modelo Vista Controlador, donde el controlador es un observador que recibe eventos de la interfaz, y la vista recibe eventos referidos a cambios en el modelo.

Este patrón debe aplicarse cuando:

- Cambios en algunos de los objetos del sistema requieren cambios en otros objetos del mismo grupo.
- El número de oyentes puede variar durante el ciclo de vida del objeto.
- El bajo acoplamiento es un requerimiento básico del diseño.

En la Aplicación se utiliza en el caso de uso Gestionar Estado de Aptitud, donde la modelo: m_AptoPACIENTERPT lanza los eventos, a los cuales se Subcriben las modelos de los estudios que se le realizan al paciente y devuelven el estado de aptitud del paciente según los estudios, el cual puede encontrarse en tres niveles: NoApto, AptoConDificultad y Apto. Estos niveles tienen nombre del estudio y descripción. Estos datos son los que le permiten al médico decidir si el paciente es Apto o NoApto.

## Patrones escogidos para el desarrollo de la aplicación

Los patrones o estilos arquitectónicos son soluciones exitosas a problemas comunes, hay patrones que abarcan las distintas etapas del desarrollo, desde el análisis hasta el diseño y desde la arquitectura hasta la implementación. En el desarrollo del proyecto se utilizaron varios patrones por las ventajas que brindan.

La implementación está basada sobre una arquitectura en tres capas según el modelo Cliente Servidor, este puede actuar como una sola entidad y también pueden actuar como entidades separadas, realizando actividades o tareas independientes. Las funciones de Cliente y Servidor pueden estar en plataformas separadas, o en la misma plataforma y el Modelo Tres Capas tiene como ventaja principal que el

desarrollo se puede llevar a cabo en varios niveles y en caso de algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado. Además, permite distribuir el trabajo de creación de una aplicación por niveles, de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles. Asimismo se utiliza el patrón Modelo-Vista-Controlador que nos brinda tres módulos que nos facilitan el trabajo:

*El modelo*: Accede a la capa de almacenamiento de datos o acceso a datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento. Lleva un registro de las vistas y controladores del sistema. Ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

*El controlador*: Responde a eventos y modifica la vista. Recibe los eventos de entrada. Contiene reglas de gestión de eventos. Estas acciones pueden suponer peticiones al modelo.

*Las vistas*: Existen distintas formas de implementar el patrón, en la vista se reciben los datos del controlador y se los muestra al usuario. Define la interfaz de usuario.

Se usó además una arquitectura basada en componentes la cual se utiliza para construir aplicaciones complejas, donde se tienen diferentes componentes que se ensamblan entre si para brindar los servicios que necesita la aplicación, en el caso de Nefrología cada módulo de la aplicación constituye un componente los cuales se integran para brindar los servicios necesarios para la atención a pacientes con enfermedades renales crónicas.

En la arquitectura de la aplicación Nefrología se puede ver un consumo de servicios de diferentes sistemas como son: Sistema de Autentificación, Autorización y Auditoría (SAAA) es el sistema que gestiona el nivel de acceso a los distintos componentes del SISalud, permitiendo que los usuarios tengan un solo usuario y una contraseña para acceder a todos los sistemas, brinda a través de Web Services una serie de métodos que facilitan todos los servicios necesarios para asegurar el sistema, Registro de Unidades de Salud ,se utiliza para obtener la información que identifica a las unidades donde está ubicada los centros de diálisis, Registro de Ciudadanos el cual brinda la información primaria de los pacientes como es: su nombre, apellidos, CI, dirección y Registro de Ubicación Geográfica ,se encarga de gestionar la información de las provincias, municipios.

Esto permite que el trabajo sea más fácil y con mayor rapidez, al poder trabajar con datos necesarios que solo se encuentran en estos servicios. Se utilizó en patrón de diseño Observador que es uno de los más utilizados en la programación de sistemas.

## Conclusiones

En este capítulo se abordaron los temas relacionados con el objeto de estudio y los objetivos del presente trabajo. Se profundiza sobre las tendencias y tecnologías actuales así como las herramientas a utilizar en el perfeccionamiento de dicha aplicación, logrando así establecer las más adecuadas para el desarrollo de la misma. Además se caracterizan los patrones de arquitectura utilizados durante el desarrollo de la aplicación. Todo esto con el objetivo de profundizar en el conocimiento de estos temas y fundamentar la necesidad de dar una solución informática a la situación existente en los centros de diálisis del país.

## Capitulo II: Descripción de la Arquitectura

En el actual capítulo se explican y seleccionan los requerimientos no funcionales, se hace una descripción de la arquitectura, analizan las posibles implementaciones, componentes o módulos ya existentes que puedan ser rehusados. Se puede encontrar, además, cómo se controla la seguridad en la aplicación, así como la vista de implementación y las estrategias de codificación, estándares y estilos a utilizar.

### 2.1 Requerimientos no funcionales

**Usabilidad**

- La aplicación Web será flexible y de fácil aprendizaje, pues se trata en todo lo posible de mantener un estándar de operabilidad que logre que las interacciones del usuario con el sistema sean predecibles y familiares.
- El sistema podrá ser usado por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente Web en sentido general.
- Deberá visualizarse en Mozilla Firefox.
- Estará disponible en todo momento.

**Rendimiento.**

- Las pantallas estarán poco cargadas de imágenes para garantizar que el tiempo de ejecución de los hipervínculos, las adiciones, modificaciones y eliminaciones, no haga tedioso el trabajo con la aplicación.

**Soporte**

- El sistema contará con una ayuda para el usuario con la cual podrá aprender rápidamente a utilizar la aplicación Web
- Estará bien documentado para garantizar futuros mantenimientos.

**Portabilidad**.

- El sistema podrá ejecutarse sobre plataforma Linux, Windows 98 o superior.

**Seguridad**.

- La protección del sistema contra el acceso desautorizado y las modificaciones de información está garantizada por el Sistema de Autentificación, Autorización y Auditoría (SAAA).Aunque en la aplicación no se controla la auditoría.

**Confiabilidad.**

- Todas las partes del diseño del sistema serán realmente aplicadas y se hará una transformación correcta del diseño en un lenguaje de programación.
- Deberá prevenir los posibles fallos y/o errores que pudieran presentarse y posibilitar una rápida recuperación en dichos casos.

**Software**

**Para el cliente:**

- Tendrán acceso al sistema a través de cualquier navegador Web, recomendados: Mozilla Firefox 2.0 o superior.
- Sistema operativo Linux o Windows 98 ó Superior.

**Para el servidor:**

- Sistema operativo Linux.
- Servidor Web Apache 2.0 PHP 5.0 instalados.
- Framework CodeIgniter
- Servidor de Base de Datos MySQL 5.0

**Hardware.**

**Para el cliente:**

- Procesador Pentium III o superior.
- 256 de memoria RAM o superior.
- Monitor VGA o superior.
- Tarjeta de red.

**Para el servidor:**

- Procesador Pentium IV o superior.
- 2Gb de memoria RAM o superior.
- Disco Duro de 80 GB
- Monitor VGA o superior.
- Tarjeta de red.

**Restricciones de diseño.**

- Utilizar los patrones de diseño establecidos.
- Para el análisis y el diseño del sistema debe ser utilizada la metodología RUP, usando el lenguaje de modelación UML y como herramienta para llevarlo a cabo el Visual Paradigm.
- Implementado con el lenguaje de programación php 5
- Desarrollado en Zend Studio

**Interfaz**

- La interfaz de usuario será sencilla, amigable, intuitiva y de fácil navegación por el usuario, con el objetivo de evitar la resistencia de los usuarios al usar el nuevo sistema.
- Se seleccionará un esquema de colores a la vez atractivo pero que no canse.
- Paginación de reportes de búsqueda, y listados.
- Diseño perfectamente encuadrado para resoluciones de 1024 x 768.

## 2.2 Descripción de la arquitectura, fundamentación.

La Arquitectura del sistema Alas NefroRed está compuesta por diferentes patrones arquitectónicos, permitiendo trabajar con mayor seguridad y rapidez. Se trabaja con el Modelo Tres Capas el cual permite dividir la carga en tres partes con un reparto claro de funciones: una Capa de Presentación, otra parte para las reglas lógicas del negocio, denominada Capa de Negocio y la Capa de Datos para el almacenamiento de los mismos, pudiéndose crear nuevas capas intermedias de ser necesario. Es decir, está soportado sobre un nivel de abstracción creciente, lo cual posibilita a los desarrolladores la fragmentación de un problema complejo en una secuencia de pasos incrementales.

También proporciona una amplia reutilización, pues los datos abstractos pueden ser utilizados por diferentes implementaciones o versiones de una misma capa en la medida que soporten las mismas interfaces. Además trabajando con el Modelo Tres Capas también se hace uso del Modelo Cliente-Servidor, donde el cliente y el servidor pueden actuar como una sola entidad y también pueden actuar como entidades separadas, realizando actividades o tareas independientes y las funciones de Cliente y Servidor pueden estar en plataformas separadas, o en la misma plataforma.

En la aplicación se hace uso además de la Arquitectura Basada en componentes, la cual se utiliza para construir aplicaciones complejas, donde se tienen diferentes componentes que se ensamblan entre si para brindar los servicios que necesita la aplicación, en el caso de Nefrología cada módulo de la aplicación constituye un componente los cuales se integran para brindar los servicios necesarios para la atención a pacientes con enfermedades renales crónicas.

Se utiliza fundamentalmente el patrón Modelo Vista Controlador, su uso es de suma importancia porque simplifica la comprensión y la organización del desarrollo de sistemas complejos, reduciendo las dependencias de forma que las capas más bajas no son conscientes de ningún detalle o interfaz de las superiores. Además, ayuda a identificar qué puede reutilizarse. Este patrón de arquitectura de software permite separar los datos de una aplicación, la interfaz de usuario y la lógica de negocio en tres componentes distintos, esto proporciona múltiples vistas sobre un mismo modelo de datos. El patrón MVC se emplea frecuentemente en aplicaciones Web donde se utilicen diferentes interfaces de usuario y el código que provee los datos a la página es dinámico.

# Capítulo 2: Descripción de la arquitectura.

Para construir una aplicación utilizando el patrón MVC hay que definir tres clases de módulos:

*El modelo*: Accede a la capa de almacenamiento de datos o acceso a datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento. Lleva un registro de las vistas y controladores del sistema. Ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.

*El controlador*: Responde a eventos y modifica la vista. Recibe los eventos de entrada. Contiene reglas de gestión de eventos. Estas acciones pueden suponer peticiones al modelo.

*Las vistas*: Existen distintas formas de implementar el patrón, en la vista se reciben los datos del controlador y se los muestra al usuario. Define la interfaz de usuario.

Se utilizaron rasgos de La Arquitectura Orientada a Servicios, que define la utilización de servicios para dar soporte a los requisitos del negocio. Permite la creación de sistemas altamente escalables que reflejan el negocio de la organización, a su vez brinda una forma estándar de exposición e invocación, lo cual facilita la interacción entre diferentes sistemas propios o de terceros.

En la aplicación Nefrología la arquitectura orientada a servicios se utiliza consumiendo servicios de diferentes sistemas como son: Sistema de Autentificación, Autorización y Auditoría (SAAA) es el sistema que gestiona el nivel de acceso a los distintos componentes del SISalud, permitiendo que los usuarios tengan un solo usuario y una contraseña para acceder a todos los sistemas, brinda a través de Web Services una serie de métodos que facilitan todos los servicios necesarios para asegurar el sistema. El Registro de Unidades de Salud ,se utiliza para obtener la información que identifica a las unidades donde está ubicada los centros de diálisis, Registro de Ciudadanos el cual brinda la información primaria de los pacientes como es: su nombre, apellidos, CI, dirección y Registro de Ubicación Geográfica ,se encarga de gestionar la información de las provincias, municipios del país. Esto permite que el trabajo sea más fácil y con mayor rapidez, al poder trabajar con datos necesarios que solo se encuentran en estos servicios.

## 2.3 Análisis de posibles implementaciones, componentes o módulos ya existentes y que puedan ser rehusados. Estrategias de integración

El sistema está compuesto por 4 módulos Registro de Enfermos Renales Crónicos (RERC), Registro de Métodos Depuradores (RMD), Registro de Pacientes en Nefrología (RPN) y Configuración (CFG).

Para el desarrollo de los nuevos módulos Gestión de Reportes y Registro de Trasplantes se hará un uso extendido de los antes mencionados:

Registro de Enfermos Renales Crónicos (RERC): Los pacientes son registrados en este módulo cuando son declarados como enfermos renales crónicos y por lo tanto necesitan atención especializada. Este módulo permite la gestión de la información relacionada a los estudios evolutivos que se les hace a los pacientes.

Registro de Métodos Depuradores (RMD): En este módulo se registrará la información sobre los métodos depuradores que se le aplican a los pacientes, tanto para las diálisis como las diálisis peritoneales, pues la mayoría de los pacientes con este tipo de enfermedad se encuentran en el programa de hemodiálisis.

Algunas de las informaciones que registra este sistema son las indicaciones de la hemodiálisis, los datos de la hemodiálisis, las complicaciones intradialíticas, las complicaciones con el acceso vascular, así como todos los accesos que se le han realizado al paciente, entre otras.

En la primera versión solo se tienen en cuenta los pacientes con enfermedad renal crónica pero el módulo está preparado para que en próximas versiones pueda registrar a todo aquel paciente que necesite un método depurador y que no sea necesariamente enfermo renal crónico.

Registro de Pacientes en Nefrología (RPN): Permite gestionar la información general de los pacientes que están registrados en el sistema cuya información no es específica a ningún módulo en particular. Como puede ser nombre, CI, dirección particular, hospitalizaciones que ha tenido, las complicaciones, etc.

Configuración (CFG): Permite gestionar los nomencladores que son necesarios para el buen funcionamiento del sistema.

Para el desarrollo del módulo Gestión de Reportes se utilizarán los cinco módulos, ya que el objetivo fundamental es obtener estadísticas de los pacientes que estén en el Registro de Pacientes en Nefrología a los cuales se les aplica algún método depurador, este paciente se encuentra además en el Registro de Enfermos Renales Crónicos en espera de un trasplante. Además se hace uso del módulo de Configuración, donde se gestionan los nomencladores que son necesarios para el buen funcionamiento del sistema y se utiliza el módulo Registro de Pacientes a Trasplantar donde se tiene la información referente a los pacientes que se le realizará el trasplante. Se realiza un consumo de información de cada uno de estos módulos que se encuentra en la base de datos.

Registro de Pacientes a Trasplantar utilizará tres módulos, Registro de Pacientes en Nefrología, donde se tiene la información general del paciente, Registro de Enfermos Renales Crónicos, son aquellos pacientes en espera de un posibles trasplante y el módulo de Configuración donde se gestionan los nomencladores que son necesarios para el buen funcionamiento del sistema. (35)

## 2.4 Seguridad

El SISalud es una aplicación que integra un conjunto de servicios y componentes distribuidos geográficamente, en constante interacción a través de la Red de INFOMED.

Este sistema está compuesto por el Registro Informatizado de Salud (RIS), el Sistema Informatizado de Atención Primaria (SIAP), el Sistema Informatizado de Gestión Hospitalaria (SIGH) y el Sistema Informatizado de Atención Especializada (SIAE).

El Sistema Red Cubana de Nefrología pertenece al Sistema Informatizado de Atención Especializada (SIAE) pues es aquí, donde se agruparán los módulos que pertenecen al nivel de atención terciario o especializado. Siguiendo esta filosofía el sistema que aquí se propone necesita para su buen funcionamiento integrarse a una serie de componentes ya desarrollados para el SISalud.

Del Registro Informatizado de la Salud (RIS) para el control de la seguridad se integra con el: **SAAA** (Sistema de Autentificación, Autorización y Auditoría) Módulo de seguridad y Administración, está basado en un modelo de autenticación, autorización y auditoría (SAAA). Este fue desplegado en diciembre del 2003, como componente de seguridad, para las aplicaciones desarrolladas con el fin de informatizar el

Sistema Nacional de Salud (SNS).En el año 2007 se desarrolló una versión renovada del componente de seguridad SAAA, solucionando dificultades existentes y agregando, además, nuevos requerimientos funcionales.

El sistema formaliza el nivel de acceso a los distintos componentes del SISalud, permitiendo que los usuarios tengan un solo usuario y una contraseña para acceder a todos los sistemas, brindando, a través de Web Services, una serie de métodos que facilitan todos los servicios necesarios para asegurar el sistema. Para su uso se crearon diferentes clases que encapsulan el manejo de la seguridad así como la comunicación con este módulo, de manera que esta queda de forma global para todo el sistema sin necesidad de ser implementada en cada subsistema pero sí utilizada por todos y de manera particular por cada uno. (36)

## 2.5 Vista de Despliegue

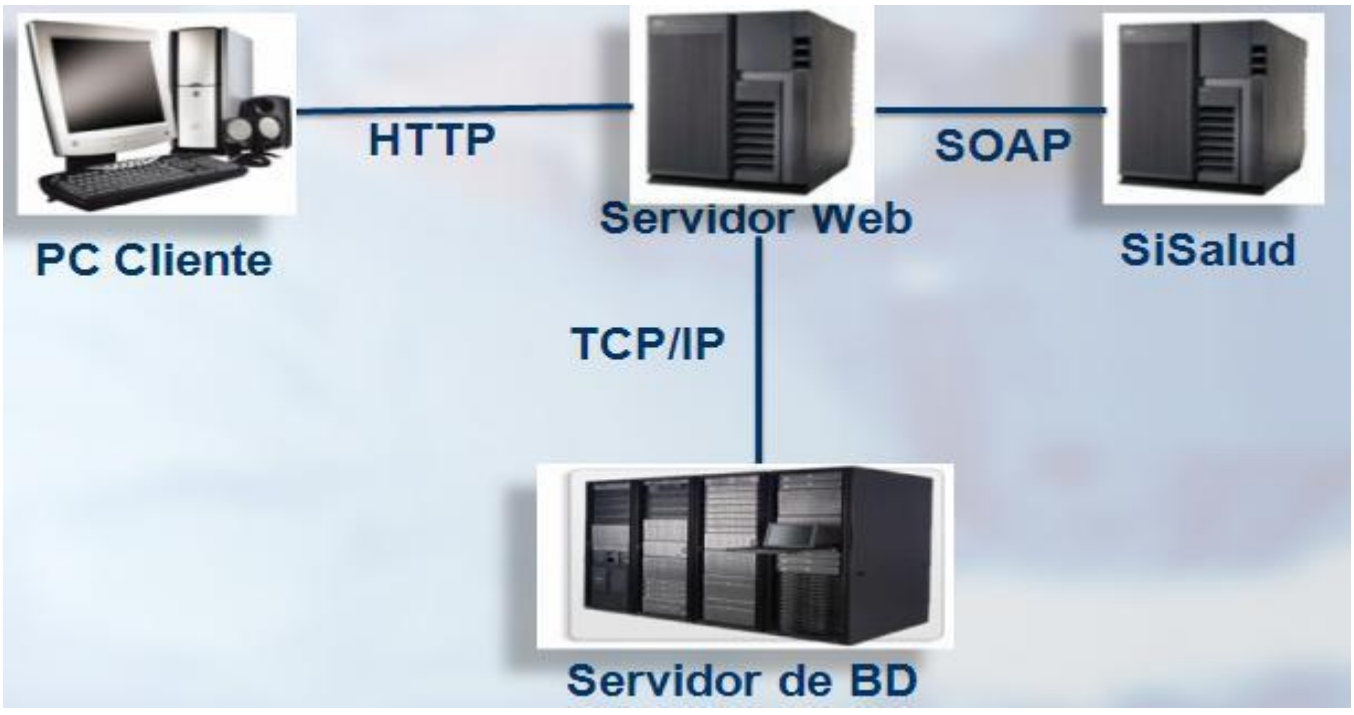



**Fig. 2. 1. Diagrama de la Vista de Despliegue**

**SISalud**: En este nodo se encuentran diferentes servicios con los que debe interactuar la Red Cubana de Nefrología.

**Servidor Web Nefrología**: En este nodo se encuentra la aplicación Red Cubana de Nefrología.

**Servidor de Base de Datos:** En este nodo se encuentra el servidor de la BD del sistema Red Cubana de Nefrología.

**PC cliente:** Desde ella se accede a través de un navegador al sistema Red Cubana de Nefrología.

## 2.6 Estrategias de codificación. Estándares y estilos a utilizar

**Estándar de codificación para PHP**

Idioma:

Se debe utilizar como idioma el español, las palabras no se acentuarán.

Palabras Reservadas:

Las palabras reservadas van en minúsculas sin excepción alguna.

| **Identación** | | |
|---|---|---|
| **0 espacios en blanco desde la izquierda en** | require<br>include<br>class | No se empleará ningún espacio en blanco desde la izquierda para las instrucciones antes mencionadas. Se tomará como inicio de la página el tag PHP <? |
| **2 espacio en blanco desde la izquierda en** | function<br>define | Se dejarán dos espacios en blanco desde la izquierda en las instrucciones antes mencionadas. |

| | | |
|---|---|---|
| **2 espacio en blanco desde la referencia en** | 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. |
| **Niveles de anidación** | Hasta 5 niveles | Se recomienda emplear hasta 5 niveles de anidación en instrucciones If, For, While. |
| **Aspectos Generales** | El indentado debe ser de espacios por bloque de código. No se debe usar el tabulador; ya que este puede variar según la PC o la configuración de dicha tecla.<br>Los inicios ({) y cierre (}) de ámbito deber estar alineados debajo de la declaración a la que pertenecen y deben evitarse si hay sólo una instrucción. Nunca colocar {en la línea de un código cualquiera, esto requiere una línea propia. | |
| | **Comentarios, separadores, líneas, espacios en blanco y márgenes.** | |
| **Ubicación de comentarios** | Al inicio de cada clase o función y al final de cada bloque de código, si existe. | Se recomienda comentar al inicio de la clase o función especificando el objetivo de la misma. |
| **Líneas en blanco** | Se emplean antes y después de métodos y clases. | 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** | Entre operadores lógicos y aritméticos. | Se recomienda usar un espacio en blanco entre estos operadores para lograr una mayor legibilidad en el código. Ejemplo: $sAcceso = $sAccesoH if (($sAcceso) && ($sLocalizacion)) |
| **Márgenes y líneas de continuidad** | Sobre márgenes y líneas de continuación | Los márgenes de cada línea de código no deben exceder los 80 caracteres, pero puede exceptuarse si es para terminar la escritura de una palabra. Las líneas de continuación deben estar alineadas entre sí e indentadas respecto al paréntesis abierto. Las líneas de continuación nunca deben comenzar con un Operador binario. |
| **Aspectos generales** | Sobre el comentario | Se debe evitar comentar cada línea de código. Cuando el comentario se aplica a un grupo de instrucciones debe estar seguido de una línea en blanco. En caso de que se necesite comentar una sola instrucción se suprime la línea en blanco o se escribe a continuación de la instrucción. |
| | Sobre los espacios en blanco | No se debe usar espacio en blanco: Después del corchete abierto y antes del cerrado de un |

| | | |
|---|---|---|
| | | arreglo. Después del paréntesis abierto y antes del cerrado. |
| | **Variables y constantes** | |
| **Apariencia de variables** | Las variables tendrán nomenclatura camello. | Se utiliza la nomenclatura camello, con la primera palabra completa en minúsculas. |
| **Apariencia de constantes** | Todas sus letras en mayúscula | Se recomienda declarar una constante por cada línea y con las asignaciones a las variables sucede lo mismo. Las constantes van en mayúscula.<br>Ejemplo:<br>define(“CONSTANT1”,”value1”);<br>define(“CONSTANT2”,”value2”);<br>$sAcceso = $sAccesoH |
| **Declaración de constantes y asignación a variables.** | Nombres de las variables y constantes | El nombre empleado, debe permitir que con sólo leerlo se conozca el propósito de la misma. |
| | **Clases y Objeto** | |
| **Para nombrar clases** | La primera letra con la inicial del tipo, y luego seguida de un (_) | Se usa c_ para las controladoras, m_ para las modelos v_ para los ficheros de las vistas, aunque las vistas no son clases. |

| | | |
|---|---|---|
| **Apariencia de clases y objetos** | Primera letra en mayúscula. | 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*. |
| **Apariencia de atributos** | Primera letra en minúscula | El nombre que se le da a los atributos de las clases debe comenzar con la primera letra en minúscula, en caso de que sea un nombre compuesto se empleará notación CamellCasing. |
| **Apariencia de las funciones** | Primera letra en mayúscula | Para nombrar las funciones se debe tratar de utilizar verbos que denoten la acción que hace la función. Se empleará notación PascalCasing. Ejemplo: function BuscarUnidad (). Si son funciones que obtienen un dato se emplea el prefijo get y si fijan algún valor se emplea el prefijo set. |
| **Declaración de parámetro en funciones** | Agrupados por tipos Poner los string numéricos, además agrupar según valores por defecto. | Los parámetros que se le pasan a las funciones se recomienda sean declarados de forma tal que estén agrupados por el tipo de dato que contienen, especificando el tipo de datos. |
| **Aspectos generales** | Sobre las clases, los objetos, los atributos y las funciones. | El nombre empleado para las clases, objetos, atributos y funciones debe permitir que con |

| | | |
|---|---|---|
| | | sólo leerlo se conozca el propósito de los mismos. |
| | **Bases de Datos, Tablas, esquemas y Campos.** | |
| **Apariencia de la BD** | Las 2 primeras letras representan el tipo. | Los nombres de las Bases de Datos deben comenzar con el prefijo bd a continuación underscoard y luego el nombre comienza con mayúscula y el resto en minúscula. Se utilizará además para las vistas tv_, para las tablas e_ y todas las entidades tendrá como sufijo el nombre del modulo a que pertenecen separado por _. |
| **Apariencia de los campos** | Todas las letras en minúscula. | El nombre a emplear para los campos debe escribirse con todas las letras en minúscula para evitar problemas con el Case Sensitive del gestor. Si está compuesto por más de una palabra se debe usar (_). |
| **Nombre de los campos** | En caso de identificadores. | Todos los campos identificadores van a comenzar con el identificador id seguido de underscoard y posteriormente el nombre del campo Ejemplo: |

| | | |
|---|---|---|
| | | id _ paciente. |
| | **Controles** | |
| **Apariencia de los controles.** | Los controles tendrán un prefijo para el tipo de datos en minúscula. | El nombre que se le da a los controles debe comenzar con las primeras letras en minúscula, las cuales identificarán el tipo de datos al que se refiere; en caso de que sea un nombre compuesto se empleará notación CamellCasing. Ejemplo: btnAceptar. |

**Tabla 2. 1. Estrategias y estándares de codificación.**

## Además se utiliza el estándar de codificación PEAR:

PEAR es la abreviatura de "Extensión de PHP y repositorio de aplicaciones". Es parte del proyecto PHP y consiste en una biblioteca de extensión programada en lenguaje PHP que permite desarrollar aplicaciones, módulos y extensiones de alto nivel y calidad, fue fundado en 1999 por Stig Bakken S. (37)

Con el propósito de facilitar:

Una estructura de biblioteca de código fuente abierto para los usuarios de PHP, un sistema de código de distribución y mantenimiento de paquetes, ya que el código de PEAR es particionada en "paquetes". Cada paquete es un proyecto independiente con su propio equipo de desarrollo, número de versión, el ciclo de liberación, de documentación y de una determinada relación con otros paquetes, los paquetes se distribuyen como ficheros tar gzip con un archivo de descripción de su interior, e instalado en su sistema local usando el instalador de PEAR. Además proporciona una norma de estilo para código escrito en PHP y La Comunidad de Extensión de PHP Library (PECL). (38)

## Conclusiones

En el capítulo se describieron los requisitos no funcionales del sistema, los que se tendrán en cuenta para realizar la aplicación. Fueron mostrados los estándares y las estrategias de codificación que están presentes en el sistema, para que el código esté más organizado y fácil de entender. Además, se explica como se controla la seguridad del sistema y se muestra mediante la vista de despliegue como será el acceso al sistema una vez instalado.

Se ha establecido una arquitectura centralizada, respetando la establecida en los servidores de Infomed, Basada en Componentes, Modelo en Tres Capas y basada en el patrón Modelo-Vista- Controlador, además se analizaron las posibles implementaciones, componentes o módulos ya existentes y que puedan ser rehusados.

# Capítulo 3: Descripción y análisis de la solución propuesta

**Capitulo III: Descripción y Análisis de la Solución Propuesta**

El presente capítulo se centrará en la descripción y análisis de la solución propuesta, donde se hará una valoración del diseño propuesto por el analista, se establecerán las clases u operaciones necesarias para el buen funcionamiento de la aplicación y las técnicas de validación establecidas para la integridad y normalización de la base de datos. Se expondrán mediante esquemas el Modelo de Datos y la Vista de Implementación con sus correspondientes Diagramas de Componentes.

## 3.1 Valoración crítica del diseño propuesto por el analista

En el análisis y diseño de los módulos Gestión de Reportes y Registro de Pacientes a Trasplantar se definieron 3 tipos de clases: modelo, controlador y entidad, las cuales están bien definidas y tienen la calidad requerida para lograr el correcto funcionamiento de la aplicación. Luego de analizar que el sistema va a ser implementado haciendo uso del patrón Modelo Vista Controlador donde la clase modelo gestiona la lógica de datos asegurando la integridad de estos, la clase controladora que es la encargada de manejar los eventos que proceden de las vistas ,las cuales muestran la información que desea el usuario. Además la clase entidad va a modelar toda la información que generalmente va a ser persistente.

Además en todos los diagramas se representa la clase NFRWeb_Controller, NFRCore_Model y NFRCore_Entidad que son clases padres, de la primera heredan todas las controladoras, de la segunda todas las modelos y de la terceras todas las entidades, estas clases son una extensión de la controladora y la modelo del Framework CodeIgniter respectivamente.

Se desarrollaron además diagramas de interacción (Secuencia) los cuales permiten entender mejor la aplicación, ya que están bien organizados y se capta de ellos lo que en realidad se desea.

3.2 Descripción de las nuevas clases u operaciones necesarias

## Caso de Uso Registrar paciente en RPT

| **Nombre: c_pacienterpt** | |
|---|---|
| **Tipo de Clase: Controladora** | |
| **Por cada responsabilidad:** | |
| Nombre: | Registrar() |
| Descripción: | Registra los pacientes que necesitan servicios en el registro de pacientes a trasplantar. |

**Tabla 3. 1. Descripción de la Clase c_pacienterpt**

| Nombre: m_pacienterpt | |
|---|---|
| **Tipo de Clase: Controladora** | |
| **Por cada responsabilidad:** | |
| Nombre: | getByIdInterno($id,$campo='id_entrada') |
| Descripción: | Si el id de entrada coincide con el id que tiene el paciente entonces me devuelve el paciente |
| Nombre: | getById($id) |
| Descripción: | Devuelve un paciente dado el Id |
| Nombre: | Listar() |
| Descripción: | Muestra un listado con los pacientes |
| Nombre: | RegistrarPaciente(E_PacienteRPT $paciente) |
| Descripción: | Registra los pacientes que necesitan servicios en el registro de pacientes a trasplantar. |
| Nombre: | getByIdPaciente($id) |
| Descripción: | Devuelve un paciente dado el Id |

**Tabla 3. 2. Descripción de la clase m_pacienterpt**

| Nombre: e_pacienterpt | |
|---|---|
| **Tipo de Clase: Entidad** | |
| Atributo | Tipo |
| $fecha_entrada | var |
| $donante | var |
| $id_entrada | var |
| **Por cada responsabilidad:** | |
| Nombre: | copiar($objetos) |
| Descripción: | Toma los valores que vienen por Post |
| Nombre: | getID() |
| Descripción: | Devuelve el id de entrada del paciente |

**Tabla 3. 3. Descripción de la clase e_pacienterpt**

## Caso de Uso Gestionar Estado de Aptitud

| Nombre:c_aptopacienterpt | |
|---|---|
| **Tipo de Clase: Controladora** | |
| **Por cada responsabilidad:** | |
| Nombre: | index () |
| Descripción: | Muestra una página con los estudios realizados al paciente. |
| Nombre: | UltimoEstudio () |
| Descripción: | Muestra el último estudio que se le realizó al paciente. |
| Nombre: | getById ($id = 0) |
| Descripción: | Devuelve el Estado de Aptitud de un paciente dado el ID. |
| Nombre: | Registrar () |
| Descripción: | Registra el Estado de Aptitud del paciente. |
| Nombre: | Editar () |
| Descripción: | Modifica el Estado de aptitud del paciente, si no han pasado 48 horas. |

| | |
|---|---|
| Nombre: | Históricos () |
| Descripción: | Muestra la vista. |
| Nombre: | Listar ($startIndex = 0, $results = 10, $orden = 'fecha', $dir = 'desc') |
| Descripción: | Inserta los datos en la vista mostrada por Históricos (). |

**Tabla 3. 4. Descripción de la clase c_aptopacienterpt.**

| **Nombre:m_aptopacienterpt** | |
|---|---|
| **Tipo de Clase: Controladora** | |
| **Por cada responsabilidad:** | |
| Nombre: | IniciarRegistro () |
| Descripción: | Lanza un evento y devuelve un resumen de los estudios realizados al paciente. |
| Nombre: | getById ($id) |
| Descripción: | Devuelve el Estado de Aptitud del |

| | |
|---|---|
| | paciente dado el ID. |
| Nombre: | Registrar ($aptoPaciente) |
| Descripción: | Registra el Estado de Aptitud del paciente |
| Nombre: | Editar (E_AptoPacienteRPT $estudio) |
| Descripción: | Cambia el Estado de aptitud del paciente, si no han pasado 48 horas. |
| Nombre: | EstadosHistoricos (EstructuraEntrada $entrada, $pacienteRPT) |
| Descripción: | Muestra todos los Estados de Aptitud del paciente. |
| Nombre: | ActualizarEstadoAptitud (E_EstudioAptoRPT $estudio). |
| Descripción: | Permite marcar la opción (caduco) al Estado de Aptitud del paciente, luego de haber modificado el estudio realizado. |

**Tabla 3. 5 Descripción de la clase m_aptopacienterpt.**

| Nombre:e_estructurasalidaestadosaptitud | |
|---|---|
| **Tipo de Clase: Entidad** | |
| Atributo | Tipo |
| $Lista=array(); | var |
| Descripción | Muestra un arreglo con los Estados de Aptitud del paciente. |

**Tabla 3. 6. Descripción de la clase e_estructurasalidaestadosaptitud**

| Nombre:e_estudioaptorpt | |
|---|---|
| **Tipo de Clase: Entidad** | |
| Atributo | Tipo |
| $nombre | var |
| $nivel | var |
| $observaciones | var |
| $id_e_apto_paciente_rpt | var |

**Tabla 3. 7. Descripción de la clase e_estudioaptorpt**

| **Nombre:e_resumenestudioaptorpt** | |
|---|---|
| **Tipo de Clase: Entidad** | |
| Atributo | Tipo |
| $arrEstudiosNivelApto = array (); | var |
| $arrEstudiosNivelNoApto = array (); | var |
| $arrEstudiosNivelAptoConDifultades = array (); | var |
| Descripción | Contiene tres arreglos con los datos referentes al Estado de Aptitud del paciente, luego de realizarle el estudio, si es Apto, NoApto o AptoConDificultades, estos datos se encuentran en la tabla de la base de datos. |
| **Por cada responsabilidad:** | |

| | |
|---|---|
| Nombre: | add ($estudio) |
| Descripción | Adiciona un estudio realizado |
| Nombre: | getNivelApto () |
| Descripción | Devuelve el Estado de Aptitud. |
| Nombre: | getNivelNoApto () |
| Descripción | Devuelve el Estado de Aptitud. |
| Nombre: | getNivelAptoConDifultades () |
| Descripción | Devuelve el Estado de Aptitud. |
| Nombre: | fucionarNiveles () |
| Descripción | Une todos los niveles para un correcto funcionamiento y devuelve un arreglo. |

**Tabla 3. 8. Descripción de la clase e_resumenestudioaptorpt**

| Nombre:e_aptopacienterpt | |
|---|---|
| **Tipo de Clase: Entidad** | |
| Atributo | Tipo |
| $fecha; | var |
| $observaciones; | var |
| $resumenEstudio = array (); | var |
| $fecha_creado; | var |
| $caduco; | var |
| $id_e_pacienterpt; | var |
| Descripción | Contiene todos los datos que se encuentran en la tabla de de la base de datos. |
| **Por cada responsabilidad:** | |
| Nombre: | setE_PacienteRPT ($paciente) |
| Descripción | Asigna el Id a un paciente RPT. |

**Tabla 3. 9. Descripción de la clase e_aptopacienterpt**

## Caso de Uso Gestionar Estudios

| Nombre:c_ecocardiogramarpt | |
|---|---|
| **Tipo de Clase: Controladora** | |
| **Por cada responsabilidad:** | |
| Nombre: | index () |
| Descripción | Muestra una página con el último estudio realizado. |
| Nombre: | Registrar () |
| Descripción | Registra el estudio realizado al paciente. |
| Nombre: | UltimoEstudio () |
| Descripción | Muestra el último estudio realizado al paciente. |
| Nombre: | getById ($id = 0) |
| Descripción | Devuelve el Estudio del paciente dado el ID. |
| Nombre: | Históricos () |
| Descripción | Muestra la vista. |
| Nombre: | Listar ($startIndex = 0, $results = 10, $orden = 'fecha', $dir = |

| | |
|---|---|
| | 'desc') |
| Descripción | Inserta los datos en la vista mostrada por Históricos (). |
| Nombre: | Editar ($id = 0) |
| Descripción | Cambia el Estudio realizado al paciente, si no han pasado 48 horas. |
| Nombre: | Eliminar ($id) |
| Descripción | Permite eliminar un estudio realizado. |

**Tabla 3. 10. Descripción de la clase c_ecocardiogramarpt**

| Nombre:m_ecocardiogramarpt | |
|---|---|
| **Tipo de Clase: Controladora** | |
| **Por cada responsabilidad:** | |
| Nombre: | Registrar (E_EcocardiogramaRPT $estudio) |
| Descripción | Registra el estudio realizado al paciente. |
| Nombre: | getById ($id) |
| Descripción | Devuelve el Estudio del paciente dado el ID. |
| Nombre: | Listar (EstructuraEntrada $entrada, $pacienteRPT) |
| Descripción | Muestra un listado con los estudios o el estudio realizado al paciente. |
| Nombre: | Eliminar ($id) |
| Descripción | Permite eliminar un estudio realizado. |
| Nombre: | Editar (E_EcocardiogramaRPT $estudio) |

| | |
|---|---|
| Descripción | Cambia el Estudio realizado al paciente, si no han pasado 48 horas. |
| Nombre: | InsertarDiagnosticos (E_EcocardiogramaRPT $estudio) |
| Descripción | Permite insertar los diagnósticos realizados al paciente por el medico. |

**Tabla 3. 11. Descripción de la clase m_ecocardiogramarpt.**

| **Nombre:e_estructurasalidaestudios** | |
|---|---|
| **Tipo de Clase: Entidad** | |
| Atributo | Tipo |
| $Lista = array (); | var |
| Descripción | Muestra un arreglo con los estudios del paciente. Esta entidad se encuentra en todos los estudios que se realizan. |

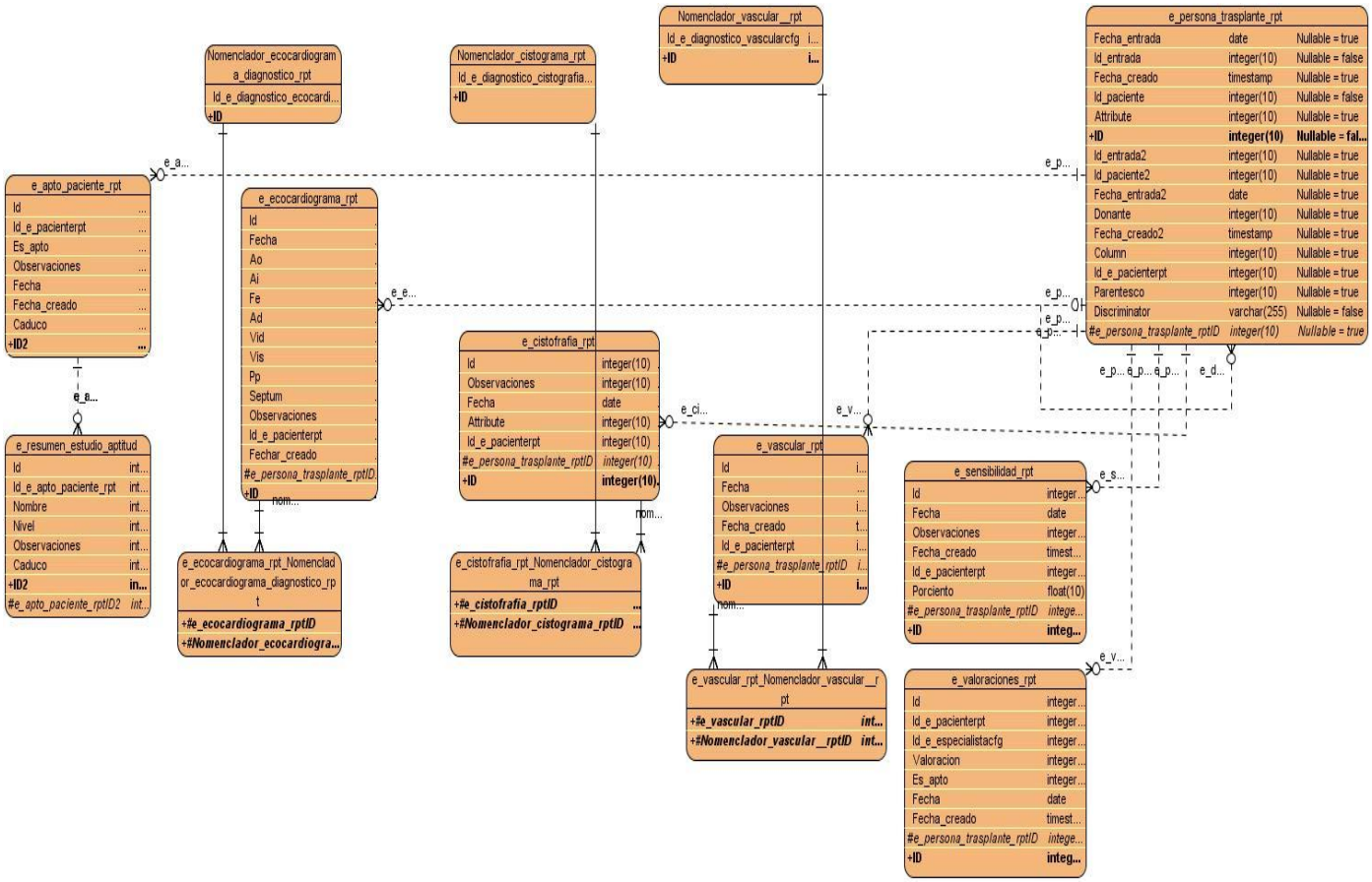
**Tabla 3. 12. Descripción de la clase e_estructurasalidaestudios**

| Nombre:e_ecocardiogramarpt | |
|---|---|
| **Tipo de Clase: Entidad** | |
| Atributo | Tipo |
| **$fecha** | var |
| $observaciones; | var |
| $ao; | var |
| $ai; | var |
| $ad; | var |
| $vid; | var |
| $vis; | var |
| $fe; | var |
| $pp; | var |
| $septum; | var |
| $diagnósticos = array (); | var |
| $fecha_creado; | var |
| $id_e_pacienterpt; | var |
| Descripción | Contiene todos los datos que se encuentran en la tabla de la base de datos. La entidad va a |

| | |
|---|---|
| | contener los datos según el estudio que se le realiza al paciente. |

**Tabla 3. 13. Descripción de la clase e_ecocardiogramarpt.**

## 3.3 Modelo de datos.



## 3.4 Descripción de las tablas

| e_paciente_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id_entrada | int | Identificador de entrada del paciente en RPN. |
| id_paciente | int | Identificador del paciente. |
| donante | enum | Enum que controla si es donante vivo o muerto. |

**Tabla 3. 14. Descripción de la Tabla e_paciente_rpt.**

| e_cistografía_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| observaciones | varchar | Observaciones del estudio. |
| fecha | date | Fecha en que se registró el estudio. |
| fecha_creado | TIMESTAMP | Fecha en que se registró el estudio en el sistema. |
| id_e_pacienterpt | int | Llave foránea, que permite saber dado un estudio a que paciente corresponde. |

**Tabla 3. 15. Descripción de la Tabla e_cistografía_rpt.**

| e_sensibilidad_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| fecha | date | Fecha en que se registró el estudio |
| observaciones | varchar | Observaciones del estudio |
| fecha_creado | TIMESTAMP | Fecha en que se registró el estudio en el sistema. |
| id_e_pacienterpt | int | Llave foránea, que permite saber dado un estudio a que paciente corresponde. |
| porciento | float | Controla el porciento del estudio. |

**Tabla 3. 16. Descripción de la Tabla e_sensibilidad_rpt.**

| e_vascular_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| fecha | date | Fecha en que se registró el estudio. |
| observaciones | varchar | Observaciones del estudio. |
| fecha_creado | TIMESTAMP | Fecha en que se registró el estudio en el sistema. |
| Id_e_persona_trasplanterpt | int | Llave foránea, que permite saber dado un estudio a que paciente corresponde. |

**Tabla 3. 17. Descripción de la Tabla e_vascular_rpt**

| e_ecocardiograma_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| fecha | date | Fecha en que se registró el estudio. |
| ao | float | Examen que se le realiza al paciente. |
| ai | float | Examen que se le realiza al paciente. |
| ad | float | Examen que se le realiza al paciente. |
| vid | float | Examen que se le realiza al paciente. |
| fe | float | Examen que se le realiza al paciente. |
| vis | float | Examen que se le realiza al paciente. |
| pp | float | Examen que se le realiza al paciente. |
| septum | float | Examen que se le realiza al paciente. |
| observaciones | varchar | Observaciones del estudio. |
| id_e_pacienterpt | int | Llave foránea, que permite saber dado un estudio a que paciente corresponde. |
| fecha_creado | TIMESTAMP | Fecha en que se registró el estudio en el sistema. |
| es_apto | TINYINT | Estado de aptitud del paciente. |

**Tabla 3. 18. Descripción de la Tabla e_ecocardiograma_rpt.**

| e_apto_paciente_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| id_e_pacienterpt | int | Llave foránea, que permite saber dado un estudio a que paciente corresponde. |
| es_apto | TINYINT | Estado de aptitud del paciente. |
| observaciones | varchar | Observaciones del Estado de Aptitud del paciente. |
| fecha | date | Fecha en que se registró el estado de aptitud del paciente. |
| fecha_creado | TIMESTAMP | Fecha en que se registró el estado de aptitud del paciente en el sistema. |
| caduco | TINYINT | Permite desechar el estado de aptitud del paciente, si modifico algún estudio. |

**Tabla 3. 19. Descripción de la Tabla e_apto_paciente_rpt.**

| e_resumen_estudio_aptitud_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| id_e_apto_paciente_rpt | int | Llave foránea, que permite ver el resumen del estado de aptitud de un paciente determinado. |
| nombre | varchar | Nombre del estudio realizado. |
| nivel | enum | Controla en estado de aptitud del paciente: Apto, NoApto y AptoConDificultad. |
| observaciones | varchar | Observaciones del Estado de Aptitud del paciente. |
| caduco | TINYINT | Permite desechar el estado de aptitud del paciente, si modifico algún estudio. |

**Tabla 3. 20. Descripción de la Tabla e_resumen_estudio_aptitud_rpt**

| e_valoraciones_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| id_e_pacienterpt | int | Llave foránea, que permite ver la valoración de un paciente determinado. |
| id_e_especialistacfg | int | Identificador del especialista que hizo la valoración. |
| valoración | varchar | Valoración del paciente. |
| es_apto | TINYINT | Estado de aptitud del paciente. |
| fecha | date | Fecha en que se registró la valoración. |
| fecha_creado | TIMESTAMP | Fecha en que se registró la valoración en el sistema. |

**Tabla 3. 21. Descripción de la Tabla e_valoraciones_rpt**

| er_vascular_diagnóstico_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| id_e_vascularrpt | int | Identificador del estudio. |
| id_e_diagnostico_vascularcfg | int | Identificador del nomenclador. |

**Tabla 3. 22. Descripción de la Tabla er_vascular_diagnóstico_rpt.**

| er_cistografía_diagnóstico_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| id_e_cistografiarpt | int | Identificador del estudio. |
| Id_e_diagnostico_cistografiacfg | int | Identificador del nomenclador. |

**Tabla 3. 23. Descripción de la Tabla er_cistografía_diagnóstico_rpt.**

| er_ecocardiograma_diagnóstico_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id | int | Identificador de la tabla. |
| id_e_ecocardiogramarpt | int | Identificador del estudio. |
| Id_e_diagnostico_ecocardiogramacfg | int | Identificador del nomenclador. |

**Tabla 3. 24. Descripción de la Tabla er_ecocardiograma_diagnóstico_rpt.**

| e_persona_trasplante_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| fecha_entrada | date | Fecha que se registró la persona. |
| id_entrada | int | Identificador de entrada del paciente en RPN. |
| id_paciente | int | Identificador del paciente. |
| fecha_creado | TIMESTAMP | Fecha en que se registró la persona en el sistema. |

**Tabla 3. 25. Descripción de la Tabla e_persona_trasplante_rpt**

| e_donante_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id_entrada | int | Identificador de entrada del paciente en RPN. |
| id_e_pacienterpt | int | Identificador del pacienteRPT. |
| parentesco | enum | Contiene si el parentesco es padres, hermanos o hijos. |

**Tabla 3. 26. Descripción de la tabla e_donante_rpt**

| e_persona_trasplante_historico_rpt | | |
|---|---|---|
| Atributo | Tipo | Descripción |
| id_entrada | int | Identificador de entrada del paciente en RPN. |
| fecha_salida | date | Fecha de salida del paciente. |
| observaciones | varchar | Observaciones realizadas al paciente. |
| causa_salida | int | Causa por la cual el paciente se le da salida. |

**Tabla 3. 27 Descripción de la tabla e_persona_trasplante_historico_rpt**

## 3.5 Vista de Implementación

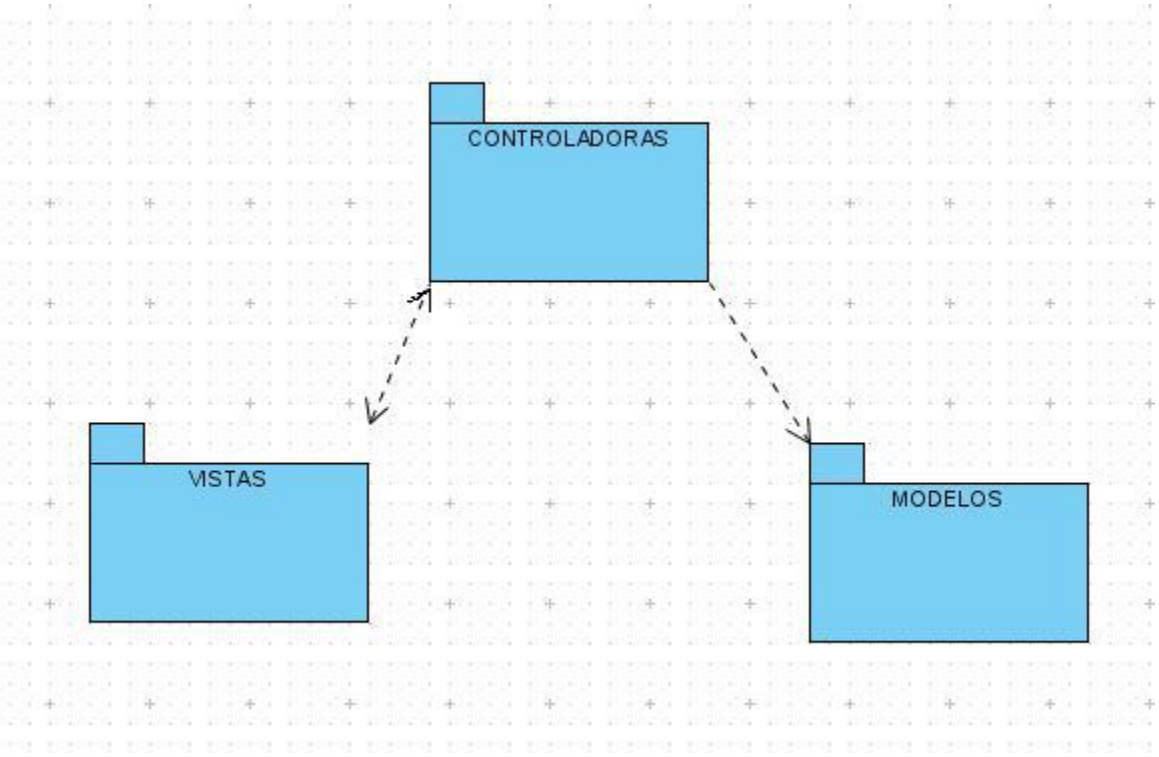



**Fig. 3. 1. Diagrama de Paquetes**

### 3.5.1 Diagramas de Componentes

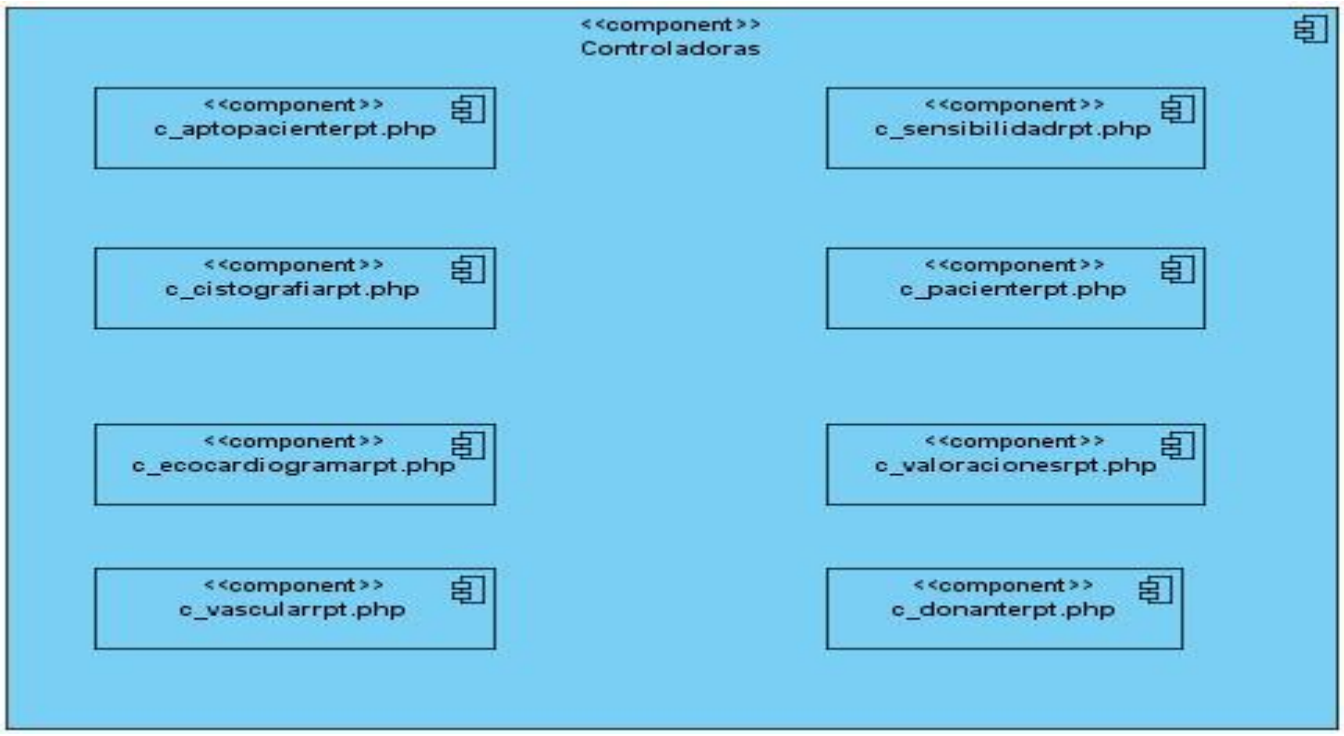



**Fig. 3. 2. Diagrama de Componentes de las Controladoras**

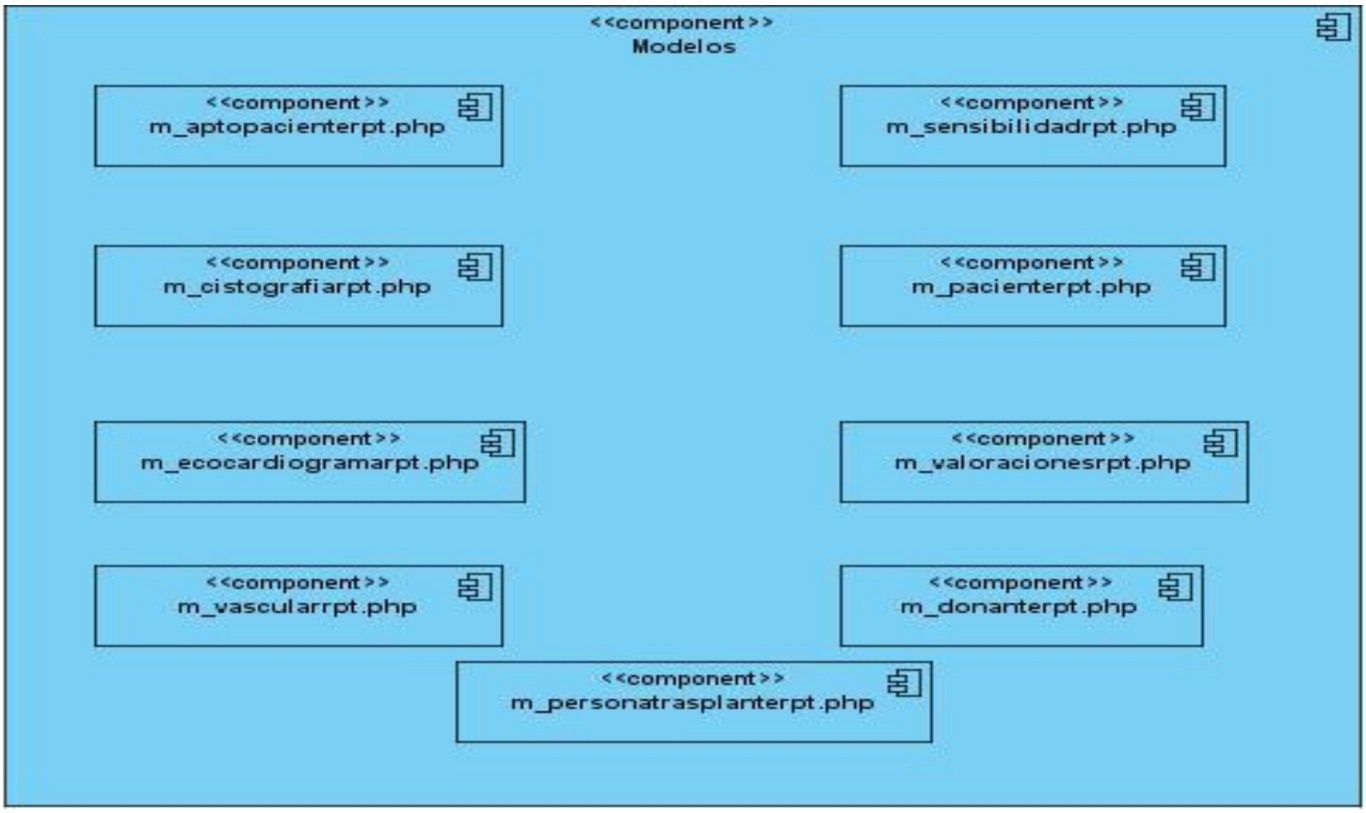



**Fig. 3. 3. Diagrama de Componentes de las Modelos**

Diagramas de componentes de las vistas por caso de uso

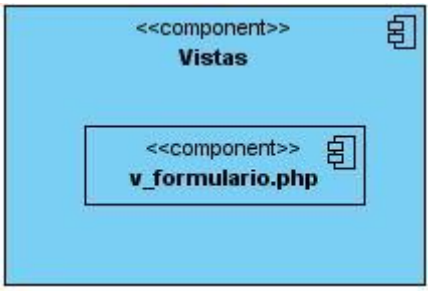



**Fig. 3.4. 1. Diagrama de componentes de las vistas del caso de uso Registrar**

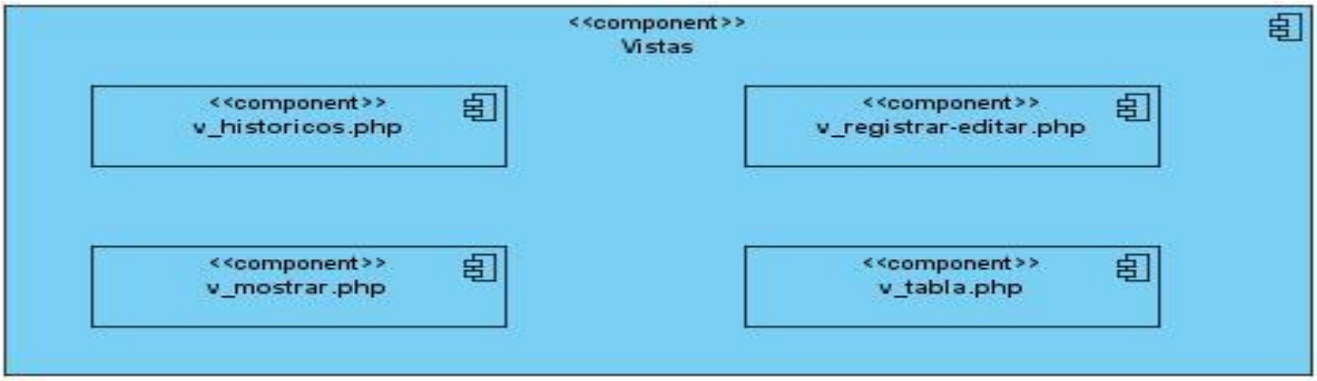



**Fig. 3.4. 2. Diagrama de componentes de las vistas del caso de uso Estado de Aptitud.**

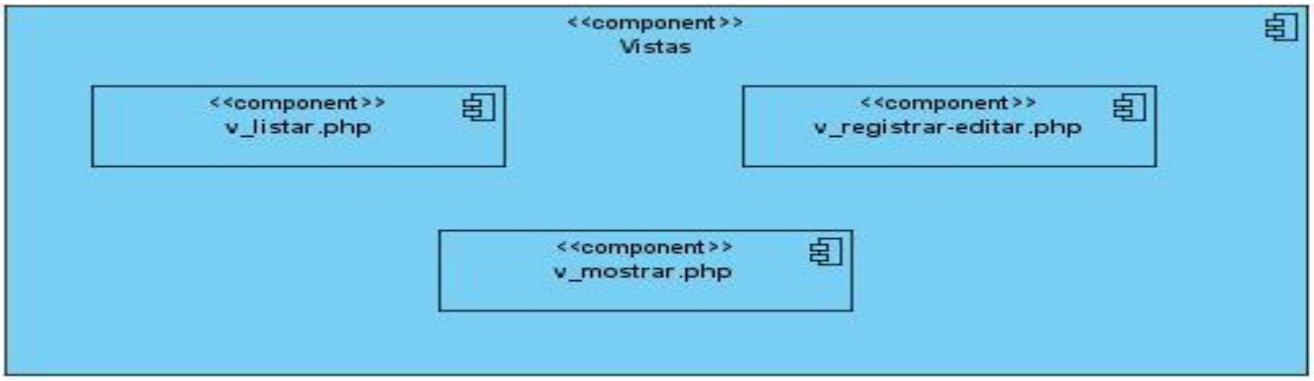



**Fig. 3.4. 3. Diagrama de componentes de las vistas del caso de uso Gestionar Estudios**

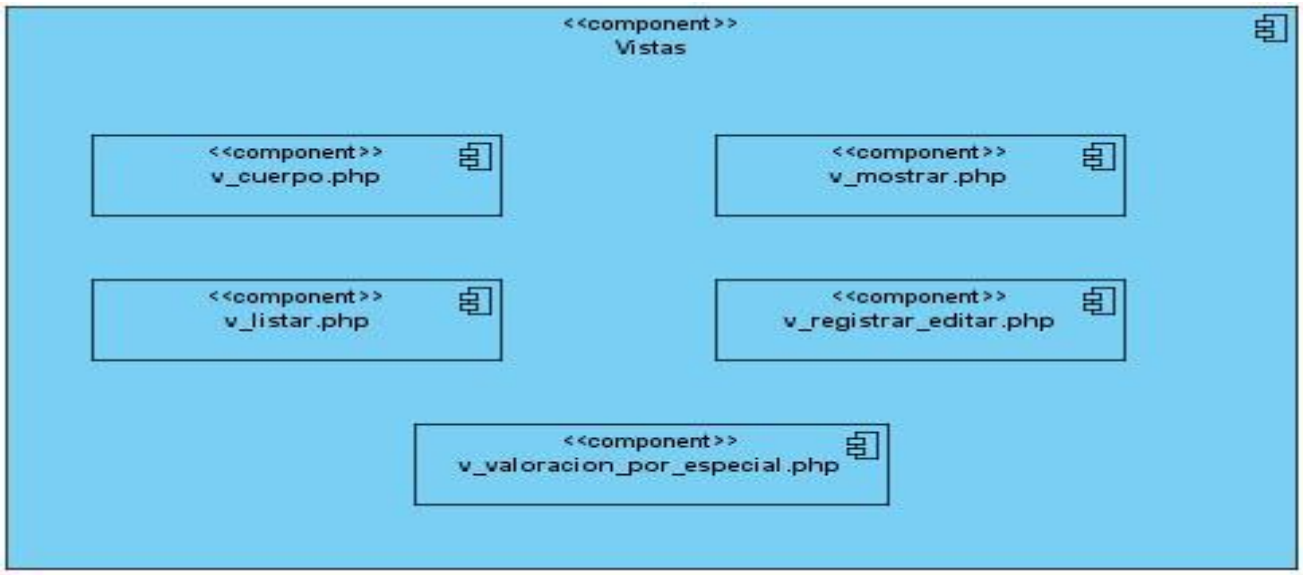



**Fig. 3.4. 4. Diagrama de componentes de las vistas del caso de uso Valoraciones**

# Conclusiones

La realización del presente capítulo se centró en la descripción y análisis de la solución propuesta, donde se analizó la valoración del diseño propuesto por el analista y se establecieron las clases u operaciones necesarias para el buen funcionamiento de la aplicación y las técnicas de validación establecidas para la integridad y normalización de la base de datos. Mediante esquemas se mostraron los Modelo de Datos y la Vista de Implementación con sus correspondientes Diagramas de Componentes.

## Conclusiones

Una vez concluida la investigación, se dio cumplimiento al objetivo y las tareas planteadas, obteniéndose los siguientes resultados:

- Se realizó la implementación de los módulos Gestión de Reportes y Registro de trasplantes utilizando los patrones de diseños establecidos y el patrón observador.
- Se obtuvo una interfaz gráfica del sistema orientada al usuario, agradable y fácil de operar.

- Se definió como lenguaje de programación PHP en su versión 5.0 respetando las políticas establecidas por la Facultad y la Universidad. Se definió usar como patrones de arquitectura Modelo-Vista-Controlador y Tres Capas.

Con la implementación de estos módulos y el funcionamiento del sistema se logrará el incremento de la capacidad organizativa de los centros de diálisis del país. El trabajo de estos, se realizará más fácil y rápido, para los médicos y demás personas que trabajan para mejorar la calidad de vida de estos pacientes, lo cual contribuye al aumento de la calidad de la asistencia médica a este tipo de Pacientes.

## Recomendaciones

Luego de la presentación del estudio realizado, que culmina con la implementación de los módulos Gestión de Reportes y Registro de Trasplantes, perteneciente al producto Alas NefroRed, se listan a continuación una serie de recomendaciones para la ampliación, modificación, mejora y construcción de nuevas versiones de este sistema:

- Mantener sobre el sistema un estricto cumplimiento del proceso de actualización periódica en un período de 6 meses como máximo logrando así que se mantenga la fiabilidad y funcionamiento óptimo del sistema y de la información que se gestiona a través de él.
- Continuar el desarrollo del módulo Registro de Trasplantes.
- Impartir cursos de capacitación a las personas que laboran en los servicios nefrológicos y trabajarán con la aplicación.
- Implementar para próximas versiones del sistema otros reportes de importancia para el trabajo en los servicios nefrológicos, tal es el caso de los reportes: reuso de dializadores, trasplantes exitosos o no, complicaciones intra-dialíticas, accesos en reuso y el reporte de métodos dialíticos.
- Implementar un nuevo módulo que se encargue de dar seguimiento a los pacientes trasplantados.
- Un componente para tratar la compatibilidad donante-receptor. A partir de las características de un órgano (riñón), el sistema debe ser capaz de seleccionar el receptor idóneo.

# Recomendaciones

- Implementar un componente para la generación de alertas, tanto a nivel de historia clínica como a nivel de centro de diálisis.

Ejemplos:

a) Alerta cuando se abra la historia clínica de un paciente que tiene atrasados algunos de los estudios que debe realizarse un paciente para ser trasplantado.

b) Alerta cuando el % de mortalidad del centro de diálisis en el mes exceda los valores normales.

## Referencias Bibliográficas:

1-Gonzalez, Lic. María del Carmen. 1999. Biblioteca Médica Nacional. [En línea] 1999. [Citado el: 10 de Noviembre de 2008.] http://www.sld.cu/sitios/bmn/.

2-Buch López, Dr. Abelardo;2007. DIA MUNDIAL DEL RIÑON Y NEFROLOGIA EN CUBA. [En línea] 2007. [Citado el: 12 de Noviembre de 2008.] http://www.ucmh.sld.cu/rhab/rhcm_vol_6num_2/rhcm19207.htm.

3-Internet. [En línea] [Citado el: 6 de Diciembre de 2008.] http://coepa.info/intro_internet/origen_de_internet/index.php.

4-Internet. [En línea] [Citado el: 5 de Diciembre de 2008.] http://coepa.info/intro_internet/servicios_de_la_red_internet/index.php.

5-Ventajas Páginas Web. [En línea] [Citado el: 6 de Diciembre de 2008.] http://www.dimagin.net/es/contenido.php?t_id=6.

6-Desventajas Páginas Web. [En línea] [Citado el: 6 de Diciembre de 2008.] http://www.avidos.net/blogold/aplicaciones-web.

7-PHP. [En línea] [Citado el: 6 de Diciembre de 2008.] http://www.masadelante.com/faq-php.htm.

8-Ventajas PHP. [En línea] [Citado el: 7 de Diciembre de 2008.] http://skindario.com/768851/4098456-ventajas-del-php/.

9-Desventajas de PHP. [En línea] [Citado el: 7 de Diciembre de 2008.] http://www.forosdelweb.com/f18/ventajas-desventajas-php-jsp-498570/.

10-HTML. [En línea] [Citado el: 25 de Enero de 2009.] https://belenus.unirioja.es/~guprado/pagweb/carachtml.html

11-Caracteristicas HTML. [En línea] [Citado el: 25 de Enero de 2009.] http://www.hooping.net/faq-caracteristicas.aspx#formato

12-JavaScript. [En línea] [Citado el: 26 de Enero de 2009.] http://www.librosweb.es/javascript/.

13-JSP. [En línea] [Citado el: 26 de Enero de 2009.] http://www.desarrolloweb.com/articulos/831.php.

14-Sistema Gestores Base Datos. [En línea] [Citado el: 10 de Diciembre de 2008.] http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base.

# Referencias Bibliográficas

15-SGDB. [En línea] [Citado el: 10 de Diciembre de 2008.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.

16-PostgreSQL. [En línea] [Citado el: 11 de Diciembre de 2008.] http://es.tldp.org/Postgresql-es/web/navegable/tutorial/x56.html.

17-PostgreSQL1. [En línea] [Citado el: 11 de Diciembre de 2008.] http://es.tldp.org/Postgresql-es/web/navegable/tutorial/intro.html#AEN34.

18-Características de PostgreSQL. [En línea] [Citado el: 11 de Diciembre de 2008.] http://www.netpecos.org/docs/mysql_postgres/x15.html

19-SQL. [En línea] [Citado el: 26 de Enero de 2009.] http://www.maestrosdelweb.com/editorial/tutsql1

20-MySQL. [En línea] [Citado el: 12 de Diciembre de 2008.] http://www.netpecos.org/docs/mysql_postgres/x57.html

21-Características de MySQL. [En línea] [Citado el: 12 de Diciembre de 2008.]

http://www.somoslibres.org/modules.php?name=News&file=article&sid=788

22-IDE Programación. [En línea] [Citado el: 13 de Diciembre de 2008.] http://elcodigok.blogspot.com/2007/09/que-son-los-ide-de-programacin.html.

23-Zend Studio. [En línea] [Citado el: 7 de Enero de 2009.] http://www.maestrosdelweb.com/editorial/zendstudio/.

24-Servicios Web. [En línea] [Citado el: 7 de Enero de 2009.] http://homepages.mty.itesm.mx/al450951/.

25-Características Servicios Web. [En línea] [Citado el: 7 de Enero de 2009.] http://www.desarrolloweb.com/articulos/1852.php.

26-Apache. [En línea] [Citado el: 8 de Enero de 2009.] http://acsblog.es/articulos/trunk/LinuxActual/Apache/html/x31.html

27-Framework. [En línea] [Citado el: 8 de Enero de 2009.] http://secuoyas.com/blog/2008/11/27/cms-y-framework-la-importancia-de-un-glosario-en-un-equipo-de-d.

28-Symfony. El Frameworks Symfony, una introducción práctica. [En línea] [Citado el: 9 de Enero de 2009.] http://www.maestrosdelweb.com/editorial/el-framework-symfony-una-introduccion-practica-i-parte/.

29-Symfony Framework. [En línea] [Citado el: 9 de Enero de 2009.] http://geekymty.blogspot.com/2007/03/symfony-framework.html.

# Referencias Bibliográficas

30-Symfony Characteristics. [En línea] [Citado el: 10 de Enero de 2009.] http://www.librosweb.es/symfony/capitulo1/symfony_en_pocas_palabras.html.

31-CodeIgniter. [En línea] [Citado el: 10 de Enero de 2009.] http://xyox.info/2008/10/27/codeigniter-framework-php-introduccion/.

32-YUI. [En línea] [Citado el: 11 de Enero de 2009.] http://www.elwebmaster.com/articulos/top-5-javascript-frameworks.

33-Modelo Tres Capas. [En línea] [Citado el: 22 de Enero de 2009.] http://www.fing.edu.uy/inco/grupos/coal/investigacion/publicaciones/vp01.pdf.

34-Características de Arq. Componentes. [En línea] [Citado el: 24 de Enero de 2009.] http://www.monografias.com/trabajos14/aplicacion-distrib/aplicacion-distrib.shtml.

35-2009**.** *Informática 2009.* Habana : s.n., 2009.

36-2009**.** *Tesis Karel Gómez.* Habana : s.n., 2009.

37-2009**.** *Manual de PEAR.* Habana : s.n., 2009.

38-PEAR_Concepto.Habana. [En línea] [Citado el: 10 de Marzo de 2009.] http://dotpress.wordpress.com/2007/03/29/pear-estandares-de-desarrollo-para-php/.

## Bibliografía

Apache. [En línea] [Citado el: 8 de Enero de 2009.]
http://acsblog.es/articulos/trunk/LinuxActual/Apache/html/x31.html

Buch López, Dr. Abelardo;2007. DIA MUNDIAL DEL RIÑON Y NEFROLOGIA EN CUBA. [En línea] 2007. [Citado el: 12 de Noviembre de 2008.] http://www.ucmh.sld.cu/rhab/rhcm_vol_6num_2/rhcm19207.htm.

Características HTML. [En línea] [Citado el: 25 de Enero de 2009.] http://www.hooping.net/faq-caracteristicas.aspx#formato

Características de PostgreSQL. [En línea] [Citado el: 11 de Diciembre de 2008.]
http://www.netpecos.org/docs/mysql_postgres/x15.html

Características de MySQL. [En línea] [Citado el: 12 de Diciembre de 2008.]

http://www.somoslibres.org/modules.php?name=News&file=article&sid=788

Características Servicios Web. [En línea] [Citado el: 7 de Enero de 2009.]
http://www.desarrolloweb.com/articulos/1852.php.

CodeIgniter. [En línea] [Citado el: 10 de Enero de 2009.] http://xyox.info/2008/10/27/codeigniter-framework-php-introduccion/.

Características de Arq. Componentes. [En línea] [Citado el: 24 de Enero de 2009.]
http://www.monografias.com/trabajos14/aplicacion-distrib/aplicacion-distrib.shtml.

Desventajas de PHP. [En línea] [Citado el: 7 de Diciembre de 2008.] http://www.forosdelweb.com/f18/ventajas-desventajas-php-jsp-498570/.

Desventajas Páginas Web. [En línea] [Citado el: 6 de Diciembre de 2008.]
http://www.avidos.net/blogold/aplicaciones-web.

Framework. [En línea] [Citado el: 8 de Enero de 2009.] http://secuoyas.com/blog/2008/11/27/cms-y-framework-la-importancia-de-un-glosario-en-un-equipo-de-d.

Gonzalez, Lic. María del Carmen. 1999. Biblioteca Médica Nacional. [En línea] 1999. [Citado el: 10 de Noviembre de 2008.] http://www.sld.cu/sitios/bmn/.

HTML. [En línea] [Citado el: 25 de Enero de 2009.] https://belenus.unirioja.es/~guprado/pagweb/carachtml.html

Internet. [En línea] [Citado el: 6 de Diciembre de 2008.]
http://coepa.info/intro_internet/origen_de_internet/index.php.

# Bibliografía


Internet. [En línea] [Citado el: 5 de Diciembre de 2008.] http://coepa.info/intro_internet/servicios_de_la_red_internet/index.php.

IDE Programación. [En línea] [Citado el: 13 de Diciembre de 2008.] http://elcodigok.blogspot.com/2007/09/que-son-los-ide-de-programacin.html.

JSP. [En línea] [Citado el: 26 de Enero de 2009.] http://www.desarrolloweb.com/articulos/831.php.

JavaScript. [En línea] [Citado el: 26 de Enero de 2009.] http://www.librosweb.es/javascript/.

Modelo Tres Capas. [En línea] [Citado el: 22 de Enero de 2009.] http://www.fing.edu.uy/inco/grupos/coal/investigacion/publicaciones/vp01.pdf.

MySQL. [En línea] [Citado el: 12 de Diciembre de 2008.] http://www.netpecos.org/docs/mysql_postgres/x57.html

PHP. [En línea] [Citado el: 6 de Diciembre de 2008.] http://www.masadelante.com/faq-php.htm.

PostgreSQL. [En línea] [Citado el: 11 de Diciembre de 2008.] http://es.tldp.org/Postgresql-es/web/navegable/tutorial/x56.html.

PostgreSQL1. [En línea] [Citado el: 11 de Diciembre de 2008.] http://es.tldp.org/Postgresql-es/web/navegable/tutorial/intro.html#AEN34.

PEAR_Concepto.Habana. [En línea] [Citado el: 10 de Marzo de 2009.] http://dotpress.wordpress.com/2007/03/29/pear-estandares-de-desarrollo-para-php/.

Sistema Gestores Base Datos. [En línea] [Citado el: 10 de Diciembre de 2008.] http://www.error500.net/garbagecollector/archives/categorias/bases_de_datos/sistema_gestor_de_base.

SGDB. [En línea] [Citado el: 10 de Diciembre de 2008.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.

SQL. [En línea] [Citado el: 26 de Enero de 2009.]

http://www.maestrosdelweb.com/editorial/tutsql1

Servicios Web. [En línea] [Citado el: 7 de Enero de 2009.] http://homepages.mty.itesm.mx/al450951/.

Symfony. El Frameworks Symfony, una introducción práctica. [En línea] [Citado el: 9 de Enero de 2009.] http://www.maestrosdelweb.com/editorial/el-framework-symfony-una-introduccion-practica-i-parte/.

Symfony Framework. [En línea] [Citado el: 9 de Enero de 2009.] http://geekymty.blogspot.com/2007/03/symfony-framework.html.

# Bibliografía

Symfony Characteristics. [En línea] [Citado el: 10 de Enero de 2009.] http://www.librosweb.es/symfony/capitulo1/symfony_en_pocas_palabras.html.

YUI. [En línea] [Citado el: 11 de Enero de 2009.] http://www.elwebmaster.com/articulos/top-5-javascript-frameworks.

Zend Studio. [En línea] [Citado el: 7 de Enero de 2009.] http://www.maestrosdelweb.com/editorial/zendstudio/.

2009**.** *Informática 2009.* Habana : s.n., 2009.

2009**.** *Tesis Karel Gómez.* Habana : s.n., 2009.

2009**.** *Manual de PEAR.* Habana : s.n., 2009.

## Glosario de Términos

Diálisis: Es un procedimiento que se realiza para retirar los elementos tóxicos (impurezas o desechos) de la sangre cuando los riñones no pueden hacerlo. La diálisis se puede llevar a cabo usando diferentes métodos: diálisis peritoneal y hemodiálisis.

Enfermedad Renal Crónica: Es una enfermedad que se presenta cuando los riñones ya no pueden funcionar al nivel necesario para la vida diaria. Este padecimiento se presenta a medida que la insuficiencia renal crónica progresa a tal punto en que la capacidad de los riñones para excretar los desechos, concentrar la orina y regular los electrolitos es menos del 10% de su capacidad normal.

INFOMED: Nombre que identifica a la primera red electrónica cubana de información para la salud y surgió como parte de un proyecto del Centro Nacional de Información de Ciencias Médicas de Cuba. INFOMED es el Portal de Salud Cubano y la red de personas e instituciones que comparten el propósito de facilitar el acceso a la información de salud en Cuba.

Nefrología: Es la especialidad médica, rama de la medicina interna, que se ocupa del estudio de la estructura y la función renal, tanto en la salud como en la enfermedad, incluyendo la prevención y tratamiento de las enfermedades renales.

Reportes: Se caracteriza por contener información de una u otra materia reflejando el resultado de una investigación adaptado al contexto de una situación.

# Glosario de Términos

Sistema Informático: Un sistema informático como todo sistema, es el conjunto de partes interrelacionadas, hardware, software y de recueros humanos (humanware). Un sistema informático típico emplea una computadora que usa dispositivos programables para capturar, almacenar y procesar datos.

SISalud: Empresa que presta servicio médico quirúrgico ambulatorio y semi-ambulatorio de amplia cobertura, con excelente calidad de atención. Brinda diferentes atenciones médicas en diversas especialidades, además contaran con la más amplia atención de un selecto grupo de profesionales de la salud, utilizando la más moderna tecnología.

Trasplantes: Es un tratamiento médico complejo. Permite que órganos, tejidos y células de una persona puedan reemplazar órganos, tejidos o células enfermos de otra persona. En algunos casos esta acción sirve para salvarle la vida, en otros para mejorar la calidad de vida o ambas cosas.

## Anexo I

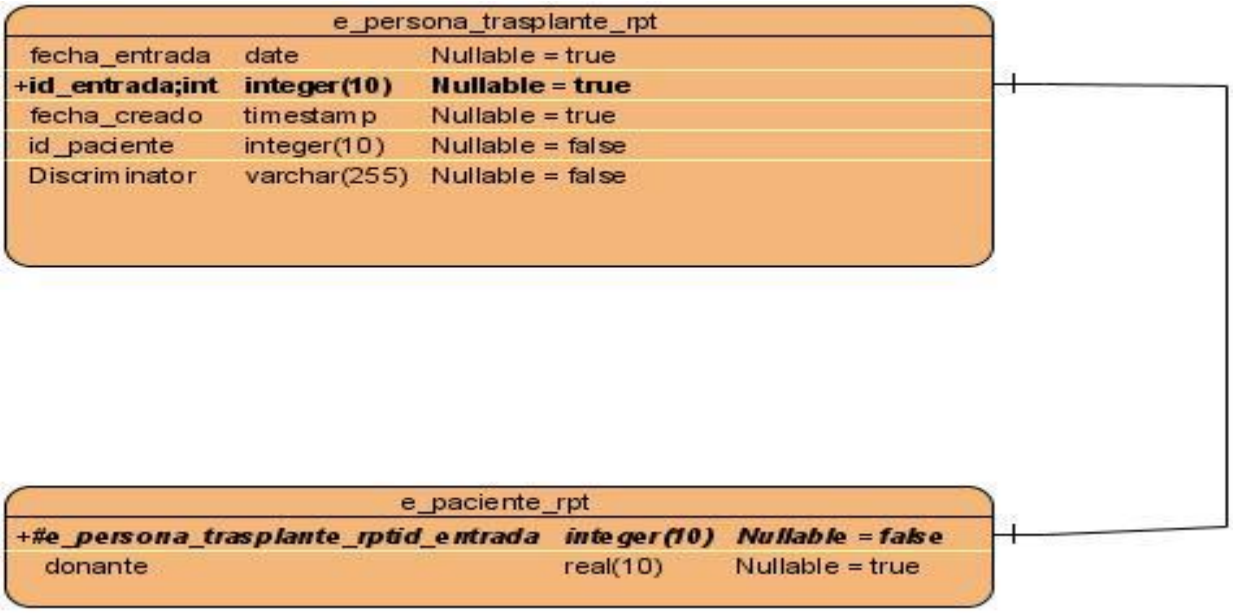



**Fig. 3.4. 5 Modelo de Datos: Caso de Uso Registrar PacienteRPT**

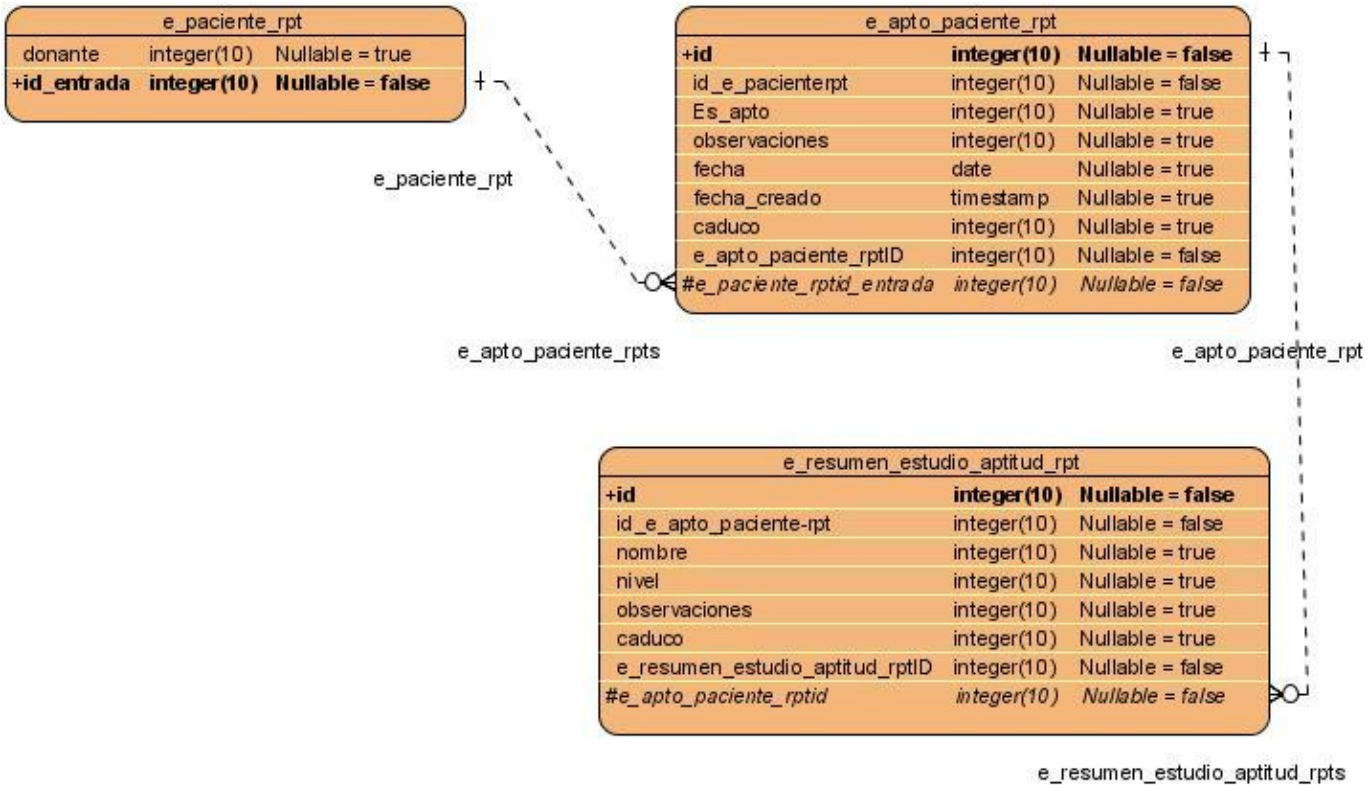



**Fig. 3.4. 6 Modelo de Datos: Caso de Uso Gestionar Estado de Aptitud**

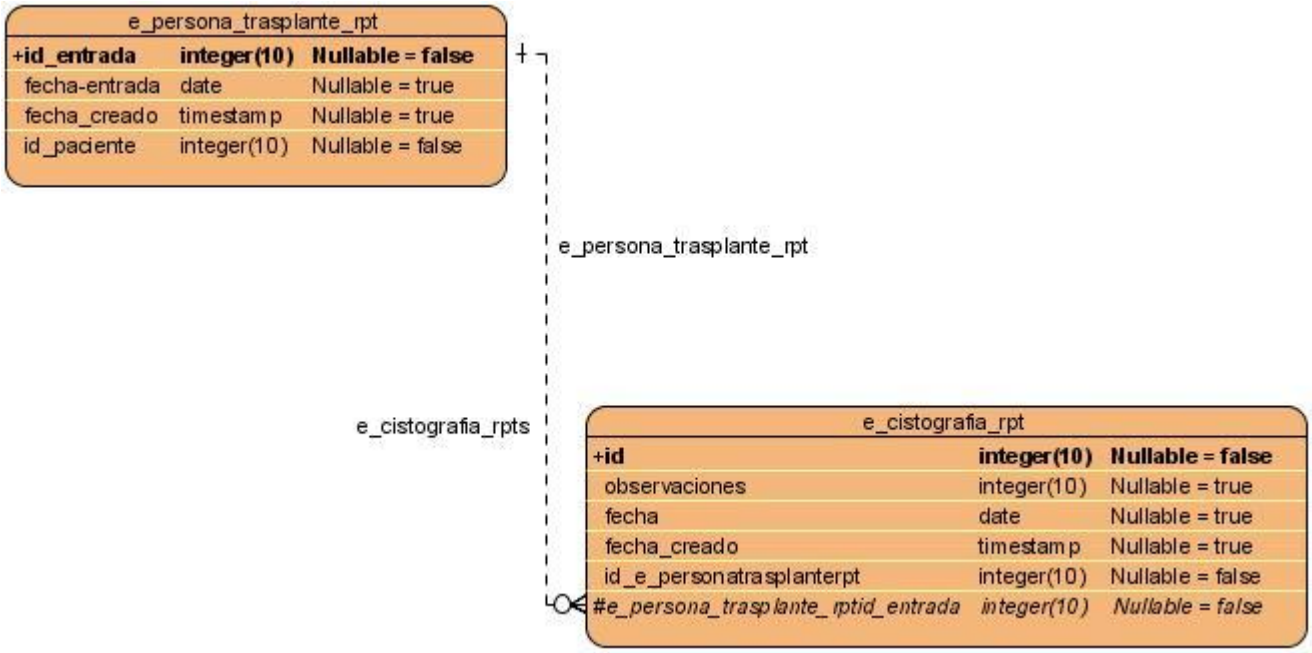



**Fig. 3.4. 7 Modelo de Datos: Caso de Uso Gestionar Estudios**

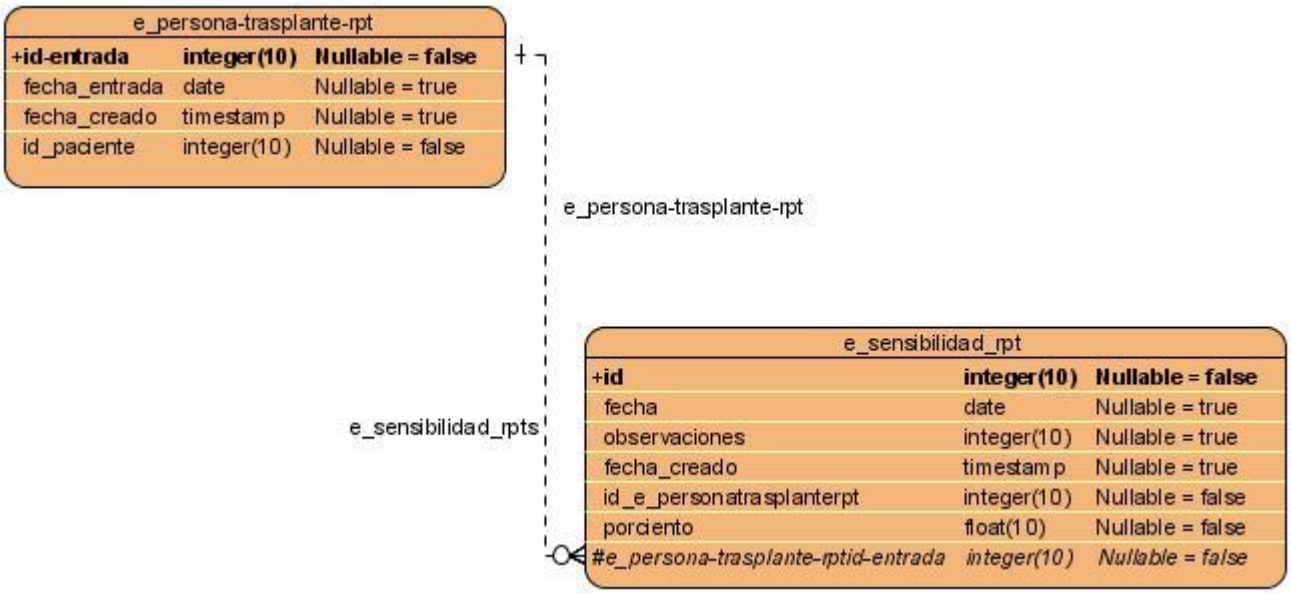



**Fig. 3.4. 8 Modelo de Datos: Caso de Uso Gestionar Estudios**

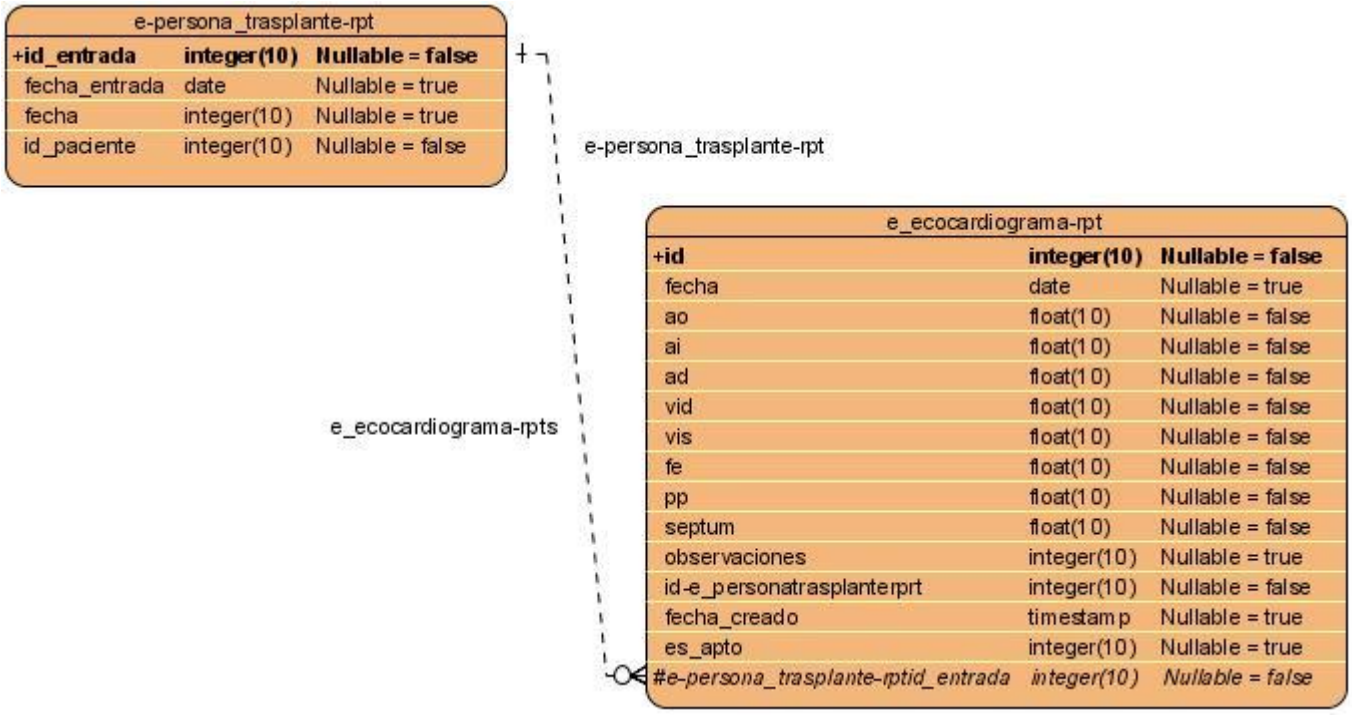



**Fig. 3.4. 9 Modelo de Datos: Caso de Uso Gestionar Estudios**

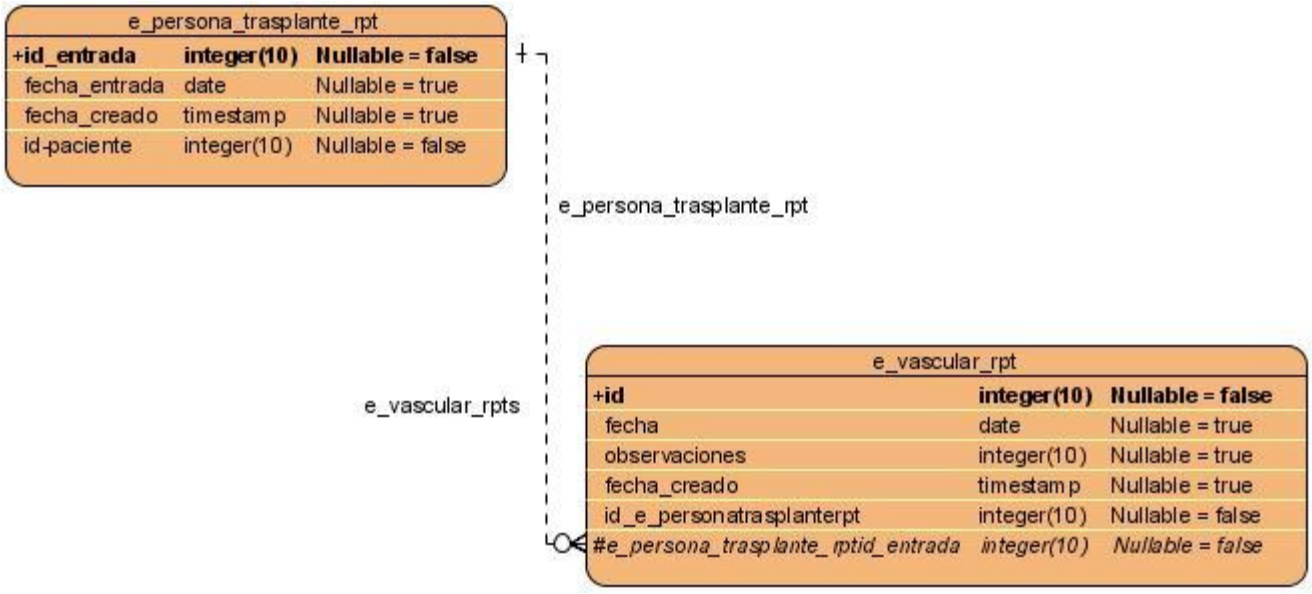



**Fig. 3.4. 10 Modelo de Datos: Caso de Uso Gestionar Estudios**

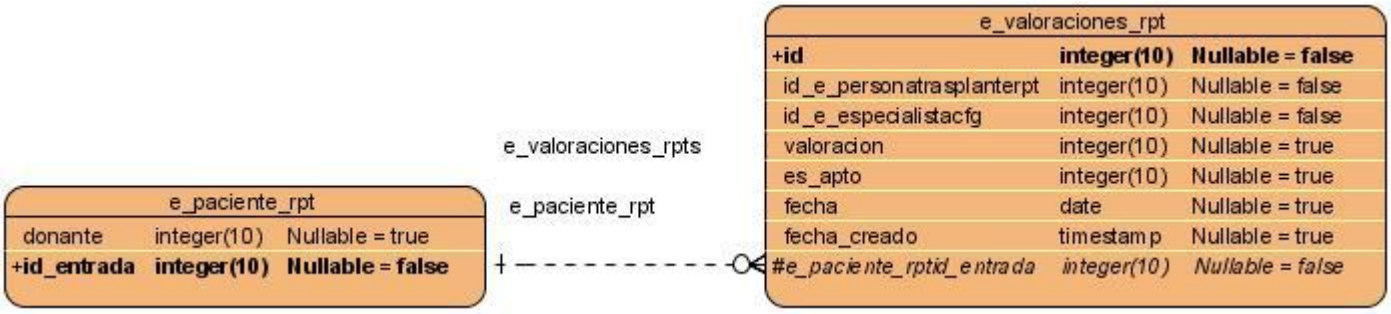



**Fig. 3.4. 11 Modelo de Datos: Caso de Uso Valoraciones**

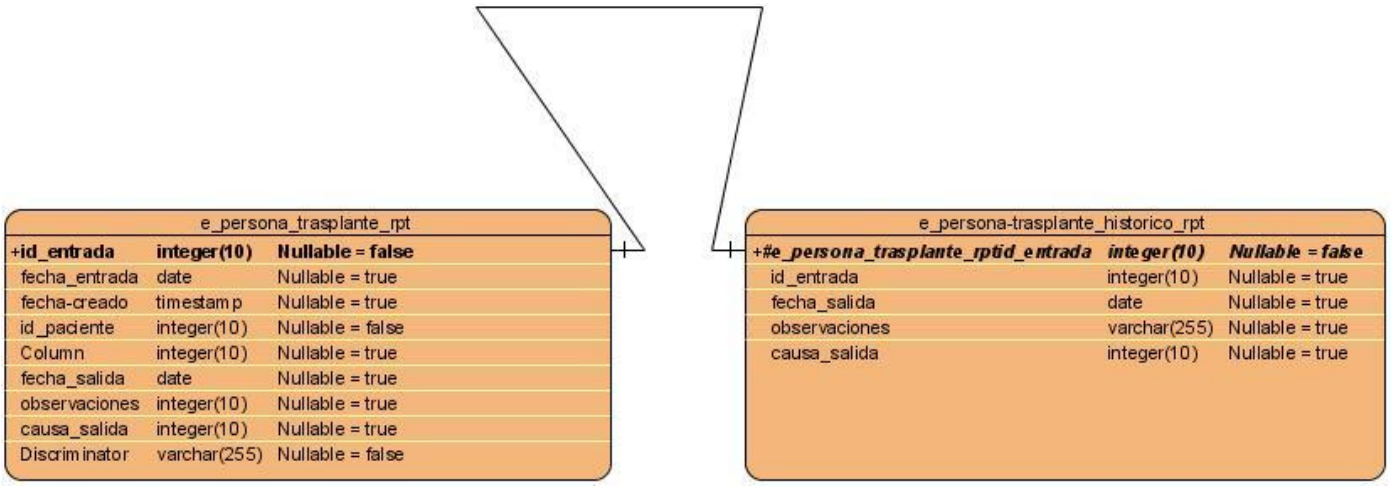



**Fig. 3.4. 12 Modelo de Datos: Caso de Uso Gestionar Donante de RPT**

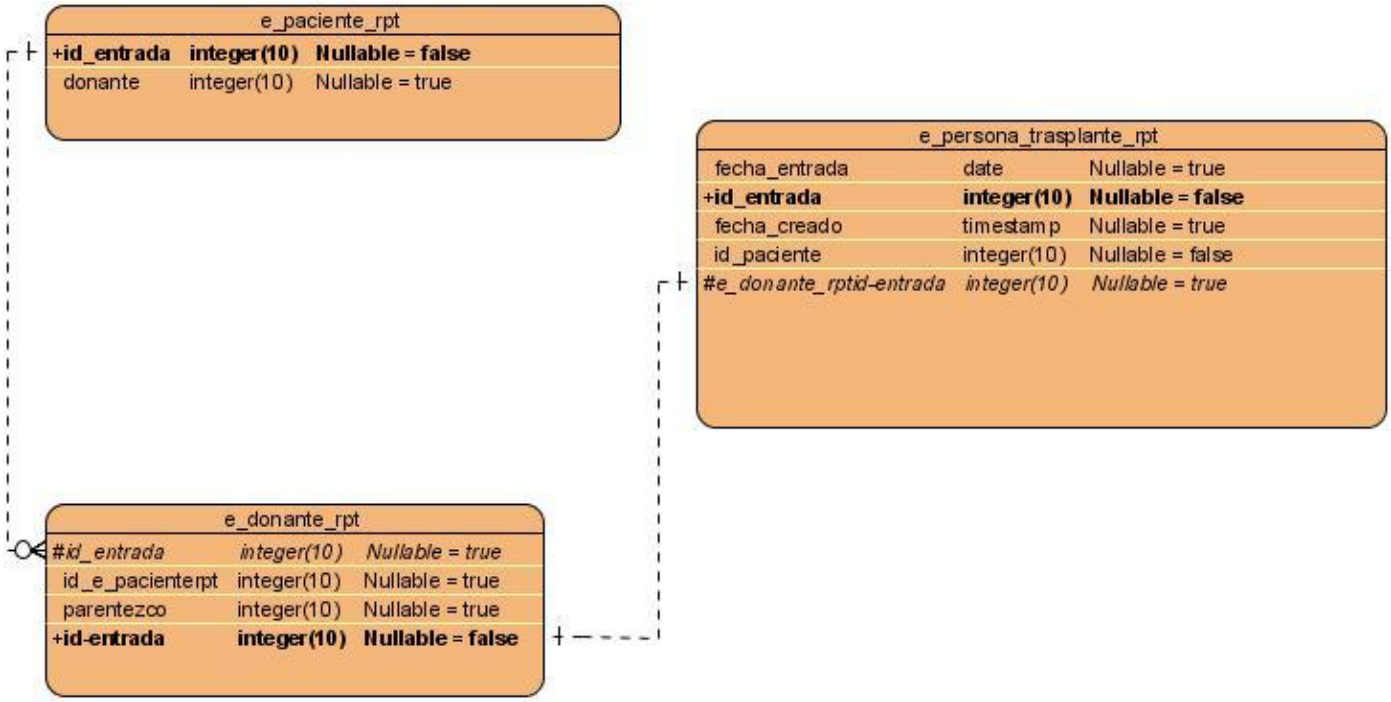



**Fig. 3.4. 13 Modelo de Datos: Caso de Uso Gestionar Donante de RPT**