# Universidad de las Ciencias Informáticas

## Facultad 7


# Título: Implementación del Módulo Centro Coordinador de Emergencia Médica Nacional

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

**Autores:** Yanosky Hernández López
Yuri Alberto Chacón Calvo

**Tutores**: Ing. Luis Mariano Reyna Soler
Ing. Maidelys Pulido Morera

**Asesora:** Roxana Cedeño Becerra.

**Ciudad de La Habana, Julio de 2008**
**"Año 50 de la Revolución"**

**Agradecimientos**

*"El agradecimiento es la parte principal de un hombre de bien"*

*Francisco de Quevedo y Villegas.*

*Nuestro más sincero agradecimiento a todas las personas que de una manera desinteresada han prestado su ayuda y apoyo en la elaboración de este trabajo; en especial a Daniel Miranda, Ricardo Collada, Dennys Lázaro, Leosdan Pozo y Alden Hernández sin cuyo aporte no hubiera sido posible la culminación satisfactoria obtenida.*

*A nuestros tutores Mariano y Maidelys por guiarnos en el trabajo y por ayudarnos durante este largo año.*

*A nuestro líder del Área Temática, José A. Segura por haber estado dispuesto a apoyarnos en los momentos difíciles.*

*A todos nuestros compañeros de estos cinco años de estudio.*

*En fin, a todos aquellos que de una forma u otra dieron su aporte para que alcanzáramos este resultado.*

*Muchas gracias.*

**Dedicatoria**

***De Yanosky:***

*A la memoria de mi amada abuela Guillermina que aunque ya no exista físicamente se que se siente orgullosa de los resultados de este nieto.*

*A la persona que catalogo en mi vida como mi más preciado tesoro: Mi Madre, por su apoyo, compresión y el amor que siempre me ha brindado.*

*A mi querido Padre por la educación que me supo inculcar y por ser mi guía todos estos años, pues hoy aunque la distancia nos separe, y a pesar de decir las cosas que dices, me siento muy orgulloso de poder haber cumplido con usted y conmigo mismo.*

*A mis hermanos Ariosky y Orlando por ser como son, y por no existir la palabra NO cuando se trate de algo referente al tema Yanosky.*

*A Leidy por todo su apoyo y comprensión durante todos estos años de relación.*

*A mis queridos abuelos Marta y Armando y a mis queridísimos tíos Lili y Guillermo por su gran preocupación y apoyo en el trayecto de estos 5 años.*

*A Juan Miguel Bruguera por ser como un Padre para mí, y a Cesar por toda su amistad.*

***De Yuri:***

*A mami y a Pozo por su guía y ejemplo y por esforzarse tanto para que yo estudiara y me graduara.*

*A mis Abuelos y mis tíos por su apoyo y cariño en todo momento.*

*A mi novia por apoyarme e impulsarme a ser mejor.*

*A toda mi familia y amigos por sus consejos y su apoyo en todos estos años de estudio.*

**Resumen**

En el Centro Nacional de Urgencias Médicas (SIUM) se encuentra el Centro Coordinador de Emergencia Médica Nacional (CCEMN), como núcleo de coordinación de las emergencias médicas en Cuba. Este centro lleva a cabo el control de los traslados interprovinciales y las coordinaciones de demandas. El objetivo del presente trabajo es desarrollar una aplicación Web que facilite la gestión de la información de las emergencias médicas y solicitudes de traslados interprovinciales en el CCEMN.

Para desarrollar el sistema se utilizó PHP como lenguaje de programación del lado del servidor, y JavaScript del lado del cliente. Rational Unified Process como metodología de desarrollo, UML como lenguaje para modelar todos los artefactos generados y Visual Paradigm como herramienta de modelado, como IDE de desarrollo Zend Studio, Dreamweaver como editor HTML y MySQL como gestor de base de datos.

Como resultado de este trabajo se obtuvo una aplicación Web que mejora la gestión de la información del Centro Coordinador de Emergencia Médica Nacional, logrando una mayor confidencialidad, integridad y disponibilidad de la información que se maneja, así como considerables mejoras en los reportes estadísticos lo cual permitirá agilizar la toma de decisiones.

**Palabras claves**

Centro Coordinador de Emergencia Médica Nacional, aplicación Web, traslados interprovinciales.

**Índice**

## INTRODUCCIÓN

En Cuba desde el triunfo de la Revolución existe un solo sistema de salud libre de pago para todos los cubanos. Este llega a todas las personas y lugares a lo largo y ancho del país, brindándole atención independientemente del sexo, raza, ideología y religión, a cualquier persona, que puede recibir desde una visita médica hasta un trasplante de órganos, cirugía del corazón o cuidados críticos.

El 1 de agosto de 1961 se crea el Ministerio de Salud Pública (MINSAP) como organismo rector de todas las actividades de salud del país, incluyendo las de las unidades privadas y mutualistas, como primer gran paso de avance para la integración del Sistema Nacional de Salud (SNS). Poco a poco fueron pasando a manos del pueblo estas instituciones privadas hasta que en septiembre de 1970 se incorpora la última entidad. Se llega así a la consolidación del primer SNS, único e integral en la historia de Cuba y del continente americano, quedando preparado el camino para continuar su perfeccionamiento y desarrollo. (1)

Desde el momento en que pasan a ser del pueblo las unidades de salud y se logró unificar el sistema de salud cubano, se han obtenido, mediante su desarrollo, logros significativos con resultados para el bienestar de la población cubana y del mundo. Este desarrollo alcanzado por la salud cubana, especialmente en el área tecnológica, no hubiese sido posible de no haber estado acompañado del desarrollo de la informática, desde sus inicios aplicada a la salud pública cubana.

En los momentos actuales en que el mundo revoluciona las tecnologías a un ritmo acelerado, el uso y desarrollo de la informática no ha quedado atrás, su aplicación se hace necesaria en cada rama de la ciencia y la técnica. El empleo de la informática, actualmente en todas las esferas de la vida, es una realidad conocida y una necesidad para toda empresa que desee estar a la vanguardia en la obtención de sus resultados. En particular en la empresa moderna, sea ésta productiva, comercial o de servicios, la computación es fundamental para asegurar sus crecientes y complejas funciones[1].

La informatización de la salud pública es una de las transformaciones más notables en el desarrollo tecnológico de la salud cubana, impulsada por la máxima dirección del país y el MINSAP con el objetivo de incorporar la informática y los avances en las Tecnologías de la Información y las Comunicaciones (TIC) a los procesos vinculados a la salud, permitiendo el intercambio de experiencias

[1] Conjunto de instrucciones que permiten procesar las variables para obtener un resultado.

con los especialistas más destacados del país e incluso con científicos foráneos a través de las redes digitales.

Actualmente, se trabaja integradamente en el desarrollo de un grupo de aplicaciones básicas para la informatización del sector de la salud. En su desarrollo e implementación participan diferentes empresas del Ministerio de la Informática y Comunicaciones (MIC), la Universidad de Ciencias Informáticas (UCI), INFOMED[2], el Centro de Desarrollo Informático para la Salud Pública (CEDISAP) y las Direcciones Nacionales del MINSAP implicadas directamente en los primeros productos. (2). De esta forma el MINSAP se acoge al proceso de informatización junto al de la sociedad cubana para expresar con eficiencia y calidad la atención médica al pueblo.

El Centro Nacional de Urgencias Médicas (SIUM) está insertado dentro del SNS, su misión es organizar la atención de urgencias y emergencias médicas desde la comunidad, consultorios, policlínicos, Coordinaciones de Ambulancias de Urgencias y Emergencia hasta el Sistema de Emergencia y Terapia Hospitalaria, que se establece mediante un proceso de evaluación y decisión médica, a través de los diferentes eslabones del SNS. (3)

Dentro del SIUM se encuentra el Centro Coordinador de Emergencia Médica Nacional (CCEMN) como núcleo de coordinación de las emergencias a nivel nacional. En este centro coordinador se lleva a cabo el control de los traslados interprovinciales y de las coordinaciones de demandas tales como: emergencias dentro de Ciudad de la Habana, traslados entre institutos y hospitales, altas hospitalarias, el movimiento del equipo médico encargado de realizar trasplantes de órganos; así como el control de las ambulancias existentes en la base, debido a que este centro cuenta con sus propios móviles para darle cumplimiento a las demandas realizadas al mismo.

Una demanda se inicia cuando se realiza una llamada telefónica solicitando un servicio, seguidamente se recogen los datos del paciente, así como otros datos de interés, se define el móvil[3] que atenderá dicha demanda y se le da seguimiento al vehículo después que se pone en marcha. En el caso de los traslados interprovinciales el Centro Coordinador Provincial de Emergencia Médica (CCPEM) llama al CCEMN para informar el traslado de un paciente y este debe autorizar la entrada del vehículo en

[2] 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.
[3] Ambulancia o vehículo equipado para atender Urgencias Medicas.

Ciudad de la Habana, de igual forma se recogen una serie de datos de interés que influyen en la toma de decisiones y el envío de pacientes, que son altas hospitalarias, para sus lugares de residencia.

Actualmente, todo este proceso de recogida de datos de las demandas efectuadas al CCEMN se realiza de forma manual y no se almacena la información de los traslados interprovinciales que se realizan entre las provincias, a pesar de que se informan, exceptuando solamente el caso de cuando el traslado es con destino a Ciudad de la Habana. Con la información recogida no se puede llevar a cabo un control estadístico de disímiles aspectos importantes para una futura investigación que se desee realizar por parte de los especialistas e investigadores de la salud. Esta obsoleta forma de trabajo vigente en este organismo trae como consecuencias: pérdida de demandas, demoras en la ejecución de demandas de emergencias, quejas de la población por incumplimiento de recogidas de pacientes y falta de control de los móviles que viajan sin la debida autorización del CCEMN.

Debido a lo planteado anteriormente surge el siguiente **Problema Científico:** ¿Cómo facilitar la gestión de la información de las emergencias médicas en el Centro Coordinador de Emergencia Médica Nacional?

Teniendo como **Objeto de Estudio** el proceso de gestión de la información en el SIUM y como **Campo de Acción** derivado del mismo, el proceso de gestión de la información del Centro Coordinador de Emergencia Médica Nacional.

Para dar solución a la problemática planteada se ha planteado como **Objetivo General**: implementar una aplicación Web que facilite la gestión de la información de las emergencias médicas en el CCEMN.

Para poder cumplir este objetivo y lograr una solución adecuada a la situación problemática especificada se plantean las siguientes **Tareas de la Investigación:**

1. Valorar las nuevas tendencias de las tecnologías de la información relacionadas con las aplicaciones Web.

2. Realizar un análisis crítico de la selección de la tecnología, metodología de desarrollo de software[4], el lenguaje de programación, el gestor de base de datos y las herramientas a utilizar.
3. Analizar las necesidades de funcionamiento del sistema informático.
4. Valorar la integración con otros componentes ya existentes en el SISalud[5], así como la integración con otras partes del Sistema Informatizado para el Centro Nacional de Urgencias Médicas (SIUM).
5. Obtener el Modelo de Implementación y los artefactos necesarios que describan la base de datos.
6. Realizar la implementación del Módulo CCEMN, basándose en la propuesta de arquitectura definida por el MINSAP (Orientada a Servicios y Basada en Componentes).
7. Realizar la implementación del Módulo CCEMN utilizando los patrones de diseño establecidos en el Análisis y Diseño, así como la implementación de la capa de acceso a datos y procedimientos almacenados.

Con el presente trabajo de investigación se espera obtener como resultado el Diseño de la base de datos e Implementación del módulo CCEMN del Sistema Informatizado para el Centro Nacional de Urgencias Médicas (SIUM).

Este documento se encuentra estructurado en tres capítulos que contienen toda la información referente a la investigación realizada:

En el **Capítulo 1**, denominado "Fundamentación Teórica", se realiza una fundamentación de las tecnologías, metodologías de desarrollo y herramientas a utilizar en el desarrollo de la aplicación.

En el **Capítulo 2**, denominado "Elementos de Arquitectura y Seguridad", se realiza un análisis de los patrones arquitectónicos que se han tenido en cuenta para el desarrollo del sistema y se describe la estrategia de seguridad trazada.

En el **Capítulo 3**, denominado "Descripción y Análisis de la Solución Propuesta", se realiza un análisis de los componentes con los que se integra el sistema y las relaciones específicas que tiene con los

[4] Programas de sistema, utilerías o aplicaciones expresados en un lenguaje de máquina.
[5] Sistema de Información para la Salud: Es la integración de un conjunto de aplicaciones cuyo propósito fundamental es la informatización del Sistema Nacional de Salud.

mismos. Además, se realiza una descripción de las clases[6], los métodos que las mismas poseen y las tablas que componen la base de datos del sistema.

[6] Abstracciones que representan a un conjunto de objetos con un comportamiento e interfaz común.

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

## Introducción.

En el presente capítulo se realiza un análisis de las tecnologías y la metodología que se hace necesario utilizar en el desarrollo de una posible solución a las necesidades planteadas, teniendo en cuenta las generalidades del entorno de aplicación de la solución propuesta y su fundamentación. Se elabora una descripción detallada de las herramientas definidas para ser utilizadas y se describen los lenguajes de programación de la aplicación Web a desarrollar, el gestor de bases de datos y el servidor Web a utilizar, teniendo en cuenta los requerimientos e interés del usuario final.

## 1.1 Fundamentación de las Tecnologías y Herramientas a utilizar.

INFOMED es la red electrónica cubana de información para del sistema de salud cubano, surgió como parte de un proyecto del Centro Nacional de Información de Ciencias Médicas de Cuba. Esta red permite la accesibilidad a toda la información relacionada con las ciencias de la salud y especialmente a la información producida en Cuba en este campo. Para la incorporación de un sistema informático a INFOMED debe ser desarrollado sobre la arquitectura de software establecida por el MINSAP.

La Facultad 7 perteneciente a la Universidad de las Ciencias Informáticas con el fin de garantizar la integración con otros componentes desarrollados en el SISalud definió un Documento de Arquitectura basado en la Arquitectura establecida por el MINSAP. El Módulo CCEMN del Sistema Informatizado para el Centro Nacional Urgencias Médicas, que se encuentra dentro del Área Temática Sistemas Especializados y que forma parte de la ya mencionada facultad, adoptó dicho documento para normar los procesos de desarrollo.

En epígrafes posteriores se realizará una descripción de los elementos que se definen dentro del Documento de Arquitectura de la Facultad 7 para aplicaciones Web y que se hacen necesarios en el desarrollo del presente trabajo.

## 1.2 Tendencias y tecnologías actuales a considerar.

### 1.2.1 Internet.

Posee definiciones como "La Red de Redes" o "La Autopista de la Información", es decir, una red que no sólo interconecta computadoras, sino que interconecta redes de computadoras entre sí. Una red de computadoras es un conjunto de máquinas que se comunican a través de algún medio (cable coaxial, fibra óptica, radiofrecuencia, líneas telefónicas, etc.) con el objeto de compartir recursos. De esta manera, Internet sirve de enlace entre redes más pequeñas y permite ampliar su cobertura al hacerlas parte de una "red global". Esta red global tiene la característica de que utiliza un lenguaje común que garantiza la intercomunicación de los diferentes participantes; este lenguaje común o protocolo se conoce como TCP/IP[7].

Internet funciona con la estrategia Cliente/Servidor[8], un paradigma de división del trabajo informático en el que las tareas se reparten entre un número de clientes que efectúan peticiones de servicios de acuerdo con un protocolo, y un número de servidores que las atienden.
Resumiendo, Internet es la "red de redes" que utiliza TCP/IP como su protocolo de comunicación. (4)

### 1.2.2 Aplicaciones Web.

Una aplicación Web es aquella que los usuarios usan accediendo a un servidor Web[9] a través de Internet o de una intranet. Las aplicaciones Web son aplicaciones basadas en el muy extendido paradigma "cliente/servidor". Este paradigma consiste en un servidor que sabe cómo proporcionar un servicio y un cliente que desea acceder al servicio. Por ejemplo al "leer el periódico" en forma digital existe un servidor que contiene las noticias del día y es necesaria una aplicación que acceda a este servidor de alguna forma para obtener la información. Esta aplicación cliente, debe ser capaz de mostrar la información al usuario final que desea leer las noticias.

En términos más simples, una aplicación Web es un sistema Web que permite a los usuarios ejecutar lógica de negocio a través de un navegador o lo que es lo mismo, modificar el estado del negocio. Las Aplicaciones Web hacen uso de las tecnologías que existen para permitir a los usuarios del sistema

[7] Transference Control Protocol / Internet Protocol: Protocolo de Control de Transferencia / Protocolo de Internet.
[8] Modelo para el desarrollo de sistemas de información, en el que las transacciones se dividen en elementos independientes que cooperan entre sí para intercambiar información, servicios o recursos.
[9] Computadora de gran capacidad y prestaciones que almacena, controla y comparte información con estaciones de trabajo clientes.

modificar la lógica del negocio en el servidor y para generar contenidos dinámicos, de no existir lógica de negocio en el servidor el sistema es considerado como un sitio, no como una aplicación Web.

Las Aplicaciones Web generalmente presentan una arquitectura [10] simple, como componentes principales usan el servidor Web, la red y el navegador. El servidor es el encargado de distribuir las páginas por cada petición de los clientes. Las peticiones se hacen a través de las conexiones de redes y para hacerlo utilizan el protocolo de comunicación HTTP. Para mostrar la información a los usuarios y hacer las validaciones en la entrada de los datos se usa siempre un browser o navegador.

Entre las principales ventajas de las Aplicaciones Web se encuentran:

- ✓ Se puede migrar de Sistema Operativo[11] o cambiar el Hardware[12] libremente sin afectar el funcionamiento de las aplicaciones de servidor, o sea, son multiplataforma.
- ✓ No se requieren complicadas combinaciones de Hardware/Software para utilizar estas aplicaciones. Solo un computador con un buen navegador Web.
- ✓ Se facilita el trabajo a distancia. Se puede trabajar desde cualquier PC[13] o computador portátil con conexión a Internet.
- ✓ Al funcionar en un navegador, se requiere un conocimiento básico de informática para utilizar una aplicación Web.
- ✓ Las actualizaciones de software son distribuidas automáticamente y de forma transparente al usuario.

No obstante, a la serie de ventajas que presenta se le suman además algunas desventajas, tales como:

- ✓ La seguridad es mucho más difícil de implementar.

[10] Estructura organizativa de un sistema que incluye su descomposición en partes, su conectividad, mecanismos de interacción y principios de guía que proporciona información sobre el diseño del mismo.
[11] Conjunto de programas que se integran con el hardware para facilitar al usuario, el aprovechamiento de los recursos disponibles.
[12] Componentes electrónicos, tarjetas, periféricos y equipo que conforman un sistema de computación; se distinguen por ser elementos tangibles.
[13] Personal Computer: Computadora Personal.

- ✓ Los elementos estándar HTML tienen algunas limitaciones. Dependiendo del navegador que se use, pueden aparecer diferentes para varios usuarios. Esto puede causar alguna confusión si los usuarios se mueven de una estación de trabajo a otra diferente.
- ✓ El desempeño puede ser bajo en un servidor poco potente debido a que se están enviando los datos y el diseño de la pantalla cada vez que se pide un documento HTML.
- ✓ Se necesita de una conexión de red permanente y rápida.
- ✓ La interactividad no se produce en tiempo real, en las Aplicaciones Web cada acción del usuario conlleva un tiempo de espera algunas veces excesivo hasta que obtiene la reacción del sistema. (5) (6)

### 1.2.3 Servidor Web.

Es un programa que implementa el protocolo HTTP (HyperText Transfer Protocol)[14]. Este protocolo está diseñado para transferir los hipertextos, páginas Web o páginas HTML (Hypertext Markup Language)[15]. Un servidor Web se encarga de mantenerse a la espera de peticiones HTTP llevada a cabo por un cliente que se conoce como navegador. El navegador realiza una petición al servidor y éste construye una página Web que ejecuta un programa en el servidor, siempre elabora una respuesta en formato HTML para mostrar al cliente o al navegador que hace la petición. (7)

### Servidor Web Apache.

Diseñado para ser un servidor Web potente y flexible que pueda funcionar en la más amplia variedad de plataformas y entornos, lo que hace que a menudo sean necesarias diferentes características o funcionalidades. Apache se ha adaptado siempre a una gran variedad de entornos a través de su diseño modular. Este diseño permite a los administradores de sitios Web elegir que características van a ser incluidas en el servidor seleccionando, que módulos se van a cargar, ya sea al compilar o al ejecutar el servidor.

Apache se suele utilizar en combinación con Linux, MySQL para las bases de datos y PHP para la programación Web. Tanto es así, que se ha llegado a acuñar el término LAMP (Linux, Apache, MySQL, PHP). La última letra también puede hacer referencia a otros lenguajes como Perl o Python[16].

[14] Protocolo de Transferencia de Hipertextos. Modo de comunicación para solicitar páginas Web.
[15] Lenguaje de marcas usado para escribir documentos para servidores World Wide Web y que se comparten mediante HTTP.
[16] Lenguaje de programación de Código Abierto administrado por la Python Software Foundation.

Actualmente el servidor apache es el servidor Web más utilizado en el mundo con un 71% de cobertura en el mercado, es la elección de las grandes compañías a nivel mundial. Incluso en servidores con Microsoft Windows [17]es utilizado Apache como servidor Web.

Se encuentran dentro de la gran multitud de plataformas para las cuales está disponible el servidor Web Apache:

- ✓ FreeBSD, NetBSD, OpenBSD.
- ✓ GNU/Linux[18].
- ✓ Mac OS y Mac OS X Server.
- ✓ Netware.
- ✓ OpenStep/Match.
- ✓ UNIX comerciales como AIX (R), Digital UNIX (R), HP-UX (R),IRIX (R), SCO (R), Solaris (R), SunOS(R), UnixWare (R).
- ✓ Windows (R).

El servidor Web tiene como elementos positivos en su aplicación que:

- ✓ Permite la autenticación de usuarios en varias formas.
- ✓ Es un servidor seguro y eficiente Open Source ideal para Sistemas Operativos modernos de Servidores.
- ✓ Permite el uso de bases de datos DBM para la autenticación de usuarios.
- ✓ Permite la creación de ficheros de log a medida del administrador.
- ✓ Permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor.
- ✓ Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.

[17] Sistema Operativo más usado en la actualidad desarrollado por Microsoft Corporation.
[18] Es el nombre de un núcleo, pero se suele denominar con este nombre a un sistema operativo de libre distribución.

- ✓ Funciona en casi todas las plataformas actuales.
- ✓ Permite la creación de sitios Web dinámicos:
  - El uso de CGI's[19].
  - El uso de Server Side Includes (SSI).
  - El uso de lenguajes de Scripting como PHP, Javascript, Python.
  - El uso de Java y páginas JSP. (8) (9)

### 1.2.4 HyperText Transfer Protocol HTTP.

HTTP es un protocolo del nivel de aplicación para sistemas de información multimedia distribuidos. Es un protocolo no orientado a estado que puede ser utilizado para más propósitos que para manejar ficheros HTML.

Entre sus propiedades se pueden destacar las siguientes:

- ✓ Es un esquema de direccionamiento comprensible.
- ✓ Utiliza el Universal Resource Identifier (URI) para localizar sitios (URL) o nombres (URN) sobre los que hay que aplicar un método. La forma general de un URL es servicio://host/fichero.ext.
- ✓ Está basado en la arquitectura Cliente-Servidor.
- ✓ Se asienta en el paradigma solicitud/respuesta. La comunicación se asienta sobre TCP/IP. El puerto por defecto es el 80, pero se pueden utilizar otros.
- ✓ Es un protocolo sin conexión y sin estado.
- ✓ Después de que el servidor ha respondido la petición del cliente, se rompe la conexión entre ambos. Además no se guarda memoria del contexto de la conexión para siguientes conexiones.
- ✓ Está abierto a nuevos tipos de datos.

[19] Common Gateway Interface: mecanismo de comunicación entre el servidor web y una aplicación externa

- ✓ Utiliza tipos MIME[20] (Multipart Internet Mail Extension) para la determinación del tipo de los datos que transporta. Cuando un servidor HTTP transmite información de vuelta a un cliente, incluye una cabecera que le indica al cliente sobre los tipos de datos que componen el documento. De la gestión de esos datos se encargan las utilidades que tenga el cliente (visor de imágenes, de vídeo, etc.)

Una transacción HTTP está compuesta por una cabecera, y opcionalmente, por una línea en blanco seguida de los datos. En la cabecera se especifica tanto la acción solicitada en el servidor, como los tipos de datos devueltos o un código de estado. (10)

### 1.2.5 Navegador Web.

Puede considerarse como una interfaz de usuario universal. Dentro de sus funciones están la petición de las páginas Web, la representación adecuada de sus contenidos y la gestión de los posibles errores que se puedan producir.

Para todo esto, los fabricantes de navegadores les han dotado de posibilidades de ejecución de programas de tipo script, con modelos de objetos que permiten manipular los contenidos de los documentos. Estos lenguajes de programación son VBScript, JScript (ambas de Microsoft) y Java Script (de Netscape) que proporcionan las soluciones llamadas del lado del cliente y permiten realizar validaciones de datos recogidos en las páginas antes de enviarlos al servidor con lo que proporcionan un alto grado de interacción con el usuario dentro del documento.

Otras de las posibilidades de los navegadores es la gestión del llamado HTML dinámico (Dinamic HTML ó DHTML). Éste está compuesto de HTML, hojas de estilo en cascada, (Cascade Style Sheets, CSS), modelo de objetos y scripts de programación que permiten formatear y posicionar correctamente los distintos elementos HTML de las páginas Web, permitiendo un mayor control sobre la visualización de las páginas.

En esta línea, los navegadores han ido un poco más allá y permiten las visualizaciones de documentos XML (eXtensible Markup Language) después de haber sido transformado adecuadamente a HTML por las hojas de estilo extensibles (eXtensible Style Sheets, XSL). De esta manera se puede elegir visualizar ciertos elementos y otros no, dependiendo de las circunstancias. (11)

[20] Especificaciones dirigidas a que se puedan intercambiar a través de Internet todo tipo de archivos (texto, audio, vídeo, etc.) de forma transparente para el usuario

### 1.2.6 Servicios Web.

Existen múltiples definiciones sobre lo que son los servicios Web, lo que muestra su complejidad a la hora de dar una adecuada definición que englobe todo lo que son e implican. Una posible sería hablar de ellos como un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web.

Estos servicios proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas, es necesaria una arquitectura de referencia estándar.

El siguiente gráfico muestra cómo interactúa un conjunto de Servicios Web:

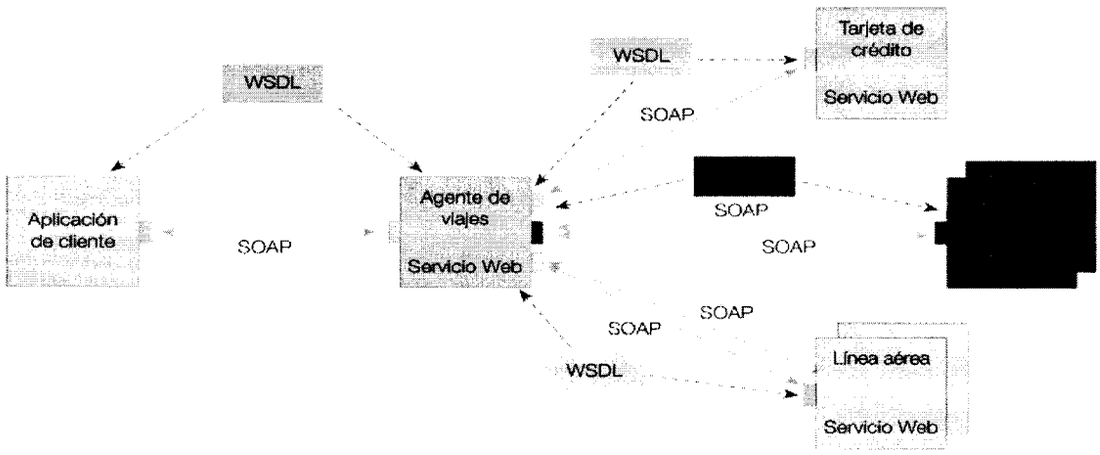


**Figura 1 Interacción de un servicio Web.**

Estándares empleados:

- ✓ XML (Extensible Markup Language): Es el formato estándar para los datos que se vayan a intercambiar.
- ✓ SOAP (Simple Object Access Protocol) o XML-RPC (XML Remote Producer Call): Protocolos sobre los que se establece el intercambio.

- ✓ 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).
- ✓ 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.
- ✓ WS-Security (Web Service 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.

Los servicios Web traen beneficios como:

- ✓ Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen.
- ✓ Fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento.
- ✓ Al apoyarse en HTTP, pueden aprovecharse de los sistemas de seguridad firewall sin necesidad de cambiar las reglas de filtrado.
- ✓ Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.
- ✓ Permiten la interoperabilidad entre plataformas de distintos fabricantes por medio de protocolos estándar. (12) (13)

### 1.2.7 XML

Extensible Markup Language es un metalenguaje extensible de etiquetas y permite definir la gramática de lenguajes específicos por lo que no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. XML es una tecnología sencilla que tiene a su alrededor

otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

La tecnología XML busca dar solución al problema de expresar información estructurada de la manera más abstracta y reutilizable posible. Que la información sea estructurada quiere decir que se compone de partes bien definidas, y que esas partes se componen a su vez de otras partes. Estas partes se llaman elementos, y se las señala mediante etiquetas.

### 1.2.8 AJAX.

AJAX es una técnica de desarrollo Web que combina tecnologías existentes para crear aplicaciones interactivas o RIA (Rich Internet Applications). Estas se ejecutan en el navegador de los usuarios y mantienen una comunicación de forma asíncrona con el servidor en segundo plano lo que permite realizar cambios sobre una página Web sin necesidad de recargarla. Esto proporciona un aumento de la interactividad, velocidad y usabilidad en la misma.

AJAX no constituye una tecnología en sí, pero es un término que engloba a un grupo de éstas que trabajan conjuntamente. Incorpora dentro de sí:

- ✓ Presentación basada en estándares usando XHTML y CSS;
- ✓ Exhibición e interacción dinámicas usando el Document Object Model (DOM)
- ✓ Intercambio y manipulación de datos usando XML y XSLT
- ✓ Recuperación de datos asincrónica usando XMLHttpRequest.

## 1.3 Metodología de Desarrollo.

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas, herramientas y ayudas a la documentación que deben tenerse presente para el desarrollo organizado de productos de software.

### 1.3.1 Rational Unified Process (RUP).

La metodología RUP, llamada así por sus siglas en inglés Rational Unified Process, es un producto de Rational Software Corporation. Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso. Se plantea que el RUP es un proceso de desarrollo de

software que describe un conjunto de actividades para transformar los requerimientos del cliente en un Sistema de Software. Tiene como objetivo asignar tareas y responsabilidades para producir software de alta calidad, buscando satisfacer las necesidades de los clientes, ajustándose al presupuesto y a los tiempos estimados. El RUP ofrece diferentes subprocesos, que brindan un marco de trabajo adaptable y extensible a las necesidades de cada organización. RUP divide el proceso de desarrollo en ciclos, teniendo un producto al final de cada uno, y estos se dividen en fases que finalizan con un hito donde se debe tomar una decisión importante:

- ✓ Inicio: Determinar la visión del proyecto.
- ✓ Elaboración: Determinar la arquitectura óptima.
- ✓ Construcción: Obtener la capacidad operacional inicial.
- ✓ Transmisión: Obtener la liberación del proyecto.

Cada una de estas etapas es desarrollada mediante un ciclo de iteraciones, consistente en reproducir el ciclo de vida del proceso en cascada y a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes. Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software. El ciclo de vida que se desarrolla por cada iteración, está basado en Flujos de Trabajo (figura 2).

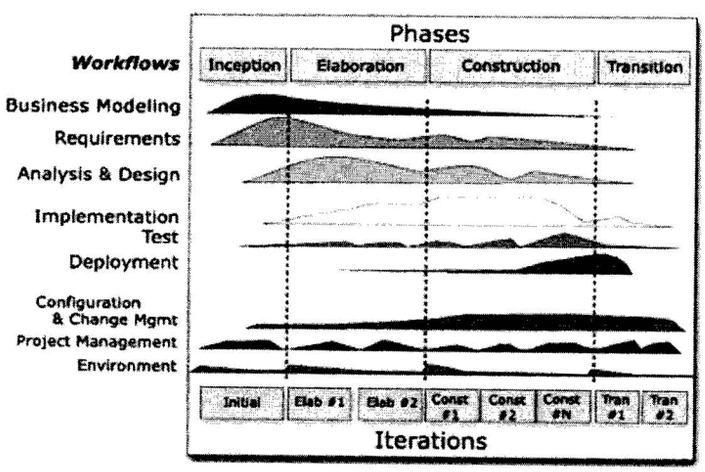


**Figura 2: Fases e Iteraciones de la Metodología RUP.**

Cada una de estas iteraciones debe ser clasificada y ordenada según su prioridad, y cada una se convierte en un grupo de entregables al cliente. Esto trae como beneficio la retroalimentación en cada iteración. Según los criterios analizados, RUP no debe catalogarse como una metodología tradicional o ágil, sino que es una metodología a la medida entre las dos clasificaciones, debido a que tiene bien definido sus procesos orientados a objetos, puede ser menos pesado si se es capaz de aceptar y adaptar a las condiciones esperadas de cada proyecto. Además de traer consigo un grupo de beneficios como la estandarización, facilita el entendimiento por parte de los usuarios del software, que no necesariamente deben tener conocimientos previos sobre el tema y facilita además, el desarrollo del producto a distancia de los clientes. (14)

### 1.3.2 UML.

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, hojear, configurar, mantener, y controlar la información sobre tales sistemas. Está pensado para usarse con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios.

Pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. Incluye conceptos semánticos, notación, y principios generales. Tiene partes estáticas, dinámicas, de entorno y organizativas. Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. La especificación de UML no define un proceso estándar pero está pensado para ser útil en un proceso de desarrollo iterativo.

Capta la información sobre la estructura estática y el comportamiento dinámico de un sistema. También contiene construcciones organizativas para agrupar los modelos en paquetes, lo que permite a los equipos de software dividir grandes sistemas en piezas de trabajo, para entender y controlar las dependencias entre paquetes, y para gestionar las versiones de las unidades del modelo, en un entorno de desarrollo complejo. Contiene construcciones para representar decisiones de implementación y para elementos de tiempo de ejecución en componentes.

UML no es un lenguaje de programación. Las herramientas pueden ofrecer generadores de código de UML para una gran variedad de lenguajes de programación, así como construir modelos por ingeniería

inversa a partir de programas existentes. No es un lenguaje altamente formal pensado para probar teoremas. Es un lenguaje de modelado de propósito discreto y general. (15)

## 1.4 Lenguajes de Programación Web Definidos.

Un lenguaje de programación es la forma mediante la cual una persona puede crear programas y software que pueden ser utilizados para controlar el comportamiento de una computadora o llevar a la misma a que provea al usuario de una respuesta o acción solicitada. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos.

### 1.4.1 Lenguajes de Programación del lado del Servidor.

Los lenguajes de programación del lado del servidor son aquellos lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él.

#### PHP

Es un lenguaje interpretado de alto nivel incrustado en páginas HTML y ejecutado en el servidor, presentando el estilo clásico de cualquier otro lenguaje de programación, con variables, sentencias condicionales, bucles y funciones. No es un lenguaje de marcas como podría ser HTML, XML o WML. El resultado es normalmente una página HTML.

Fue originalmente diseñado en Perl, seguido por la escritura de un grupo de CGI binarios escritos en el lenguaje C por el programador danés-canadiense Rasmus Lerdorf en el año 1994 para mostrar su currículum vitae y guardar ciertos datos, como la cantidad de tráfico que su página Web recibía. El 8 de junio de 1995 fue publicado "Personal Home Page Tools" después de que Lerdorf lo combinara con su propio Form Interpreter para crear PHP/FI. Esta forma de programar llego a muchos usuarios, pero el lenguaje no tomo el peso actual hasta que Zeev Surasky y Andi Gutmans le incluyeron nuevas características en 1997, que dio por resultado el PHP 3.0.

El funcionamiento de este lenguaje se puede describir a través de los pasos siguientes:

1. Escribir en las páginas HTML pero con el código PHP dentro.
2. Guardar la página en el servidor Web.
3. Un navegador solicita una página al servidor.

4. El servidor interpreta el código PHP.
5. El servidor envía el resultado del conjunto de código HTML y el resultado del código PHP que también es HTML.

En ningún caso se envía código al navegador, por lo que todas las operaciones realizadas son transparentes al usuario, el código PHP es ejecutado en el servidor y el resultado es enviado al navegador. El resultado es normalmente una página HTML. Por lo que al usuario le parecerá que está visitando una página HTML que cualquier navegador puede interpretar.

Al ser PHP un lenguaje que se ejecuta en el servidor no es necesario que el navegador lo soporte, es independiente del navegador, pero sin embargo para que sus páginas PHP funcionen, el servidor donde están alojadas debe soportar PHP. Además se encuentra libre en el mercado y se puede acceder a él por medio de Internet, tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos tales como UNIX (y de ese tipo, como Linux o Mac OS X) y Windows, y puede interactuar con los servidores Web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI.

PHP ofrece una solución simple y universal para las paginaciones dinámicas del Web de fácil programación. Su diseño elegante lo hace perceptiblemente, más fácil de mantener y ponerse al día que el código comparables en otros lenguajes. Debido a su amplia distribución PHP está perfectamente soportado por una gran comunidad de desarrolladores. Como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y se reparen rápidamente.

El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP. Últimamente, también puede ser utilizado para la creación de otro tipo de programas incluyendo aplicaciones con interfaz gráfica usando las librerías[21] Qt o GTK+, además de otras varias bibliotecas externas que permiten que el desarrollador haga casi cualquier cosa desde generar documentos en PDF (Portable Document Format) hasta analizar código XML (eXtensible Markup Language).

De modo que, con PHP se tiene la libertad de elegir el sistema operativo que más guste, también se tiene la posibilidad de usar programación procedimental ó programación orientada a objetos. Aunque

[21] Conjunto de procedimientos y funciones (subprogramas) agrupadas en un archivo con el fin de ser aprovechadas por otros programas.

no todas las características estándares de la programación orientada a objetos están implementadas en las versiones de PHP, muchas librerías y aplicaciones grandes están escritas íntegramente usando.

Las aplicaciones Web creadas con PHP son muy robustas ya que Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite.

En el 2004 fue lanzado el PHP 5 utilizando el motor Zend Engine II (o Zend Engine 2), dentro de las nuevas versiones de PHP 5 se encuentra la 5.2.5 que incluye todas las ventajas que provee el nuevo Zend Engine 2 como:

- ✓ Soporte sólido y REAL para Programación Orientada a Objetos (OOP) con PHP Data Objects.
- ✓ Mejoras de rendimiento.
- ✓ Mejor soporte para MySQL con extensión completamente reescrita.
- ✓ Mejor soporte a XML (XPath, DOM...).
- ✓ Soporte nativo para SQLite.
- ✓ Soporte integrado para SOAP.
- ✓ Iteradores de datos.
- ✓ Excepciones de errores.

### 1.4.2 Lenguajes de Programación del lado del Cliente.

Los lenguajes de programación del lado del cliente son aquellos lenguajes que no requieren de compilación ya que los navegadores son los encargados de interpretar estos códigos y generar una respuesta.

#### JavaScript.

JavaScript es un lenguaje de programación interpretado (que no requiere compilación) del lado del cliente ya que el navegador es el que soporta la carga de procesamiento. Gracias a que es un lenguaje de programación muy versátil y potente, tanto para la realización de pequeñas tareas como para la

gestión de complejas aplicaciones y a su compatibilidad con la mayoría de los navegadores modernos, en la actualidad es el lenguaje de programación del lado del cliente más utilizado.

Está diseñado para controlar la apariencia y manipular los eventos dentro de la ventana del navegador Web. Este se integra directamente en páginas HTML y la ventaja que presenta sobre el HTML es que permite crear páginas Web más dinámicas, lo que las hace más atractivas para el usuario. (16)

## 1.5 Framework

El concepto framework se emplea en muchos ámbitos del desarrollo de sistemas software, no sólo en el ámbito de aplicaciones Web, es posible encontrar frameworks para el desarrollo de aplicaciones médicas, de visión por computador, para el desarrollo de juegos, y para otros innumerables ámbitos existentes en el mundo de la ciencia y la técnica.

En general, con el término framework, se hace referencia a una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que pueden serle añadidas las últimas piezas para construir una aplicación concreta. Los objetivos principales que persigue un framework son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones.

Un framework Web, por tanto, se puede definir como un conjunto de componentes que componen un diseño reutilizable que facilita y agiliza el desarrollo de sistemas Web. (17)

### CodeIgniter

CodeIgniter es un framework de código abierto (Open Source), que proporciona un marco de trabajo para el desarrollo de aplicaciones Web utilizando como lenguaje de programación PHP. Facilita el trabajo a los desarrolladores ya que evita la complejidad promoviendo soluciones simples. Cuenta también con una serie de clases, librerías y plugins desarrollados por la comunidad de usuarios con el objetivo de extender sus funcionalidades. Soporta MVC, AJAX, manejo de sesiones de usuarios, múltiples SGBD (creadas en MySQL, MySQLi, MS SQL, PostgreSQL, Oracle, SQLite, ODBC) y conexión simultanea a más de una base de datos.

**Active Records.**

El Patrón Active Records es un patrón de software que se encuentra frecuentemente en softwares que almacenan su información en Bases de Datos Relacionales. Este es un enfoque para acceder a los datos en una base de datos.

Una tabla o vista de la base de datos es incluida dentro de una clase contenedora, por lo que la instancia de un objeto está vinculada a una fila de la tabla. Después de la creación del objeto, una nueva fila es adicionada a la tabla a salvar. Cualquier objeto cargado obtiene su información de la base de datos, cuando un objeto es actualizado, la fila correspondiente en la tabla es actualizada también. La clase contenedora implementa los métodos de accesos o propiedades para cada columna en la tabla o vista.

Este patrón es utilizado comúnmente por las herramientas de objetos persistentes y en el mapeado objeto-relacional. Típicamente las relaciones de las llaves foráneas serán mostradas como instancias de objetos del tipo adecuado a través de una propiedad.
La implementación del Patrón Active Records puede ser encontrada en diversos Frameworks para múltiples entornos de programación.

## 1.6 Sistemas Gestores de Bases de Datos.

Los Sistemas de Gestión de Bases de Datos (SGBD) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta.

Los SGBD proporcionan a los usuarios la capacidad de almacenar datos en la base de datos, acceder a ellos y actualizarlos.

En un ambiente multiusuario el SGBD ofrece un control centralizado de la información. Los objetivos que plantean estos sistemas están relacionados con la intención de evitar los problemas que existían en los sistemas de información orientados a los procesos. Los principales objetivos son:

- ✓ Evitar la redundancia de los datos, eliminando así la inconsistencia de los mismos.

- ✓ Mejorar los mecanismos de seguridad de los datos y la privacidad.
- ✓ Asegurar la independencia de los programas y los datos, es decir, la posibilidad de modificar la estructura de la base de datos sin necesidad de modificar los programas de las aplicaciones que manejan esos datos.
- ✓ Mantener la integridad de los datos realizando las validaciones necesarias cuando se realicen modificaciones en la base de datos.
- ✓ Mejorar la eficacia de acceso a los datos, en especial en el caso de consultas imprevistas.

**MySQL.**

MySQL es un sistema de administración de bases de datos relacional (RDBMS). Se trata de un programa capaz de almacenar una enorme cantidad de datos de gran variedad y de distribuirlos para cubrir las necesidades de cualquier tipo de organización, desde pequeños establecimientos comerciales a grandes empresas y organismos administrativos. MySQL compite con sistemas RDBMS propietarios como Oracle, SQL Server y Db2.

MySQL incluye todos los elementos necesarios para instalar el programa, preparar diferentes niveles de acceso de usuario, administrar el sistema y proteger los datos. Puede desarrollar sus propias aplicaciones de bases de datos en la mayor parte de lenguajes de programación utilizados en la actualidad y ejecutarlos en casi todos los sistemas operativos.

MySQL utiliza el lenguaje de consulta estructurado (SQL), es rápido y una solución accesible para administrar correctamente los datos de una empresa. MySQL AB es la compañía responsable del desarrollo de MySQL, dispone de un sistema de asistencia eficiente y a un precio razonable, como ocurre con la mayor parte de las comunidades de código abierto, se puede encontrar una gran cantidad de ayuda en la Web.

Son muchas las razones para escoger a MySQL como una solución de misión crítica para la administración de datos:

- ✓ Asistencia: Existe una nutrida y activa comunidad MySQL.
- ✓ Velocidad: MySQL es mucho más rápido que la mayoría de sus rivales.

- ✓ Funcionalidad: MySQL dispone de muchas de las funciones que exigen los desarrolladores profesionales, como compatibilidad completa con ACID, compatibilidad para la mayor parte de SQL ANSI 21, volcados online, duplicación, funciones SSL e integración con la mayor parte de los entornos de programación.
- ✓ Portabilidad: MySQL se ejecuta en la inmensa mayoría de sistemas operativos y, la mayor parte de los casos, los datos se pueden transferir de un sistema a otro sin dificultad.
- ✓ Facilidad de uso: MySQL resulta fácil de utilizar y de administrar. Las herramientas de MySQL son potentes y flexibles, sin sacrificar su capacidad de uso. (18)

## 1.7 Herramientas Seleccionadas.

### 1.7.1 Macromedia Dreamweaver 8.

Dreamweaver es un editor visual profesional para la creación de sitios y páginas Web. Con Dreamweaver resulta fácil crear y editar páginas compatibles con cualquier explorador y plataforma.

Entre las novedades presentes en la versión 8 de Dreamweaver se puede destacar la nueva validación dinámica en distintos navegadores, es decir que las etiquetas HTML y las reglas CSS, vayas dónde vayas, siempre estarán legibles, siendo éstas compatibles con cualquier navegador del mercado. El soporte para CSS es más amplio y la edición gráfica, como por ejemplo rotar, recortar o redimensionar, integrada a la interfaz de Dreamweaver, sin tener que salir del programa.

El uso de SFTP, es decir el FTP seguro, ha sido implementado, así como el soporte para tecnologías más sofisticadas como ColdFusion, J2EE, PHP, .NET, espacios de nombre XML, objetos de control de formulario ASP.NET, nuevo contenido de referencia y comportamientos de servidor PHP. Pasar de Word o Excel a Dreamweaver, sin perder fuentes, colores y estilos es posible en esta nueva versión. Entre otras características, también se encuentra un editor de imagen integrado, diferentes colores para la sintaxis HTML, soporte para posicionamiento absoluto, poder hacer cambios por todas las páginas usando elementos comunes, cliente de FTP integrado (con soporte Firewall), soporte XML, plantillas, interfaz optimizada y personalizada, así como herramientas de codificación mejoradas. (19)

### 1.7.2 Zend Studio.

Se trata de un programa de la casa Zend, impulsores de la tecnología de servidor PHP, orientada a desarrollar aplicaciones web. 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.

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 la interfaz de edición y la ayuda. Permite además 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 o, en caso de que estén instalados, los configura para trabajar juntos en depuración.

- El editor.

La parte del programa que permite escribir los scripts es bastante útil para la programación en PHP. La interfaz está compuesta por varias partes, en las que se encuentra un explorador de archivos, una ventana de depuración, los menús y otra para mostrar el código de las páginas. Una de las ayudas que ofrece a la hora de escribir son las típicas en editores avanzados, como permitir editar varios archivos, y moverse fácilmente entre ellos, marcar a qué elementos corresponden los inicios y cierres de las etiquetas, paréntesis o llaves, moverse al principio o al final de una función, identificación automática del código, etc.

- La herramienta de depuración:

Gracias a ella es posible ejecutar páginas y conocer en todo momento el contenido de las variables de la aplicación y las variables del entorno como las cookies, las recibidas por formulario o en la sesión. (20)

### 1.7.3 SQL Manager for MySql.

EMS SQL Manager for MySQL es una herramienta de alto desempeño para administración y desarrollo en Servidor de Base de Datos MySQL, entre sus principales características se encuentran:

- ✓ Soporte completo para versiones MySQL desde la 3.23 hasta la 5.06.

- ✓ Nueva interfaz gráfica de usuario último modelo.
- ✓ Ágil navegación y administración de base de datos.
- ✓ Administración sencilla de todos los objetos de MySQL.
- ✓ Herramientas de manipulación de datos avanzada.
- ✓ Acceso al servidor MySQL a través del protocolo HTTP.
- ✓ Potente Administración de seguridad.
- ✓ Excelentes Herramientas visuales y de texto para elaboración de consultas.
- ✓ Impresionantes opciones de exportación e importación de datos.
- ✓ Diseñador Visual de Base de Datos completamente rediseñado.
- ✓ Asistentes fáciles de usar para efectuar servicios MySQL.

### 1.7.4 Visual Paradigm.

Visual Paradigm para UML es una herramienta CASE que soporta el ciclo de vida del desarrollo de un sistema informático, análisis y diseño orientado a objetos[22], construcción, pruebas y despliegue. Permite diseñar todos los tipos de diagramas que son necesarios en el análisis y diseño de cualquier sistema de software. Entre sus funcionalidades se incluyen la generación de código de clases modelos en diversos lenguajes tales como C++, Java, .NET, Perl, PHP, Python y archivos de esquema XML; además de utilizar técnicas de Ingeniería inversa de una base de datos existente y generar el código correspondiente a las clases en los lenguajes ya mencionados. También permite generar modelos de clases, tablas de base de datos y el código de la misma así como la documentación referente al proyecto de software que se modela.

Esta herramienta de modelado presenta características tales como:

- ✓ Soporte de UML versión 2.1.
- ✓ Modelado colaborativo con CVS y Subversión.

[22] Significa que el software se organiza como una colección de objetos discretos que contiene tanto estructura de datos como también un comportamiento.

- ✓ Editor de Detalles de Casos de Uso - Entorno todo-en-uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso.
- ✓ Diagramas de flujo de datos.
- ✓ Ingeniería inversa - Código a modelo, código a diagrama.
- ✓ Generación de código - Modelo a código, diagrama a código.
- ✓ Generación de código y despliegue de EJB´s - Generación de beans para el desarrollo y despliegue de aplicaciones.
- ✓ Generación de bases de datos - Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- ✓ Ingeniería inversa de bases de datos - Desde Sistemas Gestores de Bases de Datos (DBMS) existentes a diagramas de Entidad-Relación.
- ✓ Generador de informes de documentación.
- ✓ Integración con Visio – Dibujo de diagramas.

## Conclusiones

En este capítulo se analizaron las tecnologías y la metodología que se utilizan en el desarrollo de aplicaciones web y que son necesarias para la elaboración del sistema que se describe en el presente documento. Se elaboró además una descripción detallada de las herramientas escogidas para ser utilizadas y se describieron los lenguajes de programación, el gestor de bases de datos y el servidor Web que estarán presentes en la aplicación Web desarrollada.

# CAPÍTULO 2: ELEMENTOS DE ARQUITECTURA Y SEGURIDAD.

## Introducción.

En el presente capítulo se realiza la descripción de los requerimientos no funcionales del sistema. Además se realiza un análisis de los patrones arquitectónicos que se han tenido en cuenta para el desarrollo de este sistema, basados en las normas establecidas por la Facultad 7 de la Universidad de las Ciencias Informáticas en su Documento de Arquitectura de Software para productos orientados al Área de la Salud. También se describe la estrategia de seguridad trazada y se muestran las vistas de despliegue y de implementación.

## 2.1 Requerimientos No Funcionales del Sistema.

Los requerimientos no funcionales son cualidades o propiedades que un sistema informático debe poseer y que describen características o restricciones en la realización de funcionalidades y necesidades que de no ser tomadas en cuenta atentarían contra el desarrollo y posteriormente contra el uso del producto.

A continuación se listan los Requerimientos No Funcionales del módulo CCEMN.

| **Nº** | **RNF 1** |
|---|---|
| **Nombre** | Apariencia o Interfaz Externa. |
| **Descripción** | La interfaz del sistema no contiene numerosas imágenes para evitar demoras en la respuesta de cualquier acción del usuario. La misma será sencilla, amigable e intuitiva, de fácil navegación por parte del usuario. Estará diseñada para una visualización óptima en cualquier resolución de pantalla. |

| **Nº** | **RNF 2** |
|---|---|
| **Nombre** | Apariencia o Interfaz Interna. |
| **Descripción** | Los componentes del sistema serán desarrollados siguiendo el principio de Alta Cohesión y Bajo Acoplamiento, y los que sean reutilizables en los diferentes módulos del sistema serán desarrollados como servicios Web XML que interactuarán con otros componentes a través de SOAP. |

| Nº | RNF 3 |
|---|---|
| Nombre | Usabilidad |
| Descripción | La aplicación Web deberá facilitar la interacción usuario – sistema con el objetivo de evitar rechazo en el uso de la misma, y guiará mediante mensajes al usuario en las diferentes acciones que realice. El usuario deberá poseer conocimientos básicos del manejo de computadoras y estar familiarizado con la gestión y el flujo de la información en el área para la cual se desarrolla el sistema informático. |

| Nº | RNF 4 |
|---|---|
| Nombre | Portabilidad |
| Descripción | El sistema informático está desarrollado sobre una Plataforma Web por lo que podrá ser utilizado sobre cualquier Sistema Operativo, recomendándose para su uso Windows o distribuciones de Linux. |

| Nº | RNF 5 |
|---|---|
| Nombre | Rendimiento |
| Descripción | El tiempo de respuesta de una petición al servidor y la velocidad de procesamiento de la información deben ser rápidas para evitar retrasos en la actualización de datos, así como en la toma de decisiones. |

| Nº | RNF 6 |
|---|---|
| Nombre | Soporte |
| Descripción | La aplicación Web contará con una ayuda donde el usuario podrá suplir las dudas que se le puedan presentar durante la utilización de la misma. El usuario del módulo deberá recibir un adiestramiento previo en la utilización del sistema con el fin de que pueda explotar las prestaciones del sistema sin contratiempos ocasionados por la falta de preparación técnica. |

| **Nº** | **RNF 7** |
|---|---|
| **Nombre** | Seguridad |
| **Subcategoría Nº** | **7.1** |
| **Nombre** | Confidencialidad |
| **Descripción** | La información que brinda el sistema estará protegida contra el acceso de usuarios no autorizados. Solamente los administradores del sistema podrán realizar cambios en la configuración y en la información. |
| **Subcategoría Nº** | **7.2** |
| **Nombre** | Disponibilidad |
| **Descripción** | El sistema estará accesible cada vez que los usuarios del mismo lo requieran. |

| **Nº** | **RNF 8** |
|---|---|
| **Nombre** | Software |
| **Subcategoría Nº** | **8.1** |
| **Nombre** | Software en el Servidor |
| **Descripción** | El servidor deberá contar con el sistema operativo Linux/Debian 4 Etch y con el servidor Web Apache. |
| **Subcategoría Nº** | **8.2** |
| **Nombre** | Software en el Cliente |
| **Descripción** | Para utilizar la aplicación Web será necesaria una computadora con el Sistema Operativo Windows o Linux, recomendándose Windows XP o superior y Ubuntu 7.10 o Superior. Además se podrá acceder desde los navegadores Web Internet Explorer 6 o superior y Firefox 2.0 o superior. |

| **Nº** | **RNF 9** |
|---|---|
| **Nombre** | Hardware |
| **Descripción** | Para garantizar el correcto funcionamiento de la aplicación se necesitan como requerimientos mínimos una computadora con un procesador Pentium II o superior, una memoria RAM de 128 MB o más, un disco duro de 10 GB o |

| | |
|---|---|
| | más y una tarjeta de red a 100 Mbps o más. |

| | |
|---|---|
| **Nº** | **RNF 10** |
| **Nombre** | Restricciones en el Diseño y la Implementación. |
| **Descripción** | • Se utilizarán los patrones de diseño y de codificación establecidos.<br>• La lógica de presentación constituirá una capa independiente de la lógica de negocio, centrando su función en la interfaz de usuario y validaciones simples de la entrada de datos.<br>• Para el análisis y el diseño del sistema deberá ser utilizada la metodología RUP, usando el lenguaje de modelado UML y como herramienta CASE el Visual Paradigm.<br>• Para la implementación se utilizará como lenguaje de programación PHP 5 y como herramienta de desarrollo el Zend Studio. |

## 2.2 Patrones o Estilos Arquitectónicos utilizados.

Un patrón de arquitectura expresa una organización estructural o esquema de sistemas de software y de la base para el fundamento estructural de una aplicación ya que no sólo divide el problema en partes, si no que también asigna a cada parte un rol, una serie de tareas de las cuales ocuparse.

### 2.2.1 Arquitectura Orientada a Servicios (SOA).

SOA es un paradigma de integración que permite diseñar y construir sistemas que serán flexibles, escalables y reutilizables. Además proporciona una metodología y un marco de trabajo para documentar las capacidades de negocio y puede dar soporte a las actividades de integración y consolidación.

En un ambiente SOA, los nodos de la red hacen disponibles sus recursos a otros participantes en la red como servicios independientes a los que tienen acceso de un modo estandarizado. La mayoría de las definiciones de SOA identifican la utilización de servicios Web (empleando SOAP y WSDL) en su implementación, no obstante se puede implementar una SOA utilizando cualquier tecnología basada en servicios.

Para mencionarlo de una forma mas simplificada SOA es un concepto de arquitectura de software que define la utilización de servicios para dar soporte a los requerimientos de software del usuario.

En el módulo CCEMN se utilizan elementos de SOA debido a la necesidad de reutilizar información proporcionada por otros componentes a través de servicios Web brindados por los mismos, facilitando de esta manera la integración de los sistemas de gestión a una plataforma de servicios única.

### 2.2.2 Arquitectura Basada en Componentes (CBA).

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.

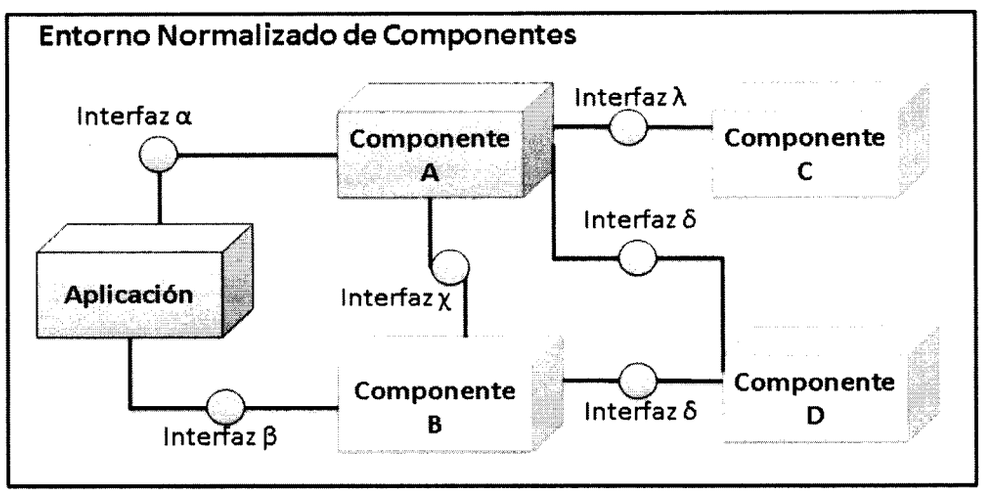


**Figura 3: Modelo de Arquitectura Basada en Componentes.**

En la tecnología de componentes la interfaz constituye el elemento básico de interoperabilidad. Cada componente debe describir de forma completa las interfaces que ofrece, así como las interfaces que requiere para su operación. Y debe operar correctamente con independencia de los mecanismos internos que utilice para soportar la funcionalidad de la interfaz.

Características muy relevantes de la tecnología de programación basada en componentes son la modularidad y la reusabilidad, y en todos ellos coincide con la tecnología orientada a objetos de la que se puede considerar una evolución. Sin embargo, en la tecnología basada en componentes también se

requiere robustez ya que los componentes han de operar en entornos mucho más heterogéneos y diversos.

El desarrollo de software basado componentes es la evolución natural de la ingeniería software para mejorar la calidad, disminuir los tiempos de desarrollo y gestionar la creciente complejidad de los sistemas. Esta arquitectura proporciona como beneficios:

- ✓ Reusabilidad de Servicios: Reducción considerable de tiempo y costos de desarrollo de aplicaciones al utilizar servicios disponibles ya desarrollados, para resolver problemáticas comunes a otras aplicaciones, aumentando la robustez del nuevo sistema al utilizarse software ya probado.
- ✓ Interoperabilidad de Aplicaciones: Disminución de la complejidad en el proceso de integración, pues interactúa con elementos que se abstraen de la tecnología y ubicación de los servicios. (21)

### 2.2.3 Patrón de Arquitectura de Software Modelo-Vista-Controlador (MVC).

El patrón conocido como Modelo-Vista-Controlador (MVC) separa el modelado del dominio, la presentación y las acciones basadas en datos ingresados por el usuario en tres clases diferentes:

- ✓ Modelo. El modelo administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista) y responde a instrucciones de cambiar el estado (habitualmente desde el controlador).
- ✓ Vista. Maneja la visualización de la información.
- ✓ Controlador. Interpreta las acciones del ratón y el teclado, informando al modelo y/o a la vista para que cambien según resulte apropiado.

Tanto la vista como el controlador dependen del modelo, el cual no depende de las otras clases. Esta separación permite construir y probar el modelo independientemente de la representación visual.

Entre las ventajas del patrón de arquitectura están las siguientes:

- ✓ Soporte de vistas múltiples. Dado que la vista se halla separada del modelo y no hay dependencia directa del modelo con respecto a la vista, la interfaz de usuario puede mostrar múltiples vistas de los mismos datos simultáneamente. Por ejemplo, múltiples páginas de una aplicación Web pueden utilizar el mismo modelo de objetos, mostrado de maneras diferentes.

- ✓ Adaptación al cambio. Los requerimientos de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas de negocios. Los usuarios pueden preferir distintas opciones de representación, o requerir soporte para nuevos dispositivos de hardware. Dado que el modelo no depende de las vistas, agregar nuevas opciones de presentación generalmente no afecta al modelo.

Entre las desventajas, se han señalado:

- ✓ Complejidad. El patrón introduce nuevos niveles de indirección y por lo tanto aumenta ligeramente la complejidad de la solución. También se profundiza la orientación a eventos del código de la interfaz de usuario, que puede llegar a ser difícil de depurar.
- ✓ Costo de actualizaciones frecuentes. Desacoplar el modelo de la vista no significa que los desarrolladores del modelo puedan ignorar la naturaleza de las vistas.
- ✓ Si el modelo experimenta cambios frecuentes, por ejemplo, podría desbordar las vistas con una lluvia de requerimientos de actualización.

Se han desarrollado a lo largo de los años, desde la presentación de este patrón a la comunidad científica 3 variantes fundamentales, que se presentan brevemente a continuación.

1. Variante I: (Figura 4)

Variante en la cual no existe ninguna comunicación entre el Modelo y la Vista y esta última recibe los datos a mostrar a través del Controlador.

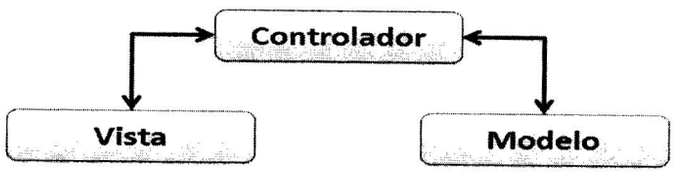


**Figura 4: Variante inicial del Patrón MVC.**

2. Variante II: (Figura 5)

Variante en la cual se desarrolla una comunicación entre el Modelo y la Vista, donde esta última al mostrar los datos la busca directamente en el Modelo, dada una indicación del Controlador, disminuyendo el conjunto de responsabilidades de este último.

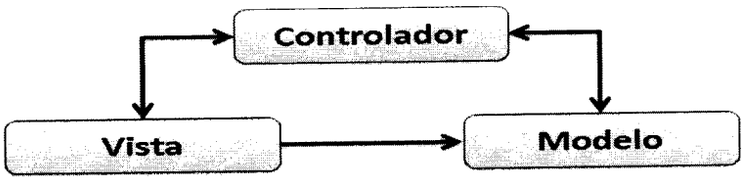


**Figura 5: Variante Intermedia del Patrón MVC.**

3. Variante III: (Figura 6)

Variante en la cual se diversifica las funcionalidades del Modelo teniendo en cuanta las características de las aplicaciones multimedia, donde tienen un gran peso las medias utilizadas en estas.

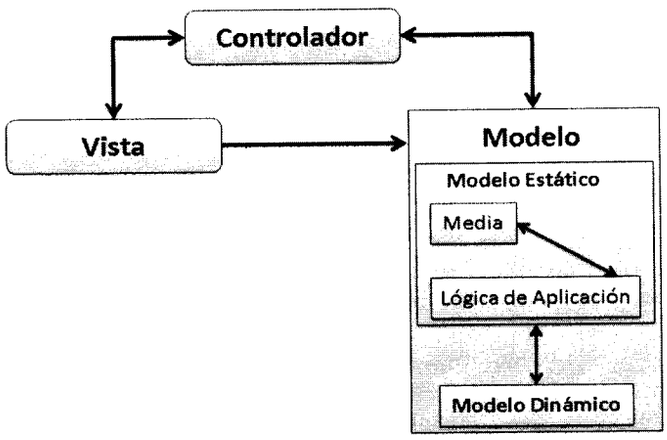


**Figura 6: Variante modificada para Aplicaciones Multimedia del MVC, conocida como MVCmm.**

Aunque se pueden encontrar diferentes implementaciones de MVC, el flujo que sigue el control generalmente es el siguiente:

1. El usuario interactúa con la interfaz de usuario de alguna forma (por ejemplo, el usuario pulsa un botón, enlace).

2. El controlador recibe (por parte de los objetos de la interfaz-vista) la notificación de la acción solicitada por el usuario. El controlador gestiona el evento que llega, frecuentemente a través de un gestor de eventos (handler) o callback.

3. El controlador accede al modelo, actualizándolo, posiblemente modificándolo de forma adecuada a la acción solicitada por el usuario. Los controladores complejos están a menudo

estructurados usando un patrón de comando que encapsula las acciones y simplifica su extensión.

4. El controlador delega a los objetos de la vista la tarea de desplegar la interfaz de usuario. La vista obtiene sus datos del modelo para generar la interfaz apropiada para el usuario donde se refleja los cambios en el modelo (si se utiliza la variante 2 descrita anteriormente, de lo contrario lo obtiene a través del Controlador).

   El modelo no debe tener conocimiento directo sobre la vista. Sin embargo, el patrón de observador puede ser utilizado para proveer cierta indirección entre el modelo y la vista, permitiendo al modelo notificar a los interesados de cualquier cambio. Un objeto vista puede registrarse con el modelo y esperar a los cambios, pero aun así el modelo en sí mismo sigue sin saber nada de la vista. El controlador no pasa objetos de dominio (el modelo) a la vista aunque puede dar la orden a la vista para que se actualice. Nota: En algunas implementaciones la vista no tiene acceso directo al modelo, dejando que el controlador envíe los datos del modelo a la vista, según lo descrito en la segunda variante.

5. La interfaz de usuario espera nuevas interacciones del usuario, comenzando el ciclo nuevamente. (22)

### 2.2.4 Modelo Cliente – Servidor.

Es la tecnología que proporciona al usuario final el acceso transparente a las aplicaciones, datos, servicios de cómputo o cualquier otro recurso del grupo de trabajo y/o, a través de la organización, en múltiples plataformas. El modelo soporta un medio ambiente distribuido en el cual los requerimientos de servicio hechos por estaciones de trabajo inteligentes o "clientes", resultan en un trabajo realizado por otros computadores llamados servidores".

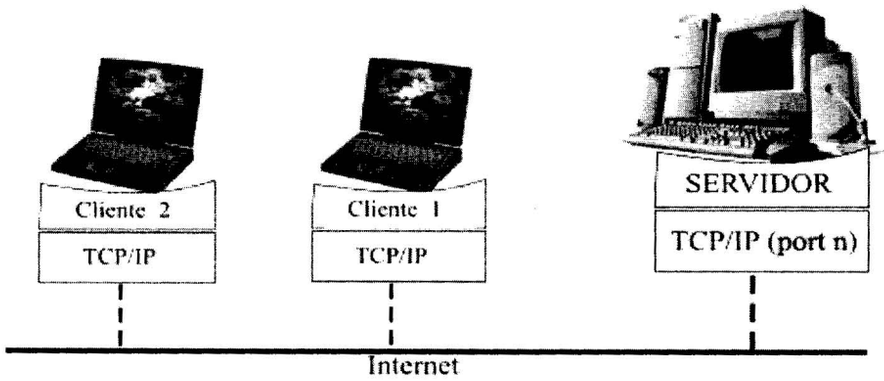


**Figura 7: Modelo Cliente – Servidor**

En este modelo las aplicaciones se dividen de forma que el servidor contiene la parte que debe ser compartida por varios usuarios, y en el cliente permanece sólo lo particular de cada usuario. Siempre que un cliente requiere un servicio lo solicita al servidor correspondiente y éste le responde proporcionándolo. Normalmente, pero no necesariamente, el cliente y el servidor están ubicados en distintos procesadores. Los clientes se suelen situar en ordenadores personales y/o estaciones de trabajo y los servidores en procesadores departamentales o de grupo.

Los clientes realizan generalmente funciones como:

- ✓ Manejo de la interfaz de usuario
- ✓ Captura y validación de los datos de entrada.
- ✓ Generación de consultas e informes sobre las bases de datos.

Por su parte los servidores realizan, entre otras, las siguientes funciones:

- ✓ Gestión de periféricos compartidos.
- ✓ Control de accesos concurrentes a bases de datos compartidas.
- ✓ Enlaces de comunicaciones con otras redes de área local o extensa.

Entre las principales características de la arquitectura cliente/servidor se pueden destacar las siguientes:

- ✓ El servidor presenta a todos sus clientes una interfaz única y bien definida.
- ✓ El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- ✓ El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- ✓ Los cambios en el servidor implican pocos o ningún cambio en el cliente. (23) (24)

## 2.3 Estrategia de Seguridad del Sistema.

Para la implementación de la seguridad del módulo CCEMN se realizó un profundo análisis de las diferentes formas en que se podría acceder a las funcionalidades que brinda el sistema y la manera en que se podría evitar que usuarios o personas sin autorización pudieran acceder a las mismas y causar dificultades con la configuración o con los datos almacenados. El framework (CodeIgniter) utilizado en la construcción del sistema informático basa su funcionamiento en la llamada a las diferentes funcionalidades por direcciones URL, las cuales se encargan de ejecutar las peticiones o acciones del usuario o del sistema en sí mismo.

Teniendo en cuenta lo antes mencionado se estableció como política crear un conjunto de roles, que se le asignarían a los usuarios de acuerdo al nivel de acceso que tendrían en el sistema. Dichos roles presentan asociados un conjunto de direcciones URL que describen las funcionalidades del sistema a las que un rol tendrá acceso. Estas relaciones que se establecen entre roles, usuarios, direcciones URL y funcionalidades son manejadas a nivel de base de datos con el fin de elevar el nivel de abstracción del proceso de implementación en lo referente al tema de la seguridad.

La estrategia de seguridad se describe de forma simplificada en el siguiente diagrama:

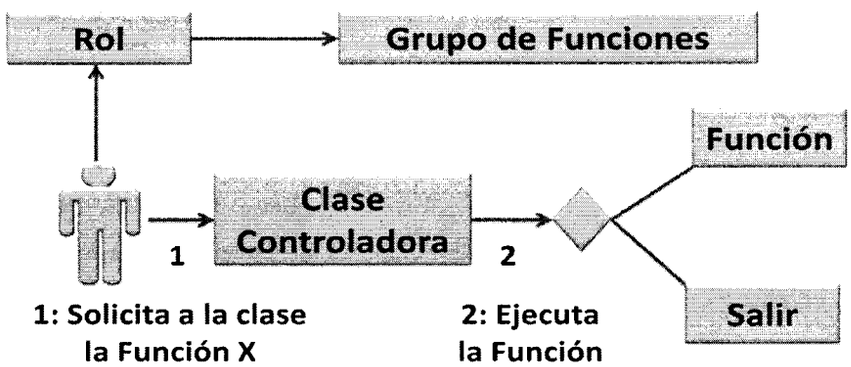


**Figura 8: Modelo del Funcionamiento de la Seguridad empleada en el Sistema**

## 2.4 Modelo de Implementación.

El Modelo de Implementación describe cómo los elementos del diseño se implementan en términos de componentes, ejemplo los ficheros de código fuente, ejecutables, etc. También establece la organización de los componentes de acuerdo con los mecanismos de estructuración y modularización que rigen el entorno de implementación y en el lenguaje de programación utilizado; y define la dependencia existente entre los componentes. Además, establece una jerarquía en la que se organiza el modelo de implementación en subsistemas con la finalidad de hacerlo más manejable.

### 2.4.1 Diagramas de Componentes.

Según UML la definición de componente está dada como: Parte modular de un sistema, desplegable y reemplazable que encapsula implementación y un conjunto de interfaces y proporciona la realización de los mismos. Un componente típicamente contiene clases y puede ser implementado por uno o más artefactos (ficheros ejecutables, binarios, etc.).

El Diagrama de Componentes muestra un conjunto de elementos del modelo tales como componentes, subsistemas de implementación y sus relaciones. Se utiliza para modelar la vista estática de un sistema y muestra la organización y las dependencias lógicas entre un conjunto de componentes software, sean éstos componentes de código fuente, librerías, binarios o ejecutables.
El uso más importante de estos diagramas es mostrar la estructura de alto nivel del modelo de implementación, especificando:

- ✓ Los subsistemas de implementación y sus dependencias a la hora de importar código.
- ✓ Organizar los subsistemas de implementación en capas.

También se utilizan para mostrar las dependencias de compilación de los ficheros de código, relaciones de derivación entre ficheros de código fuente y ficheros que son resultados de la compilación, dependencias entre elementos de implementación y los correspondientes elementos de diseños que son implementados.

La aplicación Web que motiva el presente documento contará con 5 subsistemas de implementación básicos encapsulados dentro del subsistema que constituye el módulo CCEMN. Los mismos son:

- ✓ El Subsistema de Clases Controladoras (figura 10), que contendrá todas las clases que manipulan los eventos del usuario y realizan peticiones al Subsistema Modelo para mostrarlas en las vistas.
- ✓ El Subsistema de Clases Modelo (figura 11), agrupa las clases que interactúan con la base de datos y velan por el cumplimento de las reglas del negocio.
- ✓ El Subsistema de Vistas (figura 12), agrupa los componentes que permiten la interacción directa con el usuario final del sistema y muestran y recogen información a estos usuarios.
- ✓ El Subsistema SISalud (figura 13), que contendrá los componentes con los que se comunica el Subsistema de Clases Modelos, tales como el Registro de Ciudadanos, el Registro de Ubicación, el Registro de Unidades de Salud, el SAAA y el Registro Personal de la Salud (sobre los componentes mencionados se abunda en el Capítulo 3 del presente documento).
- ✓ El Subsistema Operaciones contendrá el componente Operaciones del Sistema Informatizado para el Centro Nacional de Urgencias Médicas (SIUM) y que se comunica con el Subsistema de Clases Modelos.

En el Diagrama de Componentes de la figura 9 se observan los subsistemas que componen el módulo CCEMN y las relaciones entre los mismos.

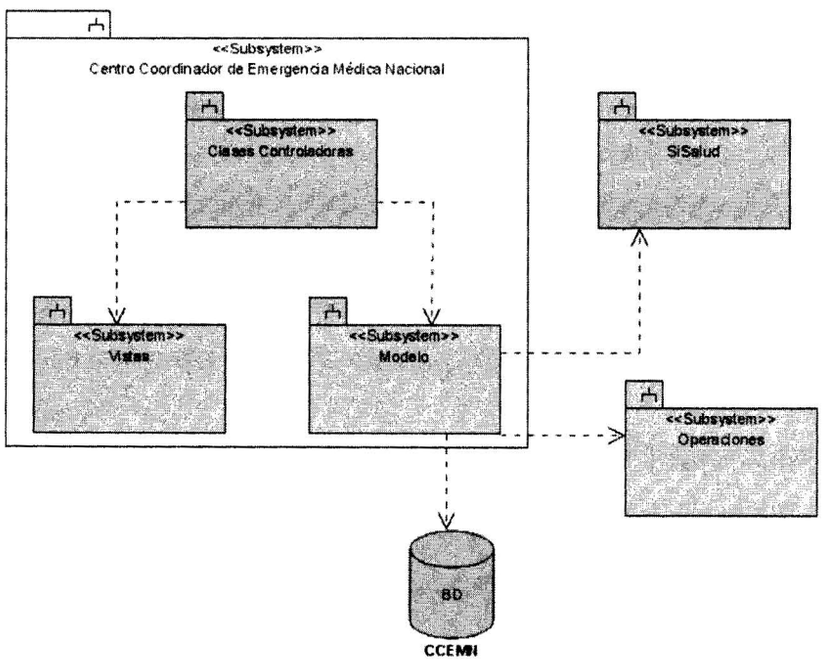


**Figura 9: Diagrama de Componentes.**

En las figuras siguientes se muestran los subsistemas de una forma más detallada:

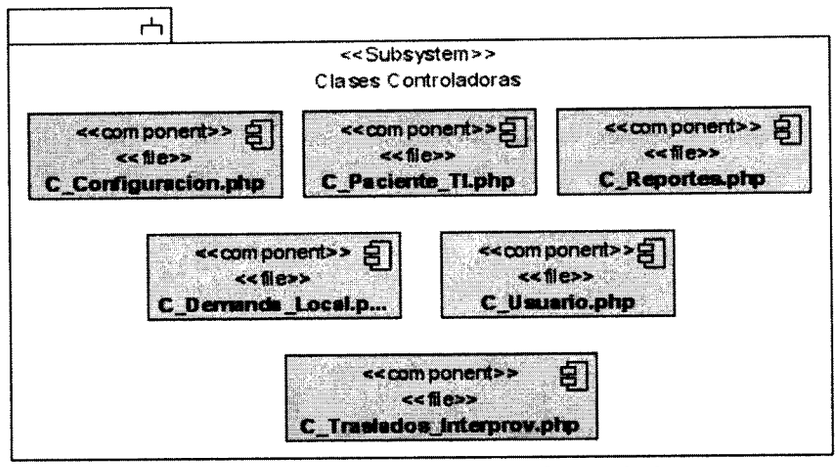


**Figura 10: Subsistema de Clases Controladoras.**

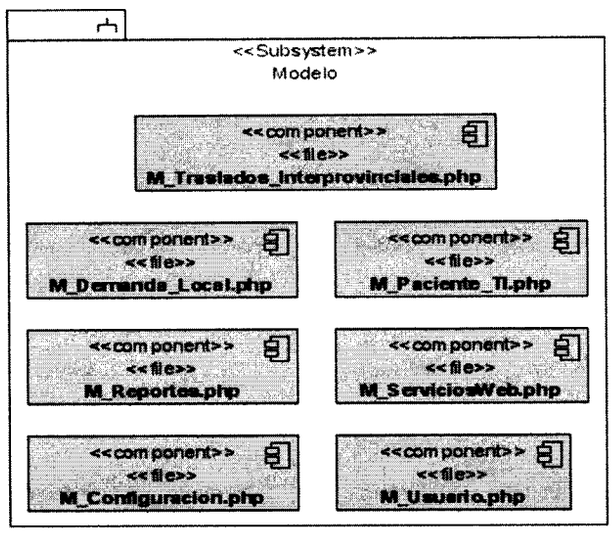


**Figura 11: Subsistema de Clases Modelos.**

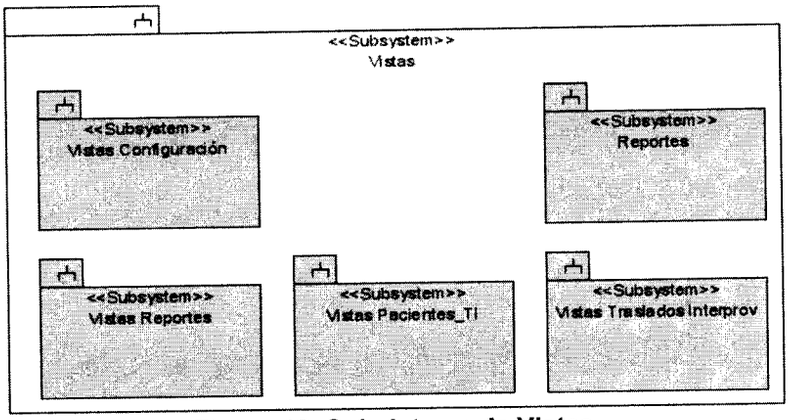


**Figura 12: Subsistema de Vistas.**

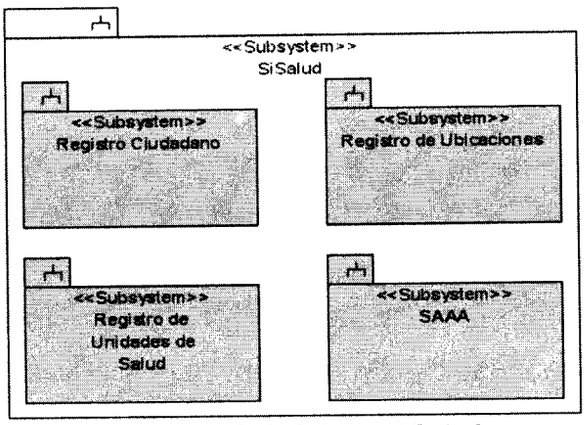


**Figura 13: Subsistema SISalud.**

A continuación se describen detalladamente cada uno de los subsistemas que componen las vistas del software:

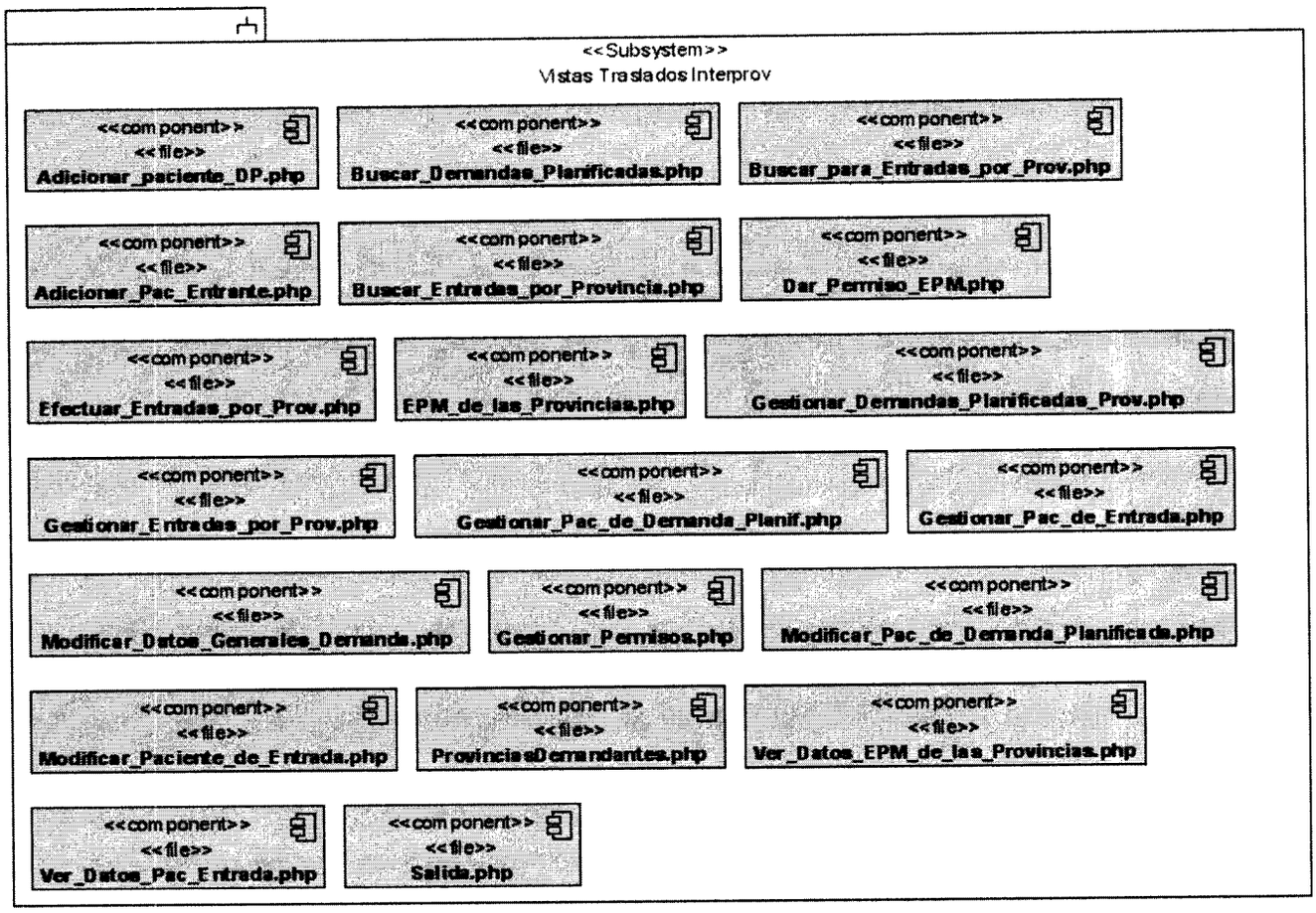


**Figura 14: Subsistema Vistas de Traslados Interprovinciales.**

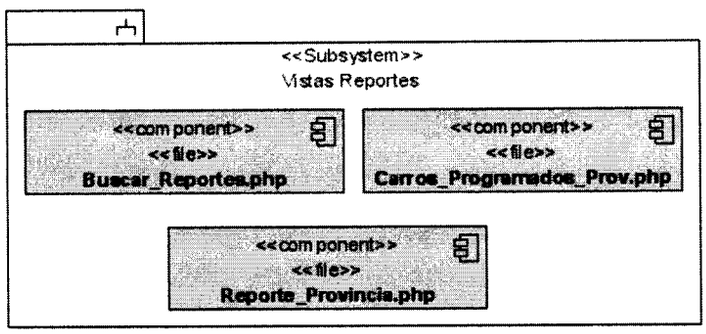


**Figura 15: Subsistema Vistas de Reportes.**

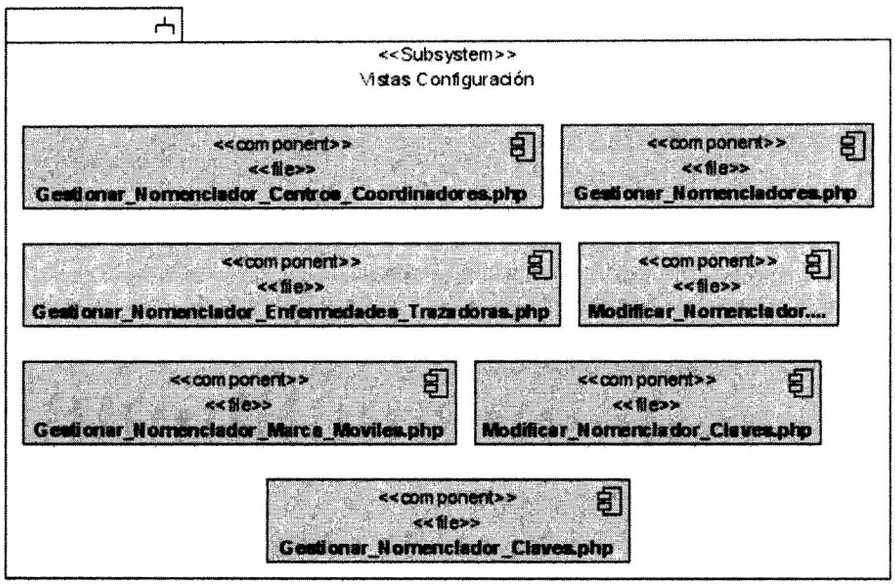


**Figura 16: Subsistema Vistas de Configuración.**

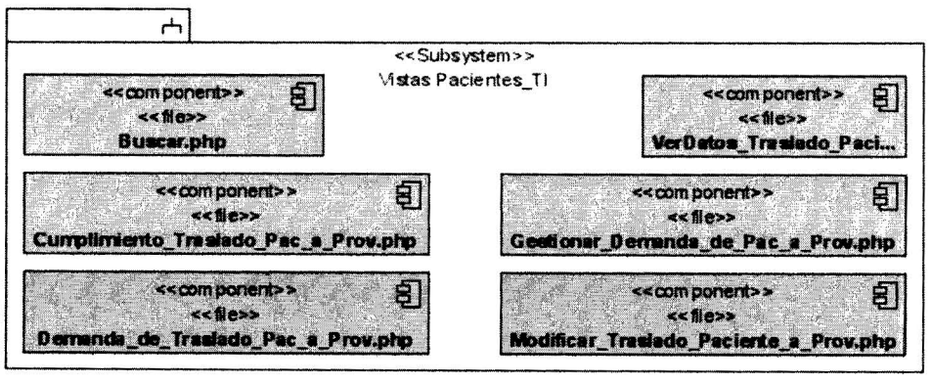


**Figura 17: Subsistema Vistas de Pacientes para Traslados Interprovinciales.**

En el Diagrama de la figura 18 se ilustran de forma detallada las relaciones existentes entre los componentes de los subsistemas a modo general. En el mismo Diagrama se puede apreciar de forma clara en la práctica como se trabaja con el patrón arquitectónico MVC en el desarrollo del sistema informático:

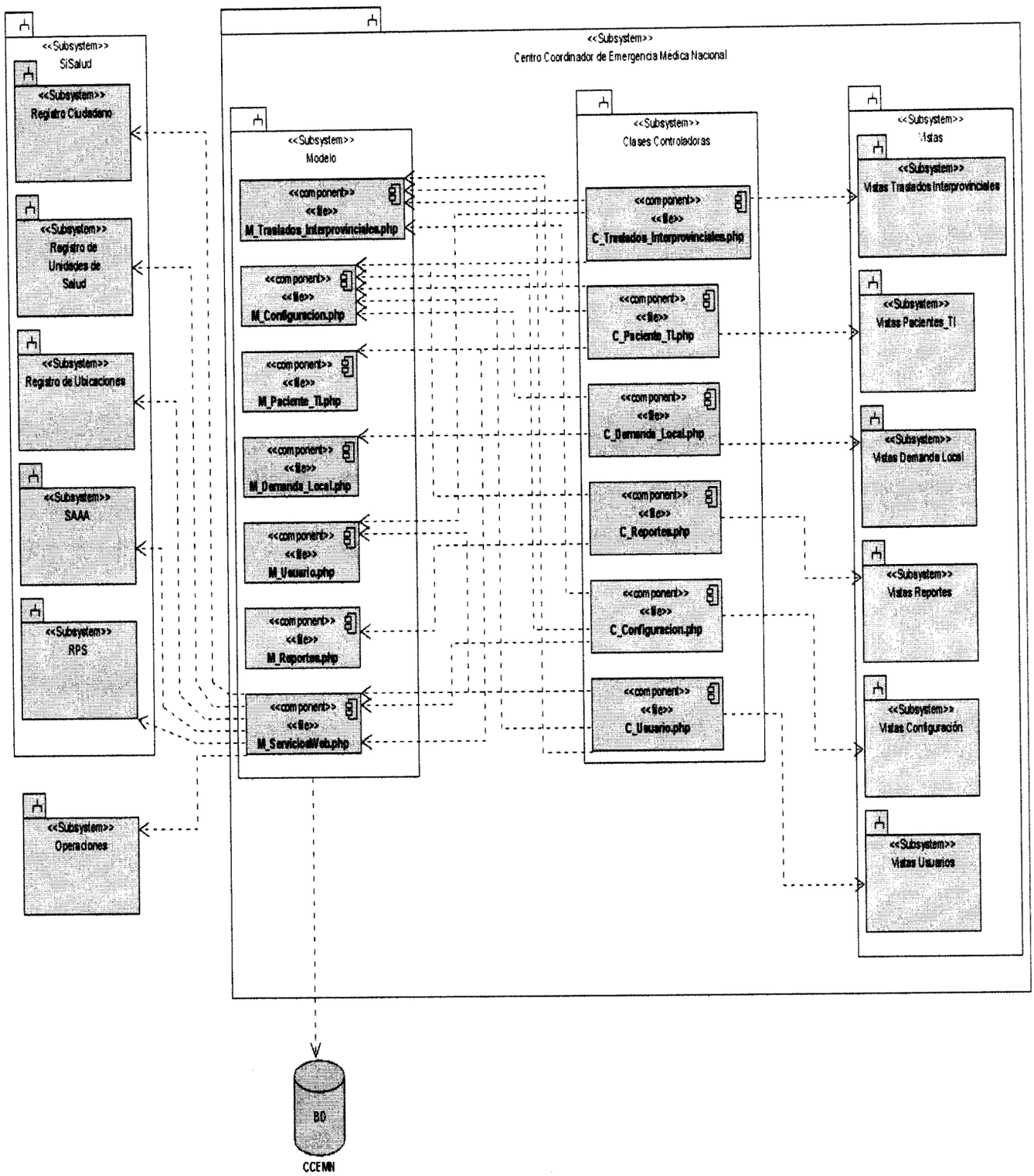


**Figura 18: Relaciones entre los Componentes.**

## 2.5 Diagrama de Despliegue.

El diagrama de despliegue es un modelo de objetos que describe la topología de la arquitectura física del sistema por medio de la distribución de funcionalidades entre nodos interconectados. Estos nodos y sus enlaces son elementos de hardware sobre los cuales puede ejecutarse el software, donde cada nodo representa un dispositivo de procesamiento y cada enlace representa los mecanismos de comunicación que se establecen entre dichos nodos.

Básicamente este modelo es una representación gráfica de la correspondencia entre la arquitectura del software y la arquitectura del sistema (hardware). En la figura 19 se muestra la vista del Diagrama de Despliegue del módulo CCEMN.

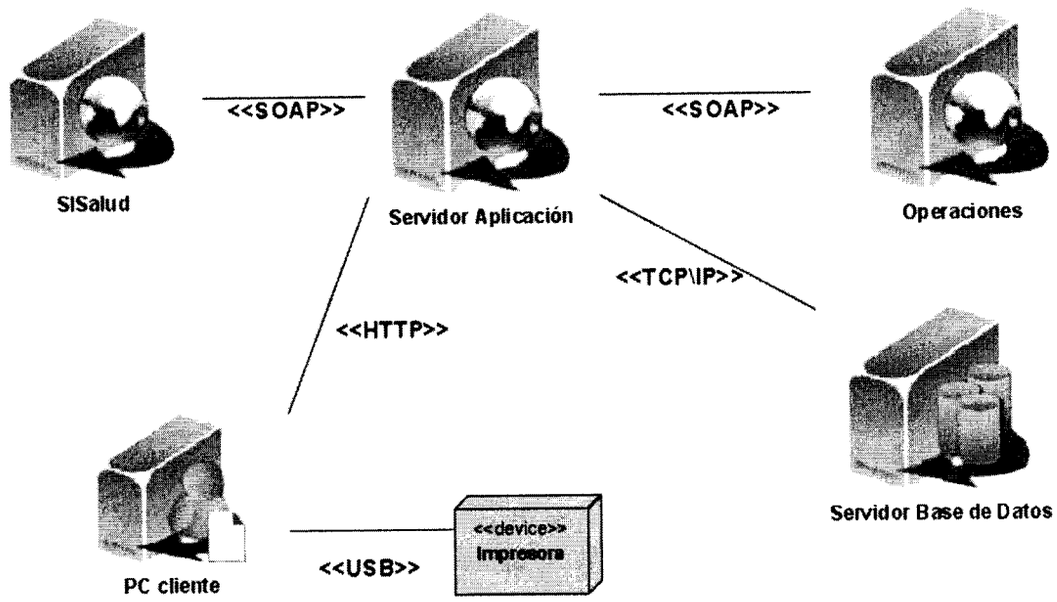


**Figura 19: Diagrama de Despliegue.**

## Conclusiones

Con este capítulo se ha realizado una descripción detallada de los Requerimientos no Funcionales del Sistema, se ha realizado una descripción y análisis de los patrones arquitectónicos. También se ha descrito la estrategia de seguridad que se ha elegido para el sistema informático y se muestran los diagramas de componentes y de despliegue, mencionando una descripción de los mismos.

# CAPÍTULO 3: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA.

## Introducción.

En el presente capítulo se realiza un análisis de los componentes con los que se integra el módulo CCEMN y las relaciones específicas que tiene con los mismos. Además se realiza una descripción de las clases y los métodos que las mismas poseen. Se especifica también el modelo de datos y se describen las tablas de la base de datos. Se realiza el modelo de componentes.

## 3.1 Dependencias y Relaciones con otros Sistemas.

La reutilización del software ha sido, y sigue siendo, uno de los principales temas de investigación en el campo de la Ingeniería del Software, citándose a menudo como una de las principales técnicas para incrementar la productividad de los desarrolladores de software. Teniendo en cuenta este principio el módulo CCEMN hace uso de una arquitectura basada en componentes y orientada a servicios, donde cada módulo es un componente con funcionalidad propia y que utiliza los servicios brindados por otros componentes, lo que trae consigo que la puesta en marcha de este sistema dependa estrictamente del funcionamiento de otros componentes pertenecientes al SISalud y al Sistema Informatizado para el Centro Nacional de Urgencias Médicas (SIUM), los cuales se detallan a continuación.

### 3.1.1 Componente de Seguridad (SAAA)

SAAA, Single Authentication Authorization and Account. Este componente tiene implementado un mecanismo de seguridad basado en el modelo de Autenticación, Autorización y Auditoria. La autenticación es la primera acción del usuario en el sistema y consiste en suministrar un nombre de usuario único y una contraseña. Si el usuario no se encuentra registrado se reportara un error de acceso. Si el usuario autenticado se encuentra registrado se autoriza su acceso y se crea un certificado digital que contiene un identificador único (token) de 32 caracteres, que tiene como información: el identificador del usuario, el nivel de acceso (Nacional, Provincial, Municipal o Unidad de Salud), el identificador de nivel de acceso, un listado de los módulos a los que el usuario tiene acceso y el tipo de acceso que tiene en cada uno de ellos.

La autenticación y autorización en el módulo CCEMN se implemento utilizando el SAAA debido a la política de integración que deben de tener todos los sistemas informáticos producidos en la UCI para la salud. El certificado digital que brinda dicho componente una vez autenticado el usuario, es empleado

para poder consumir información brindada por otros componentes, lo cuales hacen uso de este certificado para brindar su información.

### 3.1.2 Registro de Ubicación (RU).

El RU es uno de los componentes no médicos del Registro Informatizado de Salud. El mismo gestiona la información de las Provincias, Municipios, Calles y Manzanas del país. Los servicios que ofrece este componente se utilizan en el Módulo CCEMN para obtener los identificadores de las provincias y sus municipios con el fin de poder hacer uso de las instituciones de la salud que son brindadas por el componente Registro de Unidades de Salud (RUS). Se utiliza además con la finalidad de establecer las localidades de origen y destino de cualquier demanda de traslado interprovincial proporcionando de esta manera un criterio de búsqueda en el momento de gestionar toda la información de dicha demanda.

### 3.1.3 Registro de Unidades de Salud (RUS).

Este registro tiene almacenada toda la información correspondiente a las unidades de salud de todo el país. Brinda un conjunto de servicios Web que facilitan la búsqueda de información relacionada con los procesos de negocio que se gestionan en este codificador. Uno de los servicios Web que brinda es el que permite la búsqueda de las unidades de salud. La información que ofrece este servicio se realiza teniendo en cuenta un conjunto de parámetros de entrada como el nombre, la ubicación geográfica (provincia y municipio) o los atributos cualitativos de las unidades de salud. El módulo CCEMN se comunica con el RUS para definir las instituciones de la salud a las que se trasladarán los pacientes.

### 3.1.4 Registro de Personal de la Salud (RPS).

Este componente tiene la funcionalidad de gestionar la información de todos los profesionales de la salud del país, ya sean médicos o no médicos. En las Áreas de Salud se encuentran los Grupos Básicos de Trabajo y Equipos Básicos de Salud. Los mismos están compuestos por especialistas y profesionales de la salud, que ejercen sus funciones para prevenir y resolver los problemas de la salud de la población cubana.

El RPS tiene implementado un conjunto de servicios Web que brindan la información que se procesa en este componente. Estos servicios son utilizados por el Módulo CCEMN como una forma de establecer la confirmación de la identidad del personal autorizado a realizar solicitudes de traslado de un paciente a determinada provincia.

### 3.1.5 Registro del Ciudadano (RC).

Este componente maneja toda la información correspondiente a los ciudadanos cubanos. Brinda un conjunto de servicios Web que facilitan la búsqueda de información relacionada a cualquier persona y datos específicos de la misma. En el módulo CCEMN se utilizan algunos de los datos que el mismo brinda, tales como nombre, apellidos y su identificador de ciudadano para adicionar los datos de los usuarios del sistema.

### 3.1.6 Registro de Operaciones del Sistema Informatizado de Urgencias Médicas.

Este registro almacena toda la información referente a las Bases de Ambulancias de todo el país. Brinda los datos de los móviles que son utilizados por los diferentes CCEMP y el CCEMN con el fin de facilitar la asignación de los mismos a las diferentes actividades que se atienden en lo centros mencionados.

## 3.2 Descripción de las Clases Empleadas en la Implementación.

| **Nombre: C_Usuario** | |
|---|---|
| **Tipo de clase: Controladora** | |
| **Para cada responsabilidad**: | |
| Nombre: | AutenticateSAAA() |
| Descripción: | Es utilizado para que el usuario se autentique haciendo uso de la funcionalidad autenticar del componente SAAA perteneciente al SISalud. |
| | |
| Nombre: | Salir() |
| Descripción: | Se utiliza para que el usuario termine su sesión en el sistema. |
| | |
| Nombre: | AdicionarUsuario() |
| Descripción: | Se encarga de adicionar un usuario a la aplicación. |
| | |
| Nombre: | CriteriosPersona() |
| Descripción: | Se utiliza para establecer los criterios de búsqueda empleados en buscar un |

| | |
|---|---|
| | ciudadano en el componente Registro de Ciudadanos (RC). |
| | |
| Nombre: | BuscarPersona() |
| Descripción: | Se encarga de buscar un ciudadano en el componente Registro de Ciudadanos (RC). |
| | |
| Nombre: | CriterioConfigurarRol() |
| Descripción: | Se encarga de establecer los criterios de búsqueda en el momento de listar los roles de determinado usuario. |
| | |
| Nombre: | EliminarRol() |
| Descripción: | Se encarga de eliminar un rol que posea determinado usuario. |
| | |
| Nombre: | EliminarUsuario() |
| Descripción: | Se encarga de eliminar determinado usuario de la aplicación. |
| | |
| Nombre: | InsertarRol() |
| Descripción: | Se encarga de insertar un rol a determinado usuario. |
| | |
| Nombre: | LevantaBuscarCiudadano() |
| Descripción: | Se utiliza para cargar la interfaz de usuario de Buscar Ciudadano. |
| | |
| Nombre: | ListarRolesUsuarios() |
| Descripción: | Se encarga de listar los roles que posee determinado usuario. |
| | |
| Nombre: | ListarUsuarios() |
| Descripción: | Se encarga de listar los usuarios que posee la aplicación. |

| **Nombre: M_Usuario** | |
|---|---|
| **Tipo de clase: Modelo** | |
| **Para cada responsabilidad**: | |
| Nombre: | GetRoles4User($idpersona) |
| Descripción: | Dado el identificador de un usuario devuelve los roles que tiene asociado. |
| | |

| | |
|---|---|
| Nombre: | IsUserInRol($idpersona,$rol) |
| Descripción: | Esta función verifica si un rol pertenece a un determinado usuario. |
| | |
| Nombre: | GetAllUsers() |
| Descripción: | Retorna un arreglo con los nombres de usuarios que tienen acceso al sistema. |
| | |
| Nombre: | GetLinks4User($idpersona) |
| Descripción: | Se utiliza para cargar los enlaces a los que está autorizado a acceder un usuario por su rol y construir el menú del que este dispondrá. |
| | |
| Nombre: | HaveAutorization($idpersona,$link) |
| Descripción: | Se utiliza para implementar la seguridad del sistema, ya que indica si un usuario puede tener acceso a determinada URL. |
| | |
| Nombre: | GetUser($idpersona) |
| Descripción: | Se utiliza para dado el id en el Registro de Ciudadanos (RC) de una persona retornar los datos de la misma que están almacenados en la base de datos. |
| | |
| Nombre: | Model_AdicionarUsuario($id_siudadano,$result) |
| Descripción: | Se utiliza para acceder a la base de datos e insertar los datos de un usuario. |
| | |
| Nombre: | Model_EliminarRol($idrol,$iduser) |
| Descripción: | Se utiliza para acceder a la base de datos y eliminar un rol que posea determinado usuario. |
| | |
| Nombre: | Model_EliminarUsuario($id_user) |
| Descripción: | Se utiliza para acceder a la base de datos y eliminar un usuario. |
| | |
| Nombre: | Model_InsertarRol($idrol,$iduser) |
| Descripción: | Accede a la base de datos e inserta un Rol a determinado usuario. |
| | |
| Nombre: | Model_Query_ListarUsuario($tabla,$per_page,$offset) |
| Descripción: | Accede a la base de datos y devuelve un listado de los usuarios del sistema. |

| **Nombre: C_Configuracion** | |
|---|---|
| **Tipo de clase: Controladora** | |
| **Para cada responsabilidad**: | |
| Nombre: | Levanta_GestionarNomencladores() |
| Descripción: | Se encarga de cargar la interfaz de gestión de los nomencladores. |
| | |
| Nombre: | Configurar_Nomencladores() |
| Descripción: | Se encarga de establecer una variable de sesión a la tabla del nomenclador y a la vista en la que se gestionan los datos del mismo. |
| | |
| Nombre: | GestionarNomenclador() |
| Descripción: | Se utiliza para listar los datos del nomenclador establecido en: Configurar_Nomencladores(), y acceder a las funcionalidades de gestión: Modificar, Eliminar, Adicionar. |
| | |
| Nombre: | Insertar_Descripcion() |
| Descripción: | Se encarga de Insertar un nuevo nomenclador en la tabla definida por la responsabilidad GestionarNomenclador(). |
| | |
| Nombre: | Eliminar_Nomenclador() |
| Descripción: | Se encarga de Eliminar un nomenclador en la tabla definida por la responsabilidad GestionarNomenclador(). |
| | |
| Nombre: | Llamar_Modificar() |
| Descripción: | Se encarga de cargar la interfaz en la que se modifican los datos del nomenclador. |
| | |
| Nombre: | Modificar_Nomenclador() |
| Descripción: | Se encarga de Modificar los Datos de un nomenclador en la tabla definida por la responsabilidad GestionarNomenclador(). |
| | |
| Nombre: | ModificarNomencladorClave() |

| | |
|---|---|
| Descripción: | Se encarga de modificar los datos del nomenclador Claves ya que este nomenclador tiene elementos particulares no comunes en los otros nomencladores. |

| **Nombre: M_Configuracion** | |
|---|---|
| **Tipo de clase: Modelo** | |
| **Para cada responsabilidad**: | |
| Nombre: | Model_RetornaMunicipios($id) |
| Descripción: | Accede a la base de datos y retorna los municipios de una provincia determinada. |
| | |
| Nombre: | Model_RetornaProvincias() |
| Descripción: | Accede a la base de datos y retorna las provincias. |
| | |
| Nombre: | Model_RetornaNomenclador($tabla) |
| Descripción: | Accede a la base de datos y retorna los datos almacenados en la tabla de un nomenclador determinado. |
| | |
| Nombre: | Model_Query_Listar_Nomenclador($tabla,$per_page,$offset) |
| Descripción: | Accede a la base de datos y retorna los datos almacenados en la tabla de un nomenclador determinado, teniendo en cuenta los valores de la variables: $per_page, $offset. El $per_page indica cuantos elemento debe retornar y el $offset a partir de que elemento debe retornar. |
| | |
| Nombre: | Model_Insertar_Descripcion($tabla,$descripcion) |
| Descripción: | Accede a la base de datos e inserta un nomenclador en la tabla especificada. |
| | |
| Nombre: | Model_Eliminar_Nomenclador($id,$tabla) |
| Descripción: | Accede a la base de datos y elimina un nomenclador determinado de la tabla especificada. |
| | |

| | |
|---|---|
| Nombre: | Model_Modificar_Nomenclador($dato,$id,$tabla) |
| Descripción: | Accede a la base de datos y modifica la descripción de un nomenclador dado el identificador del mismo y lo sustituye por el valor contenido en el parámetro $dato. |
| | |
| Nombre: | Model_Retornnald_CH() |
| Descripción: | Se encarga de acceder a la base de datos y retornar el id de la provincia Ciudad de La Habana específicamente. |

| **Nombre: C_Paciente_TI** | |
|---|---|
| **Tipo de clase: Controladora** | |
| **Para cada responsabilidad**: | |
| Nombre: | LevantarTrasladoPacienteaProvincia() |
| Descripción: | Se encarga de cargar la interfaz que recoge los datos a insertar en una demanda de Paciente para traslado a provincia. |
| | |
| Nombre: | Levantar_Buscar_Dem_por_Provdest() |
| Descripción: | Se encarga de cargar la interfaz donde se establecen los parámetros de búsqueda para gestionar una demanda del tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Controller_Insertar_Paciente_TI() |
| Descripción: | Se encarga de insertar una demanda del tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Lista() |
| Descripción: | Establece la Variable sesión que se emplea como parámetro de búsqueda a la hora de mostrar los pacientes pendientes para trasladarse a una provincia determinada. |
| | |
| Nombre: | Mostrar_Pacientes_Pendientes() |
| Descripción: | Lista los pacientes pendientes con solicitud de traslado a una provincia determinada. |

| | |
|---|---|
| Nombre: | Eliminar_Paciente_a_Traslado_Interprovincial() |
| Descripción: | Elimina una demanda del tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Cargar_Datos_Demanda_Paciente_TI() |
| Descripción: | Se encarga de cargar la interfaz que contiene todos los datos de la demanda de tipo: Paciente para traslado a provincia, para que estos sean modificado en caso de que el usuario lo estime conveniente. |
| | |
| Nombre: | Modificar_Paciente_TI() |
| Descripción: | Se encarga de modificar los datos de la demanda de tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Cargar_Cumplimiento() |
| Descripción: | Se encarga de cargar la interfaz que contiene todos los datos para darle cumplimiento a una demanda de tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Insertar_Cumplimiento() |
| Descripción: | Se encarga de insertar el cumplimiento de una demanda de tipo: Paciente para traslado a provincia. Después de la ejecución de este método ya la demanda no sería valorada como pendiente. |
| | |
| Nombre: | Mostrar_Datos_Demanda_Paciente_TI() |
| Descripción: | Se encarga de cargar la interfaz que visualiza los datos de una demanda de tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | BuscarPersonalRegistro($registro) |
| Descripción: | Se encarga de buscar los Datos de un profesional de la Salud en el Registro de Personal de la Salud (RPS), perteneciente al SISalud. |

| |
|---|
| **Nombre: M_Paciente_TI** |
| **Tipo de clase: Modelo** |
| **Para cada responsabilidad**: |

| | |
|---|---|
| Nombre: | Model_Insertar_Paciente_TI($demanda_paciente_alta,$paciente ) |
| Descripción: | Accede a la base de datos e inserta los datos de los objetos pasados por parámetros que corresponden a una demanda de tipo: Paciente para traslado a provincia, en sus respectivas tablas. |
| | |
| Nombre: | Model_Eliminar_Paciente_TI($id) |
| Descripción: | Se encarga de acceder a la base de datos y eliminar toda la información correspondiente a una demanda de tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Model_Cargar_Datos_Demanda($id) |
| Descripción: | Se encarga de acceder a la base de datos y retornar todos los datos de una demanda de tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Modificar_Paciente_TI($demanda_paciente_alta,$paciente) |
| Descripción: | Se encarga de acceder a la base de datos y modificar los datos de una demanda de tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Model_Insertar_Cumplimiento($id,$fecha,$chapa,$procedencia_del_movil) |
| Descripción: | Se encarga de acceder a la base de datos e insertar los datos de un cumplimiento de una demanda del tipo: Paciente para traslado a provincia. |
| | |
| Nombre: | Model_QueryTot_MostrarPacientesPendientes($prov_dest,$per_page,$offset) |
| Descripción: | Se encarga de acceder a la base de datos y listar todas las demandas de tipo: Paciente para traslado a provincia, que están pendientes en espera de su cumplimiento. |

| **Nombre: ServiciosWeb.** | |
|---|---|
| **Tipo de clase: Modelo** | |
| **Para cada responsabilidad**: | |
| Nombre: | Model_BuscarPersonaSisSalud($CriteriosBusqueda,$RC) |
| Descripción: | Se encarga de obtener todos los ciudadanos que cumplan con los parámetros de búsqueda especificados desde el servicio Web que brinda el componente Registro de Ciudadanos (RC). |

| | |
|---|---|
| Nombre: | Model_BuscarPersonalRegistro($registro,$cliente) |
| Descripción: | Se encarga de buscar un profesional de la Salud dado su número de registro profesional en el Componente Registro Profesional de la Salud (RPS). |
| | |
| Nombre: | Model_DetallesPersona($id,$RC) |
| Descripción: | Se encarga de retornar los datos de una persona en específico desde el servicio Web que brinda el componente Registro de Ciudadanos (RC). |

| **Nombre: C_Traslados_Interprovinciales** | |
|---|---|
| **Tipo de clase: Controladora** | |
| **Para cada responsabilidad**: | |
| Nombre: | Activar_GestionarPaciente() |
| Descripción: | Se utiliza para verificar si un usuario con el Rol Coordinador Provincial puede gestionar los pacientes de una planificación de traslado interprovincial. |
| | |
| Nombre: | Activar_ModificarDemandaPlanificada() |
| Descripción: | Se utiliza para verificar si un usuario con el Rol Coordinador Provincial puede modificar los datos generales de la planificación de un de traslado interprovincial. |
| | |
| Nombre: | Adicionar_Paciente() |
| Descripción: | Se utiliza para adicionar un paciente a una planificación de traslado interprovincial. |
| | |
| Nombre: | Adicionar_pac_entrante() |
| Descripción: | Se utiliza para adicionar un paciente dentro de la gestión de pacientes entrantes de un traslado interprovincial. |
| | |
| Nombre: | Cargar_Datos_EPM_de_las_Provincias() |
| Descripción: | Se encarga de visualizar los datos de la interfaz de usuario encargada de insertar una planificación de traslado interprovincial. |
| | |

| | |
|---|---|
| Nombre: | Cargar_Datos_Generales_de_Demanda_Planificada($id) |
| Descripción: | Se encarga de visualizar los datos de la interfaz de usuario encargada de modificar los datos de un traslado interprovincial. |
| | |
| Nombre: | Cargar_Datos_Paciente() |
| Descripción: | Se encarga de visualizar los datos de un paciente de una planificación de traslado interprovincial para ser modificados. |
| | |
| Nombre: | Cargar_Datos_Permiso() |
| Descripción: | Se encarga de visualizar los datos de una planificación de traslado interprovincial en la interfaz de usuario en la que se autoriza o deniega dicho traslado. |
| | |
| Nombre: | Cargar_Datos_pac_entrante() |
| Descripción: | Se encarga de visualizar los datos de un paciente en la interfaz de encargada de modificar los datos de un paciente que realizara su entrada a Ciudad de La Habana. |
| | |
| Nombre: | Cargar_Pacientes_Entrada($id) |
| Descripción: | Se encarga de visualizar los datos de los pacientes de la entrada de un traslado interprovincial a Ciudad de La Habana. |
| | |
| Nombre: | Controller_Insertar_Traslado_Interprovincial() |
| Descripción: | Se encarga de insertar un nuevo traslado interprovincial. |
| | |
| Nombre: | Efectuar_Entrada() |
| Descripción: | Se utiliza para insertar los datos de la entrada de una demanda de traslado interprovincial a Ciudad de La Habana. |
| | |
| Nombre: | EsatableceSessionEntrada() |
| Descripción: | Se encarga de guardar en variables de sesión los parámetros con los que se realiza el listado de las entradas de demandas de traslados interprovinciales. |
| | |
| Nombre: | EstableceSession_Gest_Pac_Entrada() |

| | |
|---|---|
| Descripción: | Se encarga de guardar en variables de sesión los parámetros con los que se realiza el listado de los pacientes de determinada demanda de traslado interprovincial que hace entrada en Ciudad de La Habana. |
| | |
| Nombre: | Insertar_Salida() |
| Descripción: | Se encarga de insertar los datos de la salida de retorno del móvil que ha cumplido con una demanda de traslado interprovincial. |
| | |
| Nombre: | LLamar_Buscar_Demanda_Planificada() |
| Descripción: | Se utiliza para visualizar la interfaz de usuario que se encarga de escoger los parámetros de búsqueda para listar las demandas de traslados interprovinciales planificadas. |
| | |
| Nombre: | LLamar_ProvinciasDemandantes() |
| Descripción: | Se utiliza para visualizar la interfaz de usuario de las provincias que tienen planificadas demandas de traslados interprovinciales en espera de ser autorizadas. |
| | |
| Nombre: | LLamar_buscar_Entradas() |
| Descripción: | Se utiliza para visualizar la interfaz de usuario que se encarga de buscar las demandas de traslados interprovinciales que han realizado su entrada para gestionar su retorno a la provincia de origen. |
| | |
| Nombre: | Listar_Entradas() |
| Descripción: | Se utiliza para visualizar la interfaz de usuario que se encarga de listar las demandas de traslados interprovinciales que han realizado su entrada para gestionar su retorno a la provincia de origen. |
| | |
| Nombre: | ListarSinPermiso() |
| Descripción: | Se utiliza para visualizar la interfaz de usuario que se encarga de listar las demandas de traslados interprovinciales que no han sido autorizadas para gestionar su autorización o denegación. |
| | |

| | |
|---|---|
| Nombre: | Listar_Pacientes_Demanda_Planificada($id_dem) |
| Descripción: | Se encarga de guardar en variables de sesión los parámetros con los que se realiza el listado de los pacientes de una demanda de traslado interprovincial planificada. |
| | |
| Nombre: | Listar_Para_Entrada() |
| Descripción: | Se utiliza para listar las demandas de traslado interprovincial planificadas para efectuar su entrada. |
| | |
| Nombre: | Listar_Planificacion_TI() |
| Descripción: | Se encarga de guardar en variables de sesión los parámetros con los que se realiza el listado de las demandas de traslados interprovinciales planificados. |
| | |
| Nombre: | Llamar_buscar_para_Entradas() |
| Descripción: | Se utiliza para buscar las demandas de traslados interprovinciales que están pendientes de realizar su entrada. |
| | |
| Nombre: | Modificar_Datos_Generales_de_Demanda_Planificada() |
| Descripción: | Se utiliza para gestionar la modificación de los datos generales de una demanda de traslado interprovincial planificada. |
| | |
| Nombre: | Modificar_Paciente_de_Demanda_Planificada() |
| Descripción: | Se utiliza para modificar los datos de un paciente determinado de una demanda de traslado interprovincial. |
| | |
| Nombre: | Modificar_Paciente_entrante() |
| Descripción: | Se utiliza para modificar los datos de un paciente determinado de la entrada de una demanda de traslado interprovincial. |
| | |
| Nombre: | Mostrar_Pacientes_Demanda_Planificada() |
| Descripción: | Se utiliza para listar los pacientes de una demanda de traslado interprovincial planificada para gestionar los mismos. |
| | |
| Nombre: | Mostrar_Pacientes_Entrada() |

| | |
|---|---|
| Descripción: | Se utiliza para listar los pacientes de una demanda de traslado interprovincial que efectúa su entrada. |
| | |
| Nombre: | Mostrar_Planificacion_TI() |
| Descripción: | Se encarga de listar la planificación de las demandas de traslados interprovinciales para gestionar las mismas. |
| | |
| Nombre: | Permiso() |
| Descripción: | Se utiliza para insertar los datos del permiso de una planificación de traslado interprovincial determinada. |
| | |
| Nombre: | ProvinciasDemandantes() |
| Descripción: | Se utiliza para listar las provincias que han planificado alguna demanda de traslado interprovincial y que se encuentran en espera para ser autorizadas. |
| | |
| Nombre: | SessionListarSinPermiso() |
| Descripción: | Se encarga de guardar en variables de sesión los parámetros con los que se realiza el listado de las demandas de traslados interprovinciales que no han sido autorizadas. |
| | |
| Nombre: | XMLInstituciones() |
| Descripción: | Se encarga de crear un XML con los datos de las instituciones de salud del Registro de Unidades de Salud (RUS) dado el id de un municipio. |

| **Nombre: M_Traslados_Interprovinciales** | |
|---|---|
| **Tipo de clase: Modelo** | |
| **Para cada responsabilidad**: | |
| Nombre: | Listar_Tablas($tabla) |
| Descripción: | Se encarga de acceder a la base de datos y listar los elementos de una tabla determinada. |
| | |
| Nombre: | Model_Adicionar_Paciente($Paciente,$Paciente_demanda) |
| Descripción: | Se encarga de acceder a la base de datos e insertar un paciente de una |

| | |
|---|---|
| | demanda de traslado interprovincial planificada. |
| | |
| Nombre: | Model_Adicionar_pac_entrante($Paciente,$Paciente_entrante) |
| Descripción: | Se encarga de acceder a la base de datos e insertar un paciente de la entrada de una demanda de traslado interprovincial. |
| | |
| Nombre: | Model_Cant_Pac_Retornan($id) |
| Descripción: | Se encarga de acceder a la base de datos y devolver la cantidad de pacientes que retornan de una entrada de demanda de traslado interprovincial. |
| | |
| Nombre: | Model_Cantidad_Pacientes_Demanda($id,$tabla,$id_tabla) |
| Descripción: | Se encarga de acceder a la base de datos y devolver la cantidad de pacientes que tiene determinada planificación de demanda de traslado interprovincial. |
| | |
| Nombre: | Model_Cargar_Datos_Generales_de_Demanda_Planificada($id) |
| Descripción: | Se encarga de acceder a la base de datos y retornar los datos generales de una demanda de traslado interprovincial planificada. |
| | |
| Nombre: | Model_Cargar_Datos_Paciente($id) |
| Descripción: | Se encarga de acceder a la base de datos y retornar los datos generales de un paciente determinado. |
| | |
| Nombre: | Model_Cargar_Datos_Paciente_demanda($id_paciente,$id_demanda) |
| Descripción: | Se encarga de acceder a la base de datos y retornar los datos específicos del paciente de una determinada demanda de traslado interprovincial. |
| | |
| Nombre: | Model_Modificar_Paciente_entrante($Paciente, $Paciente_entrante) |
| Descripción: | Se encarga de acceder a la base de datos y modificar los datos de un paciente determinado de una entrada de demanda de traslado interprovincial planificada. |
| | |
| Nombre: | Model_MotivosDenegacion($id) |
| Descripción: | Se encarga de acceder a la base de datos y retornar las causas de la denegación de una determinada planificación de traslado interprovincial. |
| | |

| | |
|---|---|
| Nombre: | Model_Permiso($Datos) |
| Descripción: | Se encarga de acceder a la base de datos e insertar los datos de la autorización o denegación del autorizo de una planificación de traslado interprovincial. |
| | |
| Nombre: | Model_ProvinciasEspera() |
| Descripción: | Se encarga de acceder a la base de datos y retornar las provincias que tienen demandas de traslados interprovinciales en espera de ser autorizadas. |
| | |
| Nombre: | Model_QueryTot_Pac_Entrada($id_entr) |
| Descripción: | Se encarga de acceder a la base de datos y retornar la lista de pacientes que pertenecen a determinada entrada. |
| | |
| Nombre: | Model_Query_Tot_ListarEntrada($po,$per_page,$offset) |
| Descripción: | Se encarga de acceder a la base de datos y devolver una lista de todas las demandas de traslados interprovinciales que han realizado su entrada a Ciudad de La Habana. |
| | |
| Nombre: | Model_Query_Tot_ListarPlanificacion($po,$pd,$fi,$ff,$per_page,$offset) |
| Descripción: | Se encarga de acceder a la base de datos y devolver una lista de todas las demandas de traslados interprovinciales planificadas que no han efectuado su entrada. |
| | |
| Nombre: | Model_Query_Tot_PacDemandaPlanificada($id_dem) |
| Descripción: | Se encarga de acceder a la base de datos y devolver los pacientes de una determinada demanda de traslado interprovincial planificada. |
| | |
| Nombre: | Model_Query_Tot_SinPermiso($prov,$per_page,$offset) |
| Descripción: | Se encarga de acceder a la base de datos y devolver una lista de todas las demandas de traslados interprovinciales que se encuentran en espera de ser autorizadas. |
| | |
| Nombre: | Model_Query_Tot_para_Entrada($po ,$f_inicio,$f_fin,$ch,$per_page,$offset) |

| | |
|---|---|
| Descripción: | Se encarga de acceder a la base de datos y devolver una lista de todas las demandas de traslados interprovinciales que están autorizadas y se encuentran en espera de efectuar la entrada a Ciudad de La Habana. |
| | |
| Nombre: | Model_Retorna_ID_Prov($id) |
| Descripción: | Se encarga de acceder a la base de datos y devolver el id específico de Ciudad de La Habana. |
| | |
| Nombre: | Modificar_Datos_Generales_de_Demanda_Planificada($demanda_pv_p) |
| Descripción: | Se encarga de acceder a la base de datos y modificar los datos generales de una planificación de traslado interprovincial. |

| **Nombre: C_Demanda_Local** | |
|---|---|
| **Tipo de clase: Controladora** | |
| **Para cada responsabilidad**: | |
| Nombre: | LlamarDemandaLocal() |
| Descripción: | Se utiliza para visualizar la interfaz de usuario mediante la cual se introducirán los datos de una Demanda Local. |
| | |
| Nombre: | InsertarDemandaLocal() |
| Descripción: | Se utiliza para insertar los datos de una Demanda Local. |
| | |
| Nombre: | ListarEmergenciasTransplantes() |
| Descripción: | Se utiliza para mostrar todas las Demandas Locales que son del tipo Emergencias o Trasplantes. |
| | |
| Nombre: | ListarAltaHospitalaria() |
| Descripción: | Se utiliza para mostrar todas las Demandas Locales que son del tipo Alta Hospitalaria. |
| | |
| Nombre: | LevantaBuscarTurnoMedico() |
| Descripción: | Se utiliza para visualizar la interfaz de usuario mediante la cual se introducirá una fecha y se mostrarán los turnos médicos planificados para esa fecha. |

| | |
|---|---|
| Nombre: | ListarTurnoMedicoTestDiagnóstico() |
| Descripción: | Se utiliza para mostrar todas las Demandas Locales que son del tipo Turno Médico o Test Diagnóstico. |
| | |
| Nombre: | Seleccionar($id) |
| Descripción: | Se utiliza para seleccionar las vistas que son utilizadas para asignar un móvil y darle el seguimiento a una Demanda Local. |
| | |
| Nombre: | MostrarModificarDemanda($id) |
| Descripción: | Se utiliza para levantar la vista encargada de asignar el móvil a una Demanda Local. |
| | |
| Nombre: | MostrarCambiaClave($id) |
| Descripción: | Se utiliza para levantar la vista encargada de gestionar las claves por las que va transitando una Demanda Local. |
| | |
| Nombre: | C_Asignar_Movil() |
| Descripción: | Se utiliza para insertar los datos del móvil asignado a la Demanda Local. |
| | |
| Nombre: | C_Modificar_Demanda_Clave() |
| Descripción: | Se utiliza para asignar las claves a una Demanda Local. |

| **Nombre: M_Demanda_Local** | |
|---|---|
| **Tipo de clase: Modelo** | |
| **Para cada responsabilidad**: | |
| Nombre: | ModelInsertarDemandaLocal($demanda_local,$paciente) |
| Descripción: | Se encarga de acceder a la base de datos e insertar los datos de una Demanda Local en sus respectivas tablas. |
| | |
| Nombre: | M_Listar_Emergencia_Transplante($per_page,$offset) |
| Descripción: | Se encarga de acceder a la base de datos y retornar un listado de las Demandas Locales de tipo Emergencias o Trasplantes. |
| | |

| | |
|---|---|
| Nombre: | M_Listar_Alta_Hospitalaria($per_page,$offset) |
| Descripción: | Se encarga de acceder a la base de datos y retornar un listado de las Demandas Locales de tipo Alta Hospitalaria. |
| | |
| Nombre: | M_Listar_Turnos_Medicos_Test_Diagnóstico($fecha,$per_page,$offset) |
| Descripción: | Se encarga de acceder a la base de datos y retornar un listado de las Demandas Locales de tipo Turnos Médicos y Test Diagnóstico. |
| | |
| Nombre: | Model_Estado($id) |
| Descripción: | Se encarga de acceder a la base de datos y retornar en que estado se encuentra la Demanda Local (En Espera, Iniciada, Terminada). |
| | |
| Nombre: | DatosDemanda($id_demanda) |
| Descripción: | Se encarga de acceder a la base de datos y retornar los datos de una Demanda Local en específico. |
| | |
| Nombre: | DatosClave($id_demanda) |
| Descripción: | Se encarga de acceder a la base de datos y retornar el listado de las claves por las que ha transitado una Demanda Local. |
| | |
| Nombre: | M_Asignar_Movil($id_demanda,$id_ambulancia) |
| Descripción: | Se encarga de acceder a la base de datos e insertar los datos del móvil que le es asignado a determinada Demanda Local. |
| | |
| Nombre: | ModificarDemandaClave($demanda,$clave,$hora,$causa,$observacion) |
| Descripción: | Se encarga de acceder a la base de datos e insertar las claves por las que transita una Demanda Local. |

## 3.3 Diseño de la Base de Datos.

Una base de datos es una serie de datos organizados y relacionados entre si, los cuales pueden ser recolectados y explotados por sistemas de gestión de información. Con el objetivo de lograr la persistencia de los datos y que estos puedan ser utilizados en cualquier momento por el usuario del sistema encargado de gestionar la información del módulo CCEMN se utiliza una base de datos

relacional. El uso de esta técnica de almacenamiento está muy difundido actualmente debido a las garantías que ofrece en cuanto a la durabilidad de los datos se refiere.

### 3.3.1 Modelo de Datos.

El modelo de los datos es el encargado de describir de una forma abstracta la representación lógica y física de los datos persistentes en el sistema. Básicamente consiste en una colección de conceptos que se emplean para describir la estructura de la base de datos y que está constituida por entidades, atributos y relaciones.

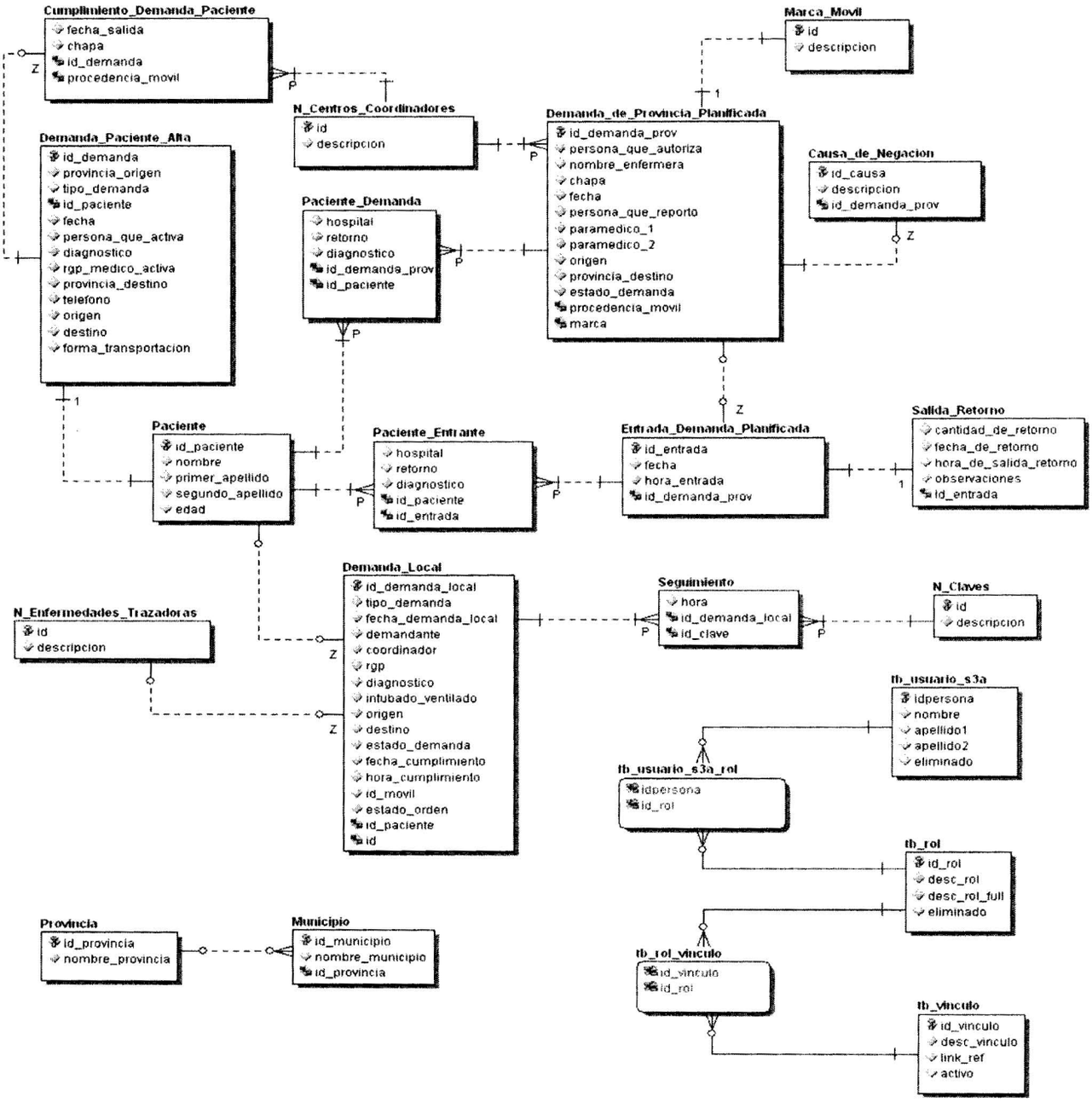


**Figura 20: Modelo de Datos.**

### 3.3.2 Descripción de las Tablas.

| **Nombre: Demanda_paciente_alta** | | |
|---|---|---|
| **Descripción:** Recoge los datos de la demanda de un paciente para traslado interprovincial. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_demanda | Integer | Identificador para los datos de esa tabla. |
| Provincia_origen | Integer | Identificador de la provincia de origen. |
| Tipo_demanda | Varchar | Especifica si es demanda domiciliar o de hospital. |
| Id_paciente | Integer | Identificador de la tabla Paciente. |
| Fecha | Date | Fecha en la que se efectúa la demanda. |
| Persona_que_activa | Varchar | Nombre del médico que hace la demanda. |
| Diagnóstico | Varchar | Responde al diagnóstico que es emitido por el médico. |
| Rgp_medico_activa | Integer | Responde al registro profesional del Médico que hace la demanda. |
| Provincia_destino | Integer | Identificador de la provincia de destino. |
| Teléfono | Integer | El teléfono del lugar donde se encuentra el paciente. |
| Origen | Varchar | Responde a la dirección de origen. |
| Destino | Varchar | Responde a la dirección de destino. |
| Forma_transportacion | Varchar | Especifica si la forma de transportación es camillado o sentado. |

| **Nombre: Cumplimiento_demanda_paciente** | | |
|---|---|---|
| **Descripción:** Recoge los datos del cumplimiento de la demanda de un paciente para traslado interprovincial. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Fecha_salida | Date | Fecha en la que se le da cumplimiento a la demanda. |
| Chapa | Varchar | Chapa del móvil que le da cumplimiento a la demanda. |
| Id_demanda | Integer | Identificador de la tabla demanda_paciente_alta. |
| Procedencia_movil | Integer | Identificador de la tabla del nomenclador n_centros_coordinadores. |

| **Nombre: Demanda_de_provincia_planificada** | | |
|---|---|---|
| **Descripción:** Recoge los datos de la planificación de traslados Interprovinciales. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_demanda_prov | Integer | Identificador para los datos de esta tabla. |
| Persona_que_autoriza | Integer | Identificador del usuario del CCEMN que autoriza o deniega. |
| Nombre_enfermera | Varchar | Nombre de la Enfermera que viene en el traslado. |
| Chapa | Varchar | Chapa del móvil que realizará el traslado. |
| Fecha | Date | Fecha en la que se hace la planificación. |
| Persona_que_reporto | integer | Identificador del usuario del CCEMP que hace la planificación. |
| Paramedico_1 | Varchar | Nombre del 1er paramédico que viene en el traslado. |
| Paramedico_2 | Varchar | Nombre del 2do paramédico que viene en el traslado. |
| Origen | Integer | Identificador de la provincia origen. |
| Provincia_destino | Integer | Identificador de la provincia destino. |
| Procedencia_movil | Integer | Identificador de la tabla del nomenclador n_centros_coordinadores. |
| Marca | Integer | Identificador de la tabla del nomenclador marca_movil. |
| Estado_demanda | | Indica si la demanda fue denegada, autorizada o está aún en espera. |

| **Nombre: Paciente** | | |
|---|---|---|
| **Descripción:** Recoge los datos generales del paciente que se informan cuando se hace una demanda. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_paciente | Integer | Identificador para los datos de esa tabla. |
| Nombre | Varchar | Responde al nombre. |
| Primer_apellido | Varchar | Responde al primer apellido. |
| Segundo_apellido | Varchar | Responde al segundo apellido. |
| Edad | Integer | Edad del Paciente. |

| **Nombre: Paciente_demanda** | | |
|---|---|---|
| **Descripción:** Recoge los datos específicos de la relación entre la tabla Paciente y Demanda_de_provincia_planificada. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Hospital | Varchar | Institución de salud a la que se traslada el paciente. |
| Retorno | Varchar | Especifica si el paciente retorna o ingresa. |
| Diagnóstico | Varchar | Recoge el diagnóstico del paciente. |
| Id_paciente | Integer | Identificador de la tabla Paciente. |
| Id_demanda_prov | Integer | Identificador de la tabla Demanda_de_provincia_planificada. |

| **Nombre: Paciente_Entrante** | | |
|---|---|---|
| **Descripción:** Recoge los datos específicos de la relación entre la tabla Paciente y Entrada_Demanda_Planificada. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Hospital | Varchar | Institución de salud a la que se traslada el paciente en Ciudad de la Habana. |
| Retorno | Varchar | Especifica si el paciente retorna o ingresa. |
| Diagnóstico | Varchar | Recoge el diagnóstico del paciente. |
| Id_paciente | Integer | Identificador de la tabla Paciente. |
| Id_entrada | Integer | Identificador de la tabla Entrada_Demanda_Planificada. |

| **Nombre: Entrada_Demanda_Planificada** | | |
|---|---|---|
| **Descripción:** Recoge los datos de la entrada de una demanda antes planificada. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_entrada | Integer | Identificador para los datos de esa tabla. |
| Fecha | Date | Fecha en que se realiza la entrada a Ciudad de la Habana. |
| Hora_entrada | Time | Hora en que se realiza la entrada a Ciudad de la Habana. |
| Id_demanda_prov | Integer | Identificador de la tabla Demanda_de_provincia_planificada. |

| **Nombre: Salida_Retorno** | | |
|---|---|---|
| **Descripción:** Recoge los datos de la salida de un móvil hacia su provincia de origen. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Cantidad_de_retorno | Integer | Cantidad de pacientes que retornan en el móvil. |
| Fecha_de_retorno | Date | Fecha en que retorna el móvil. |
| Hora_de_salida_retorno | Time | Hora en que retorna el móvil. |
| Observaciones | Varchar | Observaciones del retorno. |
| Id_entrada | Integer | Identificador de la tabla Entrada_Demanda_Planificada. |

| **Nombre: Provincia** | | |
|---|---|---|
| **Descripción:** Recoge el identificador y el nombre de las provincias del Registro de Ubicación. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_provincia | Integer | Identificador para los datos de esa tabla. |
| Nombre_provincia | Varchar | Responde al nombre de la provincia. |

| **Nombre: Municipio** | | |
|---|---|---|
| **Descripción:** Recoge el identificador y el nombre de los municipios por provincia del Registro de Ubicación. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_municipio | Integer | Identificador para los datos de esa tabla. |
| Id_provincia | Integer | Identificador de la tabla provincia. |
| Nombre_municipio | Varchar | Responde al nombre del municipio. |

| **Nombre:N_enfermedades_trazadoras** | | |
|---|---|---|
| **Descripción:** Recoge los datos de las enfermedades definidas por el Cliente. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Id | Integer | Identificador para los datos de esa tabla. |
| Descripcion | Varchar | Responde al nombre del nomenclador. |
| Eliminado | Integer | Indica si el nomenclador es eliminado o no. |

| **Nombre: Demanda_local** | | |
|---|---|---|
| **Descripción:** Recoge los datos de las demandas dentro de Ciudad de la Habana. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_demanda_local | Integer | Identificador para los datos de esa tabla. |
| Tipo_demanda | Integer | Especifica el tipo de demanda haciendo uso de un valor numérico:<br>1-Emergencia.<br>2-Trasplante.<br>3-Alta Hospitalaria.<br>4-Turno Médico.<br>5-Test Diagnóstico. |
| Fecha_demanda_local | Date | Recoge la fecha en que se activa la demanda. |
| Demandante | Varchar | Recoge el nombre de la persona que efectúa la demanda. |
| Coordinador | Integer | Identificador de la tabla tb_ususari_s3a. |
| RGP | Integer | Recoge el número del registro profesional en caso de que la persona que efectúe la demanda sea un profesional de la salud. |
| Diagnóstico | Integer | Recoge el diagnóstico emitido por el demandante. |
| Intubado_ventilado | Integer | Especifica si el paciente requiere ser intubado/ventilado a través de un valor numérico: 0 ó1 |
| Origen | Varchar | Responde a la dirección de origen. |
| Destino | Varchar | Responde a la dirección de destino. |
| Estado_demanda | Varchar | Indica si la demanda fue Iniciada, se encuentra en ejecución o fue terminada. |
| fecha_cumplimiento | Varchar | Recoge la fecha en que se le debe dar cumplimiento a una demanda. |
| hora_cumplimiento | Time | Recoge la hora en la que se le debe dar cumplimiento a una demanda. |
| id_paciente | Integer | Identificador de la tabla Paciente. |
| id_enfermedades_trazadoras | Integer | Identificador de la tabla N_enfermedades_trazadoras. |

| | | |
|---|---|---|
| estado_orden | Varchar | Recoge si el caso fue fallido, sin efecto o si es Tarjeta Negra. |
| Id_movil | Varchar | Recoge la chapa del móvil que efectúa el traslado. |

| **Nombre: N_centros_coordinadores** | | |
|---|---|---|
| **Descripción:** Recoge los datos de los diferentes centros coordinadores del país. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Id | Integer | Identificador para los datos de esa tabla. |
| Descripcion | Varchar | Responde al nombre del nomenclador. |
| Eliminado | Integer | Indica si el nomenclador es eliminado o no. |

| **Nombre:N_claves** | | |
|---|---|---|
| **Descripción:** Recoge los valores de las claves que se le asignan a una demanda. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Id | Integer | Identificador para los datos de esa tabla. |
| Descripcion | Varchar | Describe lo que significa que a una demanda le sea asignada dicha clave. |
| Eliminado | Integer | Indica si el nomenclador es eliminado o no. |

| **Nombre: N_marca_movil** | | |
|---|---|---|
| **Descripción:** Recoge las marcas de los móviles con que cuenta el SIUM. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Id | Integer | Identificador para los datos de esa tabla. |
| Descripcion | Varchar | Responde al nombre del nomenclador. |
| Eliminado | Integer | Indica si el nomenclador es eliminado o no. |

| **Nombre: tb_usuario_s3a** | | |
|---|---|---|
| **Descripción:** Recoge los datos de los usuarios del sistema. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_persona | Integer | Identificador para los datos de esa tabla cargado del Registro Ciudadano (RC). |

| | | |
|---|---|---|
| Nombre | Varchar | Recoge el nombre del ciudadano. |
| Apellido1 | Varchar | Recoge el primer apellido del ciudadano. |
| Apellido2 | Varchar | Recoge el segundo apellido del ciudadano. |
| Eliminado | Integer | Indica si el usuario es eliminado o no. |

| **Nombre: tb_rol** | | |
|---|---|---|
| **Descripción:** Recoge los roles existentes en el sistema. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_rol | Integer | Identificador para los datos de esa tabla. |
| Descripcion | Varchar | Recoge una descripción del rol. |
| Eliminado | Tinyinteger | Indica si el estado del rol es eliminado o no. |

| **Nombre: tb_usuario_s3a_rol** | | |
|---|---|---|
| **Descripción:** Establece una relación entre los usuarios y el rol que desempeñan. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_persona | Integer | Identificador de la tabla tb_usuario_s3a. |
| id_rol | Integer | Identificador de la tabla tb_rol. |

| **Nombre: tb_rol_vinculo** | | |
|---|---|---|
| **Descripción:** Establece una relación entre el rol y los vínculos a los que está autorizado a acceder. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_rol | Integer | Identificador de la tabla tb_rol. |
| Id_vinculo | Integer | Identificador de la tabla tb_vinculo. |

| **Nombre: tb_vinculo** | | |
|---|---|---|
| **Descripción:** Recoge las URL que invocan a las funcionalidades del sistema. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_vinculo | Integer | Identificador para los datos de esa tabla. |
| Desc_vinculo | Varchar | Recoge una descripción simplificada del vínculo. |

| | | |
|---|---|---|
| Link_ref | Varchar | Recoge las URL para invocar las funcionalidades del sistema. |
| Activo | Tinyinteger | Establece el estado que indica si una funcionalidad puede ser invocada desde el menú. |

| **Nombre: causa_de_negacion** | | |
|---|---|---|
| **Descripción:** Recoge el motivo de negación de una planificación de traslado interprovincial. | | |
| **Atributo** | **Tipo** | **Descripción** |
| id_causa | Integer | Identificador para los datos de esa tabla. |
| Descripcion | Varchar | Recoge el por qué fue denegada la planificación. |
| Id_demanda_prov | Integer | Identificador de la tabla Demanda_de_provincia_planificada. |

| **Nombre: Seguimiento** | | |
|---|---|---|
| **Descripción:** Recoge la hora y las diferentes claves que va tomando una demanda. | | |
| **Atributo** | **Tipo** | **Descripción** |
| Hora | Time | Recoge la hora. |
| Id_demanda_local | Integer | Identificador de la tabla Demanda_local. |
| Id_clave | Integer | Identificador de la tabla del nomenclador N_claves. |

## Conclusiones.

En el presente capítulo se ha realizado una amplia descripción de los componentes a los que se integra el módulo y la forma en que se establece esta integración. Además se ha realizado, para una mejor comprensión de la aplicación, una breve descripción de las funcionalidades que están presentes en las diferentes clases del sistema y tablas de la base de datos. También se ha mostrado la base de datos de forma gráfica.

## CONCLUSIONES

Con la implementación del módulo "Centro Coordinador de Emergencia Médica Nacional" del Sistema Informatizado para el Centro Nacional de Urgencias Médicas (SIUM) se logra mejorar la gestión de la información que tiene lugar en el CCEMN.

Con el diseño de la base de datos a utilizar se logra preservar a largo plazo la información referente al CCEMN, lo cual posibilitará llevar a cabo un control estadístico de disímiles aspectos importantes para futuras investigaciones que se desee realizar por parte de los especialistas e investigadores de la salud.

La sustitución de la gestión manual por el nuevo sistema repercutirá en la disminución de los gastos e insumos propios de la institución.

## RECOMENDACIONES

Tomando como base las experiencias obtenidas durante el desarrollo del presente trabajo de diploma se proponen, las siguientes recomendaciones:

- ✓ Elaborar un Diseño de Interfaz de Usuario con el asesoramiento de un Diseñador Gráfico, con el fin de hacer más amigable la aplicación al usuario basándose en los estándares establecidos por el Documento de Arquitectura de la Facultad 7 perteneciente a la UCI.
- ✓ Realizar pruebas de carga al sistema con mayor cantidad de información almacenada en la Base de Datos y mayor número de usuarios interactuando al mismo tiempo con el fin de comprobar el tiempo de respuesta de las funcionalidades implementadas.
- ✓ Mejorar la interacción entre el cliente y los desarrolladores, con el fin de lograr una mejor adecuación a la realidad del sistema informático desarrollado y garantizar un nivel de aceptación óptimo por parte del usuario final.
- ✓ Esforzarse por lograr un mayor compromiso con el trabajo por parte de todos los integrantes del proyecto durante el desarrollo de futuras versiones.

## REFERENCIAS BIBLIOGRÁFICAS

1. **Delgado Garcia, Gregorio.** Etapas del desarrollo de la salud pública revolucionaria cubana. *Rev Cubana Salud Pública.* [En línea] ene.-jun. de 1996. [Citado el: 27 de noviembre de 2007.] http://scielo.sld.cu/scielo.php?pid=S0864-34661996000100011&script=sci_arttext. ISSN 0864-3466.

2. **Delgado Ramos, Ariel y Vidal Ledo, María.** Informática en la salud pública cubana. *Rev Cubana Salud Pública.* [En línea] Julio-Septiembre de 2006. [Citado el: 27 de noviembre de 2007.] http://bvs.sld.cu/revistas/spu/vol32_3_06/spu15306.htm.

3. *Urgencia, Emergencias, Terapias y Ambulancias. Proyecto de resolución de urgencias. Informe del SIUM.* **Sosa Acosta, Dr. Alvaro.** Habana Cuba : s.n., 2005, Informe del SIUM.

4. **Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Lawrence G. Roberts, Stephen Wolff.** Una breve historia de Internet . *ATI.* [En línea] [Citado el: 24 de 1 de 2008.] http://www.ati.es/DOCS/internet/histint/.

5. Las Aplicaciones Web. *Dimagin.* [En línea] [Citado el: 10 de diciembre de 2007.] http://www.dimagin.net/es/contenido.php?t_id=6.

6. **Colado Rodríguez, Cesar.** Diseño y desarrollo de aplicaciones web. *Germinus.* [En línea] Febrero de 2003. [Citado el: 10 de noviembre de 2007.] http://www.germinus.com/sala_prensa/articulos/Diseno_desarr_aplicaciones_web_multidispo%20(Febrero%202003).pdf.

7. Servidor web. *Wikipedia.* [En línea] http://es.wikipedia.org/wiki/Servidor_web.

8. Servidor Web. *Ingenieros Consultores.* [En línea] 2005 - 2007. [Citado el: 13 de noviembre de 2007.] http://www.ingenierosconsultores.net/productosyservicios/servidorweb.html.

9. Servidor Web. *LinuxParaTodos.* [En línea] 4 de mayo de 2007. [Citado el: 13 de novembre de 2007.] http://www.linuxparatodos.net/portal/staticpages/index.php?page=servidor-web.

10. **Vegas, Jesús.** HyperText Transfer Protocol, HTTP. [En línea] 21 de febrero de 2002. [Citado el: 2 de febrero de 2008.] http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node13.html.

11. Navegador Web o Browser. [En línea] [Citado el: 14 de diciembre de 2007.] http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node19.html.

12. Guía Breve de Servicios Web. *W3C.* [En línea] [Citado el: 5 de febrero de 2008.] http://www.w3c.es/divulgacion/guiasbreves/ServiciosWeb.

13. Servicio Web. *Wikipedia.* [En línea] [Citado el: 5 de febrero de 2008.] http://es.wikipedia.org/wiki/Servicio_Web.

14. —. *El Proceso unificado de Software.* 2000.

15. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El lenguaje Unificado de Modelado.Manual de Referencia.*

16. Qué es Javascript. *Desarrollo Web.* [En línea] [Citado el: 10 de febrero de 2008.] http://www.desarrolloweb.com/articulos/25.php.

17. **Gutierrez, Javier J.** LSI(¿Que es un Framework?). *Lenguajes y Sistemas Informáticos.* [En línea] [Citado el: 20 de febrero de 2008.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

18. **Burbano, Diego Javier.** Analisis Comparativo de Sistemas de Bases de Datos de Codigo Abierto. *MySQL hispano.* [En línea] 17 de mayo de 2006. [Citado el: 2 de marzo de 2008.] http://www.mysql-hispano.org/page.php?id=43.

19. Macromedia Dreamweaver 8 Español. [En línea] 17 de Noviembre de 2007. [Citado el: 8 de marzo de 2008.] http://www.guanakoo.org/archive/index.php/t-16231.html.

20. **Alvarez, Miguel Angel.** Evaluando Zend Studio. *Maestros del web.* [En línea] [Citado el: 14 de Marzo de 2008.] maestrosdelweb.

21. monografias.com. *Diseñando Aplicaciones Distribuidas.* [En línea] [Citado el: 2 de junio de 2008.] http://www.monografias.com/trabajos14/aplicacion-distrib/aplicacion-distrib.shtml.

22. **Ciudad Ricardo, Ing. Febe Ángel.** Utilización del Patrón Modelo – Vista – Controlador (MVC) en el diseño de software educativos. *Monografia.* [En línea] Abril de 2006. [Citado el: 2 de Marzo de 2008.] http://www.monografias.com/trabajos43/patron-modelo-vista/patron-modelo-vista2.shtml.

23. *Arquitectura Cliente/Servidor.* [En línea] [Citado el: 10 de enero de 2008.] http://www.csi.map.es/csi/silice/Global71.html.

24. TECNOLOGIA CLIENTE SERVIDOR. *INEI.* [En línea] Instituto Nacional de Estadística e Informática (INEI),Peru. [Citado el: 10 de enero de 2008.] http://www.inei.gob.pe/biblioineipub/bancopub/inf/lib5038/indice.HTM.

## BIBLIOGRAFÍA

1. **Delgado Garcia, Gregorio.** Etapas del desarrollo de la salud pública revolucionaria cubana. *Rev Cubana Salud Pública.* [En línea] ene.-jun. de 1996. [Citado el: 27 de noviembre de 2007.] http://scielo.sld.cu/scielo.php?pid=S0864-34661996000100011&script=sci_arttext. ISSN 0864-3466.

2. **Delgado Ramos, Ariel y Vidal Ledo, María.** Informática en la salud pública cubana. *Rev Cubana Salud Pública.* [En línea] Julio-Septiembre de 2006. [Citado el: 27 de noviembre de 2007.] http://bvs.sld.cu/revistas/spu/vol32_3_06/spu15306.htm.

3. *Urgencia, Emergencias, Terapias y Ambulancias. Proyecto de resolución de urgencias. Informe del SIUM.* **Sosa Acosta, Dr. Alvaro.** Habana Cuba : s.n., 2005, Informe del SIUM.

4. **Barry M. Leiner, Vinton G. Cerf, David D. Clark, Robert E. Kahn, Leonard Kleinrock, Daniel C. Lynch, Jon Postel, Lawrence G. Roberts, Stephen Wolff.** Una breve historia de Internet . *ATI.* [Online] [Cited: 1 24, 2008.] http://www.ati.es/DOCS/internet/histint/.

5. Las Aplicaciones Web. *Dimagin.* [En línea] [Citado el: 10 de diciembre de 2007.] http://www.dimagin.net/es/contenido.php?t_id=6.

6. **Colado Rodríguez, Cesar.** Diseño y desarrollo de aplicaciones web. *Germinus.* [En línea] Febrero de 2003. [Citado el: 10 de noviembre de 2007.] http://www.germinus.com/sala_prensa/articulos/Diseno_desarr_aplicaciones_web_multidispo%20(Febrero%202003).pdf.

7. Servidor web. *Wikipedia.* [En línea] http://es.wikipedia.org/wiki/Servidor_web.

8. Servidor Web. *Ingenieros Consultores.* [En línea] 2005 - 2007. [Citado el: 13 de noviembre de 2007.] http://www.ingenierosconsultores.net/productosyservicios/servidorweb.html.

9. Servidor Web. *LinuxParaTodos.* [En línea] 4 de mayo de 2007. [Citado el: 13 de novembre de 2007.] http://www.linuxparatodos.net/portal/staticpages/index.php?page=servidor-web.

10. **Vegas, Jesús.** HyperText Transfer Protocol, HTTP. [En línea] 21 de febrero de 2002. [Citado el: 2 de febrero de 2008.] http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node13.html.

11. Navegador Web o Browser. [En línea] [Citado el: 14 de diciembre de 2007.] http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node19.html.

12. Guía Breve de Servicios Web. *W3C.* [Online] [Cited: febrero 5, 2008.] http://www.w3c.es/divulgacion/guiasbreves/ServiciosWeb.

13. Servicio Web. *Wikipedia.* [En línea] [Citado el: 5 de febrero de 2008.] http://es.wikipedia.org/wiki/Servicio_Web.

14. Qué es Javascript. *Desarrollo Web.* [Online] [Cited: febrero 10, 2008.] http://www.desarrolloweb.com/articulos/25.php.

15. **Gutierrez, Javier J.** LSI(¿Que es un Framework?). *Lenguajes y Sistemas Informáticos.* [En línea] [Citado el: 20 de febrero de 2008.] http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf.

16. **Burbano, Diego Javier.** Analisis Comparativo de Sistemas de Bases de Datos de Codigo Abierto. *MySQL hispano.* [En línea] 17 de mayo de 2006. [Citado el: 2 de marzo de 2008.] http://www.mysql-hispano.org/page.php?id=43.

17. Macromedia Dreamweaver 8 Español. [Online] Noviembre 17, 2007. [Cited: marzo 8, 2008.] http://www.guanakoo.org/archive/index.php/t-16231.html.

18. **Alvarez, Miguel Angel.** Evaluando Zend Studio. *Maestros del web.* [En línea] [Citado el: 14 de Marzo de 2008.] maestrosdelweb.

19. monografias.com. *Diseñando Aplicaciones Distribuidas.* [Online] [Cited: junio 2, 2008.] http://www.monografias.com/trabajos14/aplicacion-distrib/aplicacion-distrib.shtml.

20. **Ciudad Ricardo, Ing. Febe Ángel.** Utilización del Patrón Modelo – Vista – Controlador (MVC) en el diseño de software educativos. *Monografia.* [En línea] Abril de 2006. [Citado el: 2 de Marzo de 2008.] http://www.monografias.com/trabajos43/patron-modelo-vista/patron-modelo-vista2.shtml.

21. *Arquitectura Cliente/Servidor.* [En línea] [Citado el: 10 de enero de 2008.] http://www.csi.map.es/csi/silice/Global71.html.

22. TECNOLOGIA CLIENTE SERVIDOR. *INEI.* [En línea] Instituto Nacional de Estadística e Informática (INEI),Peru. [Citado el: 10 de enero de 2008.] http://www.inei.gob.pe/biblioineipub/bancopub/inf/lib5038/indice.HTM.

23. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Proceso unificado de Software.* 2000.

24. —. *El lenguaje Unificado de Modelado.Manual de Referencia.*

25. **Reynoso, Carlos y Kiccillof, Nicolás.** *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* 2004.

26. **Billy Reynoso, Carlos.** *Introducción a la Arquitectura de Software.* 2004.

27. CodeIgniter User Guide Version 1.6.1. [En línea] 1.6.1. [Citado el: 20 de Noviembre de 2007.] http://codeigniter.com/user_guide/.

28. Manual de PHP. [En línea] [Citado el: 23 de noviembre de 2007.] http://es.php.net/manual/es/.

29. MySQL con Clase. [En línea] [Citado el: 10 de enero de 2008.] http://mysql.conclase.net/curso/index.php.

30. Zend Studio. *tufunción.* [En línea] [Citado el: 12 de febrero de 2008.] http://www.tufuncion.com/zend-studio.

## ANEXOS

Anexo #1: Instrumento actual para las demandas de pacientes a provincia en el CCEMN.

DEMANDA DE PACIENTES PARA TRASLADO A PROVINCIA. _____

| Fecha | Activa | Paciente | Edad | Diagnostico | Origen | Destino | C | S | Fecha Salida |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

Anexo # 2: Instrumento actual para recogida de datos de las entradas de móviles de provincias a Ciudad de la Habana.

**ENTRADA PLANIFICADA DE MOVILES DE LAS PROVINCIAS**

| FECHA | CHAPA | PROVINCIA | MUNICIPIO | HOSPITAL | DIAGNOSTICO | ENTRADA | SALIDA | PACIENTE | RETORNO | OBSERVACIONES |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |
| | | | | | | | | | | |