# UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

## Facultad 7

# Título: *"Centro de Control para el Sistema de Información para la Salud"*

**TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE**

**INGENIERO EN CIENCIAS INFORMÁTICAS**

**Autores:** Annia Arencibia Morales

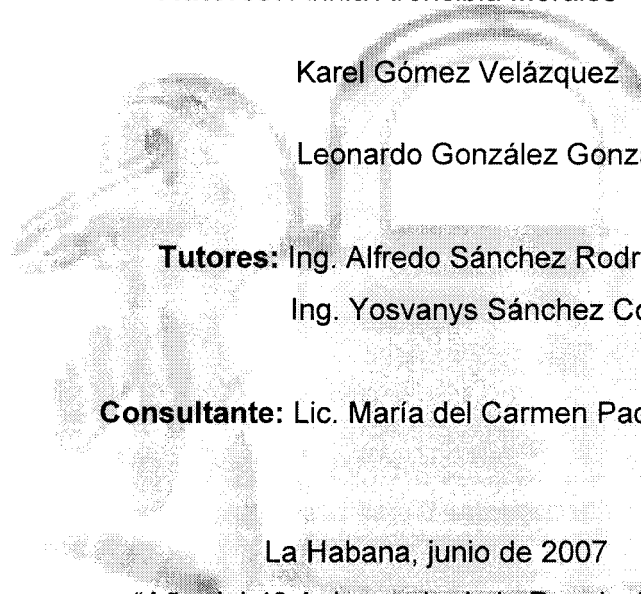
Karel Gómez Velázquez

Leonardo González González

**Tutores:** Ing. Alfredo Sánchez Rodríguez

Ing. Yosvanys Sánchez Corales

**Consultante:** Lic. María del Carmen Paderni López



La Habana, junio de 2007

"Año del 49 Aniversario de la Revolución"

## AGRADECIMIENTOS

*A nuestro Comandante Fidel y a la Revolución por habernos dado la oportunidad de formarnos en una Universidad del Futuro.*

*A Mirna y Alfredo por confiar en nosotros para realizar esta investigación y contribuir extraordinariamente en la formación profesional que hemos adquirido, a lo largo de estos tres años de trabajo.*

*A todos nuestros profesores de la UCI y de la Empresa SOFTEL.*

*A nuestros amigos y a todas aquellas personas que han brindado su apoyo para el desarrollo exitoso de nuestro Trabajo de Diploma.*

*En especial a nuestras familias, por guiarnos ejemplarmente por el camino más certero.*

# DEDICATORIA

*A las personas más importantes en mi vida, mis padres: Alexis, Mercedes y mi tía mamá Pilar, por apoyarme en todo momento y depositar toda su confianza en mí. A mi hermano Anniel, para inspirarlo a esforzarse por llegar a ser alguien en la vida. A mis compañeros de tesis Karel y Leonardo por haberme escogido para la realización de este trabajo, a mi novio que ha estado siempre a mi lado para apoyarme, a todos Uds. va dedicado este trabajo.*

*Annia Arencibia Morales*

*A toda mi familia, en especial a mis padres que fomentaron en mí las ansias de superación y me han brindado su apoyo incondicional en todo momento. A mis compañeros de estudio de la Universidad, sobre todo a Annia y Karel con los que he compartido la recta final para convertirme en un profesional acorde a los tiempos actuales.*

*Leonardo González González*

*A mamí y papí quienes han sido un ejemplo certero a seguir, inculcándome las ansias de superación continua. A todas las personas que han puesto su granito de arena en mi formación profesional: Mima, Alfredo, César, Valido. A Leo y Annia por su abnegación y responsabilidad.*

*Karel Gómez Velázquez*

# RESUMEN

Como parte del programa general para informatizar el Sistema Nacional de Salud cubano, se desarrolla en el 2003, el Registro Informatizado de Salud (RIS). Posteriormente se concibe el Sistema de Información para la Salud (SISalud), que es una plataforma única para la gestión; procesamiento y transmisión de la información clínica, basado en la misma arquitectura que el RIS y conteniéndolo.

La investigación, está encaminada a obtener un producto de software, que heredando los requerimientos del actual módulo de administración y seguridad del RIS, sea capaz de perfeccionar y optimizar los procesos de administración de usuarios y asignación de privilegios a estos. Así como el mantenimiento de la integridad referencial en el flujo de información distribuida, que se establecerá entre los componentes a integrarse en SISalud.

El Centro de Control aporta un conjunto de beneficios como la agilización de los ciclos de desarrollo. También, proporciona el mantenimiento de la integridad referencial entre los componentes, protegiéndose así la información contra estados corruptos e inconsistentes. El proceso se realizó utilizando software no propietario, respondiendo a una de las políticas del Ministerio de Salud Pública (MINSAP) para el desarrollo de sus soluciones informáticas.

***Palabras claves:*** *Sistema Nacional de Salud, Informatizar, Seguridad, Información Clínica.*

**TABLA DE CONTENIDOS**

# INTRODUCCIÓN

El sustancial mejoramiento de la infraestructura tecnológica o la masiva y profunda preparación del capital humano desde edades tempranas, son ejemplos de ingentes esfuerzos del Estado Socialista por transitar aceleradamente hacia la Informatización de la Sociedad Cubana, como vía para aumentar la calidad de vida, la eficiencia y la competitividad del país, garantizando la estabilidad, confiabilidad, vitalidad, seguridad e inviolabilidad de estas tecnologías. [1]

El Sistema Nacional de Salud cubano (SNS) viene realizando importantes reformas como parte de las transformaciones del período revolucionario y no; como una imposición de los tiempos actuales en los que reina la unipolaridad, globalización y crisis económica. Sino como una tarea priorizada de la Revolución por alcanzar una equidad social sin precedentes en la historia. El uso de la informática en la medicina es una de las aplicaciones más comunes e importantes desde hace ya varias décadas, permitiendo no sólo contar con métodos novedosos para la gestión administrativa en consultas, hospitales y centros de investigación biomédica, sino también disponer de sistemas automatizados, que apoyen al diagnóstico, tratamiento y rehabilitación de los problemas de salud.

El proceso de informatización del SNS tiene como misión general, desde el año 2000, la implementación de un programa general teniendo al policlínico como centro gestor de la información, que se apoye en las estrategias y políticas trazadas por la dirección del país y el Ministerio de Salud Pública (MINSAP), organismo rector del SNS, de manera que se logre la incorporación progresiva y sistemática de las Nuevas Tecnologías de la Información y las Comunicaciones (NTIC) en los procesos de salud. El SNS está organizado por los niveles de dirección nacional, provincial y municipal, en correspondencia con la división político-administrativa del país y según la complejidad de las acciones preventivas, curativas y de rehabilitación, así como el grado de especialización de los servicios, su organización responde a los niveles de Atención Primaria, Atención Secundaria y Atención Terciaria de Salud.

En particular, en esta nueva etapa de fortalecimiento del Sistema Nacional de Salud; la Atención Primaria de Salud (APS) es el eje fundamental de las transformaciones. A través de las cuales, se pretende consolidar el Nivel Municipal de Salud; para dar cumplimiento al principio de descentralización de las soluciones de los problemas de salud de la comunidad y acercar eficiente y gradualmente la prestación de estos a la población. La informatización de los procesos de la APS tienen como objetivo principal analizar,

diseñar y desarrollar un producto de software, que cumpla con los lineamientos establecidos por el MINSAP, facilitando la gestión de la información en este nivel de atención y permitiendo el flujo de la misma hacia los diferentes niveles de toma de decisiones.

La Empresa de Soluciones Informáticas SOFTEL, entidad del Ministerio de Informática y las Comunicaciones (MIC), en el año 2003, reorienta su trabajo hacia el desarrollo de productos y servicios informáticos, que eleven la eficiencia del sistema de salud cubano y se ganen un espacio en el mercado internacional. Para el desarrollo de su labor, la empresa dispone de consultores especializados en diseño, implantación y gestión de soluciones, cobertura nacional de servicios y soporte técnico, así como especialistas de gran profesionalidad encargados del desarrollo de sistemas informáticos para este sector. SOFTEL se inserta en la Infraestructura Productiva (IP) de la Universidad de las Ciencias Informáticas y se vincula a los estudiantes y profesores de la Facultad 7, que se forman en un segundo perfil de informática para la salud con el objetivo de cumplir la misión encomendada.

La integración de las soluciones informáticas desarrolladas para el sector de la salud es un hecho ineludible, en la actualidad estas se comportan como islas de información, lo cual trae consigo su duplicación y por consiguiente la falta de integridad. Existen antecedentes de intentos para lograr este objetivo pues empresas como SOFTEL, CENTERSOFT y VIRTUS, que aproximadamente en 1997, vieron la necesidad de acoplar sus aplicaciones con el fin de satisfacer las exigencias comerciales asociadas a la exportación de software y lograr una gama de productos para el sector de la salud mucho más valiosa a la que cada empresa individualmente poseía. Este experimento fue muy complejo y costoso, pues implicó la realización de ajustes al código fuente de los sistemas, además de que dependía totalmente de las partes involucradas y los desarrolladores pertenecientes a estas entidades, no existiendo una política trazada para la reutilización. [2]

El Registro Informatizado de Salud (RIS) desplegado en la Red Telemática INFOMED en diciembre de 2003, sentó las bases para la existencia de un sistema constituido por siete aplicaciones independientes, desarrolladas sobre una arquitectura Orientada a Servicios (SOA) y Basada en Componentes (CBA). Esto permitió que cada una de ellas gestionara de manera eficiente la información correspondiente a su negocio y se comunicaran entre sí utilizando Servicios Web basados en XML según las necesidades del flujo de información; ofreciéndole al usuario final una visión integrada de los datos almacenados, los

cuales podrían ser utilizados por los diferentes niveles de dirección, la docencia, la investigación y la gestión de salud.

Con el fin de crear una plataforma única para la gestión, procesamiento y transmisión de la información clínica se concibe en el año 2004, el Sistema de Información para la Salud (SISalud), infraestructura en la cual deberán integrarse todos los sistemas o soluciones que se conciban, con el fin de dar cumplimiento al programa general para informatizar el SNS. Mediante esta infraestructura se pretende abordar los problemas del sector de forma integral e innovadora, satisfaciendo las necesidades de los clientes en todos los niveles de salud, desde la Atención Primaria hasta las redes de Especialidades en los diferentes Institutos y Hospitales. SISalud estará compuesto por el Registro Informatizado de Salud (RIS), el Sistema Informatizado de Atención Primaria (SIAP), el Sistema Informatizado de Gestión Hospitalaria (SIGH) y el Sistema Informatizado de Atención Especializada (SIAE).

El módulo de Administración y el componente asociado SAAA (Single Authentication Authorization and Account) del Registro Informatizado de Salud en su versión actual, no están concebidos para administrar una infraestructura como SISalud, donde pueden integrarse componentes heterogéneos, en cuanto a su negocio e información manipulada, con diferente distribución física. Cuenta con un limitado nivel en cuanto a la gestión para la administración de usuarios y no permite validar y homologar los datos personales de los usuarios con un registro externo que contenga los datos de los ciudadanos que son recogidos por la Oficina del Carné de Identidad.

Asociado al proceso de administración de usuarios, sólo se concibió que los administradores de un nivel determinado registraran los administradores del nivel inmediato inferior, no teniendo en cuenta que existen Unidades de Salud que no se subordinan directamente a una Dirección Municipal de Salud determinada, sino a la Dirección Provincial e inclusive a la Dirección Nacional, por lo que estos administradores actualmente se están creando con privilegios que no les corresponde. Por otra parte los tipos de usuarios del RIS están predefinidos, no pudiéndose crear nuevas clasificaciones o roles para los usuarios, los tipos de usuario predefinidos son administradores, editores o visualizadores, representando los privilegios que pueden ser asignados a los usuarios sobre las funcionalidades y no la función que realizan en el SNS.

Del mismo modo, en la versión actual del SAAA el mantenimiento de la integridad referencial en el flujo de información se realiza parcialmente, utilizando un grupo de tablas denominadas caché, en las bases de datos de los componentes, contenedoras de información no asociada a su negocio de frecuente utilización, las que tienen que ser objeto de constantes actualizaciones en forma manual. Asimismo no es posible configurar las restricciones de integridad establecidas entre los componentes ni especificar el tipo de dependencia, como restrictiva o en cascada.

Estos son algunos de los motivos que en la fase de conceptualización de los nuevos módulos, conllevaron a la necesidad de desarrollar el Centro de Control, el cual será una versión renovada que sustituirá al SAAA actual. Entre sus funcionalidades previstas se encuentran la configuración de componentes y relaciones de dependencia que se establezcan entre sí, facilitando de esta manera el enrutamiento de las peticiones y el mantenimiento de la integridad en el flujo de información distribuida; además de una administración eficiente de los usuarios y la asignación de sus privilegios en dependencia de su nivel y tipo, pudiéndose crear nuevas clasificaciones para este último atributo. También abarcará una optimización en la política de trazabilidad para SISalud, de tal manera que se tenga un control estricto de las operaciones en las que se involucran los usuarios.

En este sentido se manifiestan los siguientes **problemas a resolver** por el Centro de Control:

I. Insuficiente nivel de gestión para la administración de usuarios y asignación de privilegios por el componente SAAA del RIS.

II. Inadecuado proceso de gestión para preservar la integridad referencial, sobre el flujo de información distribuida por el componente SAAA del RIS.

**Objeto de estudio:** Proceso de desarrollo del Sistema de Información para la Salud (SISalud).

**Campo de Acción:** Proceso de gestión de requerimientos de seguridad para el Sistema de Información para la Salud (SISalud).

Para resolver los problemas identificados se propone el siguiente **objetivo general**: Adaptar el componente SAAA del RIS, de manera tal que permita una gestión eficiente de los requerimientos de seguridad y garantice la integridad referencial en el flujo de información distribuida gestionada por el Sistema de Información para la Salud.

Para guiar la investigación se proponen las siguientes **ideas a defender:**

1. Con el desarrollo del Centro de Control se obtendrá una versión renovada del componente de seguridad SAAA del RIS.

2. La implementación de la solución al problema crítico de arquitectura existente, garantizará el control de la integridad referencial en los datos gestionados por SISalud.

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

1. Estudio de tendencias y tecnologías actuales para llevar a cabo el proceso de desarrollo del Centro de Control, enfatizando en los patrones de arquitectura y diseño: Modelo Vista Controlador (MVC), Arquitectura en capas, Single Sign On (SSO) y Observador - Observable.

2. Asimilación de la arquitectura definida por el MINSAP (Orientada a Servicios y Basada en Componentes (SOA-CBA)) para el desarrollo de sus aplicaciones, Plataforma de Servicios (PlaSer) y Registro Informatizado de Salud (RIS), así como el negocio actual del componente SAAA del RIS.

3. Asimilación de la metodología de desarrollo definida por la Empresa SOFTEL, basado en el Proceso Unificado de Desarrollo (RUP), para los Flujos de Trabajo "Modelamiento del Negocio", "Gestión de Requerimientos", "Diseño" e "Implementación".

4. Diseño de una base de datos capaz de organizar y almacenar de manera eficiente la información que es manipulada por el sistema, que soporte integridad referencial y transacciones.

5. Implementación de los métodos del negocio y capa de presentación para el Centro de Control, así como de la solución al problema crítico de arquitectura asociado a la garantía de la integridad referencial en la información distribuida.

En sentido general se puede destacar que el desarrollo del Centro de Control proporcionará un grupo de **beneficios** para la informatización del SNS, entre los que se pueden mencionar los siguientes:

1. Gestión eficiente de requerimientos de seguridad para todos los sistemas informáticos que solucionen problemas asociados a la salud pública, permitiendo la agilización del proceso de construcción de estos, donde los desarrolladores podrán obviar tales mecanismos debido a que serán proporcionados de manera automática por SISalud, en el momento de la integración.

2. Protección a la información manejada por SISalud evitando su corrupción e inconsistencia, pues proporcionará chequeos de integridad sobre la información e implementará el control de la integridad referencial en el flujo de información que se intercambiará entre los componentes.

3. El proceso de integración es un hecho inevitable en la informatización del SNS, complementado con el Centro de Control, el cual evitará que cada sistema posea de manera aislada la administración de sus usuarios ya que esta información será almacenada y gestionada centralizadamente.

4. Perfeccionamiento de los procesos de gestión de usuarios y asignación de privilegios.

5. Homologación de los datos de los usuarios con el Registro de Ciudadanos (RC), debido a necesidades de los nuevos módulos, así como para la agilización en los procesos de búsquedas de usuarios en el sistema.

6. La eliminación de los usuarios y sus respectivas trazas es lógica y no física, fortaleciéndose el proceso de auditoría.

7. Generación automática de ficheros de configuración y despliegue, evitándose la ocurrencia de errores o equivocaciones.

8. Implementación del patrón de diseño Observador-Observable para el mantenimiento eficaz de la integridad referencial en el flujo de información, a nivel de componentes distribuidos.

El presente documento se encuentra estructurado en cuatro capítulos, el primero de ellos, **"FUNDAMENTACIÓN TEÓRICA"**, ubica al lector en el ambiente de desarrollo del Centro de Control, justificándose las tendencias, tecnologías, metodologías y herramientas actuales que fueron utilizadas. Seguidamente el capítulo dos, **"CARACTERÍSTICAS DEL SISTEMA"**, contiene un marco conceptual asociado a la información que será manipulada por el sistema, llegándose a un acuerdo con el cliente sobre las funcionalidades y requerimientos deseados. El tercer capítulo **"DISEÑO DEL SISTEMA"** se centra en la modelación detallada y la construcción de la estructura de la aplicación, obedeciendo a la arquitectura definida por el MINSAP para sus soluciones informáticas.

El cuarto y último capítulo, **"IMPLEMENTACIÓN"**, contiene una justificación de la integración del Centro de Control con sistemas externos, describe cómo será desplegado y se detalla la propuesta de solución para garantizar la integridad referencial en el flujo de información de SISalud. También se arriba a conclusiones y se realizan un grupo de recomendaciones a tener en cuenta para futuras versiones del sistema.

# CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

## 1.1 Introducción

El presente capítulo tiene como objetivo abordar los diferentes elementos que brindan la base teórico conceptual para el desarrollo del Centro de Control. Se exponen además los antecedentes e intentos de integración entre sistemas informáticos en el área de la salud pública, tomando esto como punto de partida para la concepción de la solución en cuestión. Además se realiza un análisis de las técnicas, tecnologías, metodologías y herramientas de software sobre los cuales se llevará a cabo el proceso de desarrollo.

## 1.2 Sistema Nacional de Salud

La forma y los métodos que sirven de base para la organización de la atención a la salud en un país determinado, es lo que se conoce como Sistema Nacional de Salud (SNS). La Organización Mundial de la Salud lo define como: «Un complejo de elementos interrelacionados que contribuyen a la salud en los hogares, los lugares de trabajo, los lugares públicos y las comunidades, así como el medio ambiente físico y psicosocial en el sector de salud y otros sectores afines. Además es el conjunto de unidades administrativas, de producción, investigación y servicios, responsabilizado con la atención integral de la salud de una población». [3]

El SNS cubano está estructurado en tres niveles de dirección, los cuales se encuentran identificados con la estructura político – administrativa del país. Estos niveles son los siguientes:

- ✓ **El Nivel Nacional**, representado por el **Ministerio de Salud Pública (MINSAP),** como órgano rector con funciones metodológicas, normativas, de coordinación y de control en la aplicación de las políticas del Estado y el Gobierno en cuanto a la salud pública, el desarrollo de las Ciencias Médicas y la Industria Médico Farmacéutica.

- ✓ **El Nivel Provincial**, representado por las **Direcciones Provinciales de Salud**, directamente subordinadas, administrativa y financieramente a la Asamblea Provincial del Poder Popular (órgano de gobierno a esa instancia).

- ✓ **El Nivel Municipal**, representado por las **Direcciones Municipales de Salud** y dependientes, administrativa y financieramente de la Asamblea Municipal del Poder Popular (órgano de gobierno a esa instancia).

**(Ver Anexo I)**

De acuerdo con la complejidad de las acciones preventivas, curativas y de rehabilitación, así como la especialización de los servicios de salud brindados, los diferentes niveles de atención médica se han organizado en:

- ✓ **Atención Primaria de Salud (APS):** Da solución aproximadamente al 80 % de los problemas de salud de la población y que correspondan con las acciones de promoción y protección de la salud. Aunque sus actividades se realizan en cualquier unidad del SNS, están relacionados fundamentalmente con las que se realizan en el Policlínico y en los Consultorios del Médico de la Familia, Hospitales Rurales, Dispensarios y Postas Médicas.

- ✓ **Atención Médica Secundaria:** Este nivel da cobertura a cerca del 15 % de los problemas de salud, su función fundamental es tratar al hombre ya enfermo, tanto desde el punto de vista individual como colectivo, pero también desempeña funciones de rehabilitación, promoción y prevención de la salud. Se llevan a cabo acciones de salud más complejas y especializadas (Especialidades). Comprende la atención médica brindada en los distintos Hospitales.

- ✓ **Atención Médica Terciaria:** El nivel terciario debe abarcar alrededor del 5 % de los problemas de salud, relacionados con secuelas o aumento de las complicaciones de determinadas dolencias. Se brindan servicios de muy alta complejidad, con la óptima utilización de los recursos y medios existentes en los mismos y el desarrollo de la investigación. A este nivel pertenecen los Institutos y Hospitales especializados.

En la actualidad se ha concebido un diseño estratégico para la salud pública cubana con el propósito de elevar la calidad de la atención médica, mejorar los niveles de salud, aumentar la eficiencia económica dentro del sistema, incrementar el nivel de satisfacción de la población y brindar una mayor atención al hombre, aprovechando las fortalezas y las oportunidades en el desarrollo ya alcanzado de los recursos humanos, destacándose el fortalecimiento de la descentralización, la intersectorialidad y la participación comunitaria en salud. Se trata de modernizar y generar un nuevo tipo de modelo sanitario con el desarrollo de los niveles operativos del sistema de salud y de la sociedad en su conjunto. [4]

### 1.2.1 Informatización del Sistema Nacional de Salud

La informatización del SNS está apoyada en estrategias y políticas trazadas por la dirección del país y el MINSAP, siendo esta una tarea de vital y prioritaria importancia. Con este proceso, se pretende crear una infraestructura informática para el sector, al que se integrarán todos los productos o servicios, respondiendo a una arquitectura Orientada a Servicios y Basada en Componentes (SOA-CBA). Permite que todas las unidades de salud del país alcancen un nivel de informatización elevado en las actividades que realicen, influyendo directamente en el aumento gradual de la eficiencia del personal de salud y en la calidad de los servicios que se brinden a la población. Algunas de las políticas a tener en cuenta para la informatización del SNS son las siguientes: [5]

1. Alinearse con tecnologías de punta, estándares de calidad desarrollados en el mundo adecuándolos a nuestras condiciones particulares.

2. Para garantizar su viabilidad, sostenibilidad y mantenimiento se basarán fundamentalmente en la Dirección Integrada de Proyectos.

3. Todas las inversiones y proyectos que se desarrollen para el SNS deben considerar el elemento informático desde su concepción inicial.

4. La superación y especialización de la informática en salud será una actividad básica para la formación de los recursos humanos.

5. Los productos se desarrollarán basados en tecnología LAMP (Linux, Apache, MySQL y PHP) para garantizar sus sostenibilidad en el tiempo donde se emplearán estándares internacionales para productos relacionados con la salud pública.

Por otra parte, los proyectos que se coordinen con otros organismos y entidades dentro o fuera del país, están en la obligación de reconocer y cumplir las políticas e intereses del MINSAP y admitir la evaluación, control y certificación de las soluciones informáticas para el sector de la salud pública. Esta solución integral para la informatización significa la articulación de un nuevo paradigma para la prestación de servicios de salud, regido por el principio básico de lograr la descentralización, permitiendo que estos puedan ser accesibles por la población en su propio escenario social. Los principales impactos esperados son los siguientes: [6]

Para el SNS:

- ✓ Gestión oportuna de una información confiable y actualizada que propiciará una optimización considerable de recursos, con su lógico resultado en la reducción significativa de costos de operación de las entidades que conforman el SNS.

- ✓ Elevación de la capacidad y calidad de las tomas de decisiones asistenciales y gerenciales por la disposición oportuna de información actualizada para todos los niveles del SNS, que permitirá una rápida transferencia de la información sanitaria de un paciente.

- ✓ Disponer de un soporte y herramientas poderosos para la formación y actualización constante de sus miembros, desde sus propios escenarios de desempeño, con la consecuente equidad de conocimientos independientemente de áreas geográficas y nivel de atención.

Para la población:

- ✓ Equidad en el acceso a servicios, tecnologías e información de salud independientemente del área geográfica y nivel de atención.

- ✓ Atención por un personal médico mejor preparado y actualizado, entre los cuales se podrá intercambiar información, criterios y diagnósticos; elevándose la confianza hacia el sistema de atención.
- ✓ Reducción del número de desplazamientos innecesarios entre instituciones de salud.
- ✓ Reducción de tiempos de espera para el acceso a servicios especializados.

SOFTEL, Empresa del Ministerio de Informática y las Comunicaciones (MIC), tiene la misión de generar las soluciones informáticas especializadas en salud y organizar un esquema para la prestación de los servicios informáticos en dicho sector. Esta entidad cumple estos objetivos en colaboración con la Facultad 7 de la Universidad de las Ciencias Informáticas (UCI), quien vincula a sus estudiantes y profesores líderes de proyectos a la producción desde un inicio, para lograr su formación integral en un segundo perfil relacionado con la salud. Además, se ha organizado un proceso productivo con una eficiente gestión de requerimientos, donde participan igualmente médicos y trabajadores de la salud pertenecientes a otras ramas, todos en estrecho vínculo con los especialistas de la empresa.

### 1.2.2 Antecedentes del proceso de integración de aplicaciones para la salud

Actualmente la informatización del SNS no ofrece un mecanismo único para la integración. Las instituciones de Salud Pública poseen un conjunto de aplicaciones que brindan solución a determinados problemas, pero estos se comportan como islas de información al no interactuar entre sí, para obtener un flujo lógico y coherente de la información clínica relacionada con los pacientes. Existen evidencias que manifiestan cómo los desarrolladores de aplicaciones para la salud, así como las entidades a las que estos pertenecen, han sentido la necesidad de una articulación coherente entre sus soluciones informáticas.

En el año 1997, empresas como SOFTEL, CENTERSOFT y VIRTUS, pertenecientes al MIC, dedicadas al desarrollo de aplicaciones para el sector de la salud pública, vieron en la integración de sus soluciones una alternativa para satisfacer exigencias comerciales para la comercialización y exportación de este tipo de producto. Se integraron las aplicaciones que fortuitamente estas empresas desarrollaban, siguiendo

una misma arquitectura Cliente-Servidor de dos capas, estando la capa de datos sostenida sobre SQL Server con procedimientos almacenados que describían el negocio del lado del servidor. De esta forma, las aplicaciones pudieron considerarse como componentes que con el esfuerzo de sus desarrolladores y haciendo ajustes al código fuente, podían integrarse entre sí para formar una solución integral.

La empresa de Servicios Informáticos de Holguín (VIRTUS), por su parte, desarrolló aplicaciones Web como respuesta a los proyectos de creación de las Redes Nacionales de Cardiología y Cirugía Pediátrica, surgiendo así los sistemas CardioWeb y Alianza Vital respectivamente. Los desarrolladores concibieron ambos sistemas, desde el principio, integrados a módulos del sistema Galen Hospital, en acuerdo con la empresa CENTERSOFT, quien cedió el modelo de datos y códigos fuente necesarios a VIRTUS, de manera que CardioWeb y Alianza Vital pudieran generalizarse más fácilmente en las instituciones que ya poseían este sistema para la gestión hospitalaria.

El sistema Alianza Vital presenta un estricto control de acceso que permite a cada usuario acceder sólo a los datos autorizados. Para ello, el usuario debe autentificarse y en dependencia del grupo al que pertenece, dispondrá de acceso a la información, de esta forma, la información de los pacientes queda protegida y se limita sólo a aquellos especialistas autorizados a manejarla. Se desarrolló íntegramente para trabajar sobre plataforma Web, con servidor de base de datos SQL Server y una red de bases de datos distribuidas para la consulta a nivel nacional de información con carácter estadístico y criterios de expertos. En la actualidad el principal ejemplo de integración de soluciones para la salud lo representa el Registro Informatizado de Salud.

### 1.2.2.1 Registro Informatizado de Salud (RIS)

El surgimiento de los Servicios Web basados en XML, implicó la aparición de variantes mucho más factibles para la integración. El Registro Informatizado de Salud (RIS) constituyó la aplicación pionera en la utilización de esta tecnología de avanzada, debido a las ventajas que ofrecía para implementar cómputo distribuido, arquitectura orientada a servicios y basada en componentes. Además permitiría la reutilización, la no duplicación de esfuerzos y el logro de la confiabilidad e integridad de la información clínica que se gestionaría. Fue implementado a finales del año 2003 por cuatro grupos de trabajo en la

Empresa SOFTEL; integrados por especialistas de esta entidad, de la empresa ESATEL de Santiago de Cuba, del Centro para el Desarrollo Informático de la Salud (CEDISAP) e INFOMED.

Los módulos que fueron desarrollados se denominaron Registro de Equipos Médicos (REM), Registro de Equipos no Médicos (RENM), Registro de Personal de la Salud (RPS), Registro de Ciudadanos (RC), Registro de Unidades de Salud (RUS), Registro de Ubicación Geográfica (RU) y el módulo de Seguridad y Administración (SAAA). Posee tres tipos de usuario, los administradores que sólo realizan tareas de administración en los diferentes niveles del SNS, sin tener acceso a la información del sistema; los editores que son responsables de la inserción y modificación y por último los visualizadores quienes sólo tienen derecho a consultar y evaluar la información. El proceso de administración del RIS tiene carácter jerárquico, ya que el administrador nacional crea los editores y visualizadores para su propio nivel, así como los administradores para el nivel inmediato inferior. (Ver Anexo II)

El módulo de Administrador y el componente asociado SAAA del RIS están basados en un modelo de Autenticación, Autorización y Auditoría (AAA), siendo la autenticación la primera acción del usuario en el sistema y consiste en suministrar un nombre de usuario único y una contraseña que debe ser de conocimiento exclusivo de la persona que se autentica. Si el usuario autenticado no se encuentra registrado se reporta un error de acceso. En caso contrario, se autoriza su acceso y se crea un certificado digital retornándose todos los datos y permisos del usuario desglosados por módulos. Cada petición de usuario, autorizada o no, es registrada, así como el día, mes, año, hora, minuto y segundo en que se registra y si fue o no autorizada.

### 1.2.3 Sistema de Información para la Salud (SISalud)

Posteriormente en el año 2004, se concibe el Sistema de Información para la Salud (SISalud), infraestructura que permitirá la integración de los componentes, servicios o sistemas que se desarrollen para darle cumplimiento al programa de informatización del Sistema Nacional de Salud. Pretende abordar los problemas del sector de forma integral e innovadora, satisfaciendo las necesidades de los clientes en todos los niveles de salud, desde la Atención Primaria hasta las redes de Especialidades en los diferentes Institutos y Hospitales. SISalud estará compuesto por el Registro Informatizado de Salud (RIS), el Sistema

Informatizado de Atención Primaria (SIAP), el Sistema Informatizado de Gestión Hospitalaria (SIGH) y el Sistema Informatizado de Atención Especializada (SIAE).

La estructura general prevista para SISalud es la siguiente: [7]

- ✓ **Registro Informatizado de Salud (RIS):** Formado por los nomencladores o codificadores que serán administrados o gestionados a nivel nacional, que integran el **Registro No Médico Informatizado de Salud** y por los registros que pueden ser accesados desde cualquier nivel de atención o institución de salud para lograr la continuidad en el seguimiento del paciente, agrupándose estos en el **Registro Médico Informatizado de Salud**.

- ✓ **Sistema Informatizado de Atención Primaria (SIAP):** Se incluirán los componentes específicos de este nivel de atención. Estos constituirán una nueva herramienta para la transformación de los servicios que se brindan en este nivel, ya que integrarán diversos subsistemas como las actividades diarias del Equipo Básico de Salud (EBS), la dispensarización y la planificación de las acciones de salud, tanto individuales como familiares, entre otros.

- ✓ **Sistema Informatizado de Gestión Hospitalaria (SIGH):** Se agruparán los módulos que pertenecen al nivel de atención secundario u hospitalario.

- ✓ **Sistema Informatizado de Atención Especializada(SIAE):** Se encontrarán los módulos del nivel terciario de salud, que por el carácter especializado de sus servicios, sólo se brindan en determinados centros como el Instituto de Neurocirugía, Instituto de Cirugía Cardiovascular, Instituto de Nefrología, Instituto de Gastroenterología, entre otros, o en centros hospitalarios de investigación categorizados como instituciones de referencia nacional o internacional.

Además, se complementará con las soluciones asociadas a los servicios de: fisioterapia, Sistema Integrado de Urgencia Médica (SIUM), docencia médica, economía, recursos humanos, telemedicina entre otros, quienes estarán agrupados bajo la clasificación de **Otros Sistemas Informatizados en Salud**.

## 1.3 Flujo actual de los procesos involucrados en el campo de acción

El actual módulo de Administración SAAA es utilizado por los administradores del RIS en los diferentes niveles del SNS, para crear y configurar los usuarios del mismo. Entre las consideraciones actuales que posee el SAAA pueden ser mencionadas, que cada usuario puede pertenecer únicamente a un nivel del SNS. Los roles definidos para los usuarios son "Administrador" y en caso contrario serían "Editor" o "Visualizador" de uno o varios módulos del RIS, o sea un usuario puede ser "Editor Nacional" del Módulo "Registro de Unidades de Salud" y "Visualizador Nacional" del módulo "Registro de Equipos Médicos". Las tareas que pueden realizar cada uno de los tipos de usuarios antes mencionados son las siguientes:

**Administrador:** Es un tipo de usuario especial dentro el RIS, ya que sólo realiza tareas de administración en los diferentes niveles del SNS, sin tener acceso a la información gestionada por el sistema. Sólo depende del nivel de acceso, gestionando la información de los usuarios editores y visualizadores en su nivel y la de los administradores del nivel inmediato inferior. Además controla la trazabilidad de los usuarios en el sistema, llevando a cabo procesos de auditoría sobre las transacciones en las que ha intervenido un determinado usuario. Por otra parte el Administrador Nacional es el responsable de las configuraciones del sistema, registrando los métodos y componentes que se encuentren desplegados en el RIS. (Ver Anexo II)

**Editor:** Es el tipo de usuario responsable de la edición y mantenimiento de la información en los diferentes niveles, depende del nivel para el cual fue creado y del módulo del RIS al cual tiene acceso.

**Visualizador:** Este tipo de usuario sólo tiene acceso a la información para consultarla y evaluarla en los diferentes niveles, depende del nivel para el cual fue creado y del módulo del RIS al cual tiene acceso.

En el año 2005, se realizó un primer acercamiento al estudio de los procesos para la implementación de un sistema de seguridad para SISalud, momento en el cual fue desarrollada una investigación que tenía como alcance analizar y diseñar un posible sistema informático que fuese capaz de gestionar estos requerimientos, en aquella etapa aún no se había comenzado a acometer el desarrollo de los nuevos módulos, motivo por el cual no se habían identificado muchas de sus necesidades y requerimientos. Además el proceso de mantenimiento de la integridad referencial en el flujo de información clínica en forma eficiente era un elemento indispensable a tener en cuenta por el Centro de Control.

## 1.4 Situación problémica y problemas a resolver

Con el desarrollo del RIS, en diciembre del 2003, el SAAA respondía y daba solución a los requerimientos de seguridad para este conjunto de aplicaciones para la salud. Con la concepción de SISalud y el desarrollo de los nuevos módulos que lo conformarán fueron identificadas un conjunto de insuficiencias en el SAAA que influyen en la adecuada gestión de los requerimientos de seguridad para esta plataforma de integración, entre ellas pueden ser mencionadas las siguientes:

1. Los Administradores sólo pueden crear usuarios con estos mismos privilegios en el nivel inmediato inferior, es decir, el Administrador Nacional gestiona los Administradores Provinciales, estos a su vez gestionan los Administradores Municipales y finalmente estos gestionan los Administradores de Unidades de Salud. Téngase en cuenta que existen Unidades de Salud, que no se subordinan a las Direcciones Municipales de Salud Pública donde se encuentran ubicadas, sino que pueden estar subordinadas directamente a la Dirección Provincial o Nacional. En estos momentos los administradores de estas unidades, se crean con el mismo nivel y privilegios de gestión que cualquier administrador ubicado en el nivel al cual se subordinan, pudiendo manipular información que no les concierne.

2. No es posible validar la existencia de la persona asociada al usuario y la veracidad de sus datos personales; actualmente sólo se recoge el nombre y apellidos del mismo en el momento de registrar al nuevo usuario. Además los nuevos módulos necesitan identificar a la persona que se

encuentra autenticada, por ejemplo el Registro de Población (RPOB) utiliza este proceso para asignar a un médico determinado las poblaciones a las cuales les prestará asistencia médica.

3. No se pueden realizar búsquedas de usuarios en el sistema, este proceso es inadecuado en estos momentos. Debido a que un administrador en un nivel determinado, es capaz de manipular la información de usuarios que no fueron creados desde su ubicación. Un ejemplo es que un Administrador Provincial tiene acceso a los usuarios de todos los municipios del país y no sólo a los municipios de su provincia, violando la jerarquía de administración de usuarios aprobada para el RIS.

4. Escasa gestión de información asociada a los componentes, no es posible configurar sus dependencias y restricciones de integridad. No es manejado el concepto de módulo, entiéndase este como un conjunto de componentes integrados para gestionar información distribuida, la cual será mostrada en forma unificada en una misma presentación. La modularidad es una de las principales características que describen a SISalud, debido al fuerte proceso de reutilización que será llevado a cabo.

5. Los tipos de usuarios en la versión actual del SAAA, como se observó en el epígrafe anterior, son Administrador, Editor y Visualizador, no pudiéndose crear nuevas clasificaciones o roles para un componente determinado. Estas descripciones no responden a las funciones de los usuarios dentro del SNS, sino a privilegios que pueden ser asignados a los mismos. En los nuevos módulos concebidos, un mismo usuario puede realizar tareas de edición o visualización de información simultáneamente, actualmente ahora estos tipos son excluyentes.

6. El proceso de mantenimiento de la integridad en la información gestionada en el RIS no es el más adecuado, debido a que el mismo es resuelto parcialmente utilizando un conjunto de tablas denominadas caché, las cuales contienen información de frecuente utilización por un componente determinado, trayendo consigo una duplicación innecesaria de información que no pertenece al negocio del componente que las posee. Además tienen que ser objeto de un frecuente proceso de actualización en forma manual.

Por tales motivos los problemas que serán resueltos mediante el componente Centro de Control son los siguientes:

I. Insuficiente nivel de gestión para la administración de usuarios y asignación de privilegios por el componente SAAA del RIS.

II. Inadecuado proceso de gestión para preservar la integridad referencial sobre el flujo de información distribuida por parte del componente SAAA del RIS.

## 1.5 Objeto de Automatización e información manipulada

El Centro de Control y su componente asociado SAAA se encuentran estrechamente vinculados a la arquitectura aprobada para SISalud, cuyo objetivo principal es garantizar los requerimientos de seguridad para esta infraestructura. Teniendo en cuenta que este proceso debe ser muy estricto en correspondencia con la sensibilidad de la información que es intercambiada entre sus componentes, donde el paciente es el centro de toda la atención.

Por esta razón, es necesario tener centralizada la información referente a los usuarios y los privilegios que tienen estos sobre los módulos de SISalud, de tal manera que la trazabilidad esté presente en cualquier operación en la que intervengan los usuarios. Estas trazas persistirán en el tiempo, realizándose de ellas copias periódicas de seguridad, debido a que la eliminación de un usuario del sistema será lógica y no física, contándose con un registro histórico de los usuarios y operaciones realizadas por los mismos.

Para que el proceso de autorización sea más estricto, es necesario tener la relación de los métodos que conforman a cada componente; así como los permisos de ejecución que tienen los tipos de usuario de estos componentes en los diferentes niveles. Además se almacenará la URL a donde será redireccionada la petición hecha a un determinado componente. Del mismo modo, se contará con la configuración de las dependencias y restricciones de integridad que sean necesarias para los componentes, siendo usada posteriormente para el mantenimiento de la integridad en la información gestionada por SISalud. Además, será registrada la información relacionada con los módulos y los componentes que se integran en el

mismo, indicándose cuál de estos será su componente base; entiéndase como componente base de un módulo aquel que está encargado de integrar en forma unificada la información de todos los componentes que lo conforman.

## 1.6 Tendencias y tecnologías actuales a considerar

En este epígrafe se tratarán los conceptos fundamentales relacionados con la arquitectura, tecnologías, herramientas y metodología que se considerarán para el proceso de desarrollo del Centro de Control; Para ello en primer lugar es preciso definir qué es la arquitectura de software, que según Roger Pressman representa, "... la descripción de los subsistemas o componentes de un sistema informático y las relaciones entre ellos (...)" [8]. El tópico más importante asociado a esta definición, sin lugar a dudas, son los patrones, tanto en lo que concierne a patrones de diseño como a los de arquitectura.

Conviene caracterizar el escenario que ha motivado la aparición del concepto de patrón, antes de intentar definirlo. Desde los inicios de la arquitectura de software, se observó que en la práctica del diseño y la implementación, ciertas regularidades en la configuración aparecían una y otra vez como respuestas a similares demandas. Muy pronto se les llamó estilos o patrones, a estas soluciones o configuraciones estándar. A continuación se expondrán los patrones arquitectónicos y de diseño que serán utilizados para el desarrollo del Centro de Control; haciendo énfasis en los estilos cliente-servidor, modelo-vista-controlador y arquitectura en 3 capas. Estos patrones no serán expuestos agrupados según su clasificación natural de diseño o arquitectura; sino como una secuencia lógica que permita justificar el por qué de su utilización.

### 1.6.1 Sistemas distribuidos. Modelo Cliente-Servidor

Actualmente, el escenario típico de cualquier sistema informático para la salud, debe insertarse en un panorama donde coexisten numerosos sistemas heterogéneos y distribuidos en diferentes ubicaciones físicas. Es inevitable la integración de sus datos lo que consiste en la combinación de la información que se encuentra almacenada en componentes con estas características, proporcionado al usuario final una vista unificada de los mismos. Tal integración desde el punto de vista clínico; tiene el propósito de facilitar

a los usuarios, ya sea personal médico, administrativo, investigativo o al propio paciente un conglomerado de la información obtenida a partir del paciente en el proceso de atención.

Un sistema de cómputo distribuido es una colección de sistemas de cómputo autónomos, llamados nodos, interconectados a través de una red y de software de comunicaciones, capaces de cooperar para la realización de una tarea común mientras que la computación distribuida es el conjunto de modelos, técnicas y herramientas computacionales que ayudan a resolver los problemas planteados por los sistemas distribuidos en cuanto a la generación, el procesamiento y la utilización de la información requerida para su funcionamiento. [9]

El contexto tecnológico actual obliga a la distribución de los sistemas, pero hay que tener en cuenta para este proceso la utilización de una estricta política de compatibilidad, estándares, escalabilidad y tecnologías. La amplia difusión y aceptación de los sistemas distribuidos se debe a un grupo de logros, entre los que podemos destacar el descenso del precio de las computadoras personales y el desarrollo acelerado que sufrieron las redes de ordenadores. Además como los datos van a coexistir de manera autónoma implica que los componentes del sistema distribuido, deciden cuándo y cómo responder a las peticiones de información procedentes de otros sistemas, siendo necesario configurar qué datos y qué funcionalidades serán compartidas durante la integración.

Las principales ventajas de los sistemas distribuidos son las siguientes: [10]

- ✓ **Rapidez de respuesta y rendimiento:** Los recursos se encuentran compartidos, de manera que pueda ser procesada una petición determinada con mayor velocidad.

- ✓ **Fiabilidad:** En el caso de los sistemas no distribuidos, si el servidor principal falla, el sistema colapsa en su totalidad, no ofreciéndose ningún servicio hasta que este se repare. Mientras que si se realiza una distribución del sistema esta situación no implica que el mismo deje de brindar los servicios que no hayan sido afectados.

- ✓ **Escalabilidad:** Este tipo se sistema puede crecer más fácilmente que uno centralizado. Pues si se decide mejorar la productividad de este es necesario adquirir un nuevo servidor principal, mientras que si desea mejorar uno distribuido basta con incrementar el número de servidores que conforman el sistema.

- ✓ **Modularidad y reutilización en el desarrollo:** Esto permite la agilización de los procesos de desarrollo pues se tendrá en cuenta para implementar un nuevo componente qué funcionalidades podrá obtener de otro ya desplegado.

Estos elementos no descartan la existencia de desventajas, siendo una de las más significativas y comunes, la existencia de incompatibilidades en la información distribuida y la falta de integridad referencial en la misma, al no considerarse en el proceso de definición y establecimiento de la arquitectura la forma de solucionar esta situación crítica. Además que se complejiza el proceso de administración y existe una interdependencia entre desempeño y confiabilidad de la red.

Los sistemas distribuidos más extendidos responden a un modelo de arquitectura Cliente-Servidor, donde una parte de estos sistemas (los servidores) proveen operaciones comunes llamadas servicios, mientras que los otros (llamados Clientes) acceden a estos. Es decir, el Cliente es el proceso que permite al usuario formular los requerimientos y enviarlos al Servidor, quien es el encargado de atender a múltiples clientes que hacen peticiones de algún recurso administrado por él. El servidor normalmente maneja todas las funciones relacionadas con la mayoría de las reglas del negocio y los accesos a datos. Algunas ventajas de la arquitectura Cliente-Servidor son las que siguen:

- ✓ La estructura inherentemente modular facilita la integración entre sistemas heterogéneos y el crecimiento de la infraestructura computacional, favoreciendo así la escalabilidad de las soluciones.

- ✓ Rápido mantenimiento y desarrollo de aplicaciones, ya que se pueden reutilizar componentes y servicios ya existentes.

- ✓ Reducción considerable del tráfico en la red, el Cliente sólo se conecta al Servidor estrictamente cuando necesita un servicio; una vez que se obtenga la respuesta, la conexión se cierra dejándose libre la red.

Por otra parte deben tenerse en cuenta las desventajas siguientes:

- ✓ Necesidad de estrategias para el tratamiento de los errores y para mantener la consistencia de los datos.

- ✓ Pobre desempeño del sistema si la congestión de la red y el tráfico en la misma es muy elevado.

## 1.6.2 Patrones de arquitectura y diseño

Los patrones en sentido general son unidades de información nombrada, instructiva e intuitiva; que captura la esencia de una familia exitosa de soluciones probadas a un problema recurrente dentro de un cierto contexto. El objetivo de los patrones es crear un lenguaje común, para la comunidad de desarrolladores, que permita generalizar la experiencia sobre un determinado problema, así como la solución estándar que se le dará al mismo. Estos pueden referirse a distintos niveles de abstracción, desde un proceso de desarrollo hasta la utilización eficiente de un lenguaje de programación. Existen varios tipos de patrones, dependiendo del contexto particular en el cual se aplican o de la etapa en el proceso de desarrollo, algunos de estos tipos son: de Diseño, de Arquitectura, para Ambientes Distribuidos, de Negocios, de Análisis, entre otros.

### 1.6.2.1 Arquitectura Orientada a Servicios y Basada en Componentes (SOA-CBA)

Cada vez las empresas exigen aplicaciones más complejas, con menos tiempo y presupuesto. Crear estas aplicaciones, requiere en muchos casos de funcionalidades ya implementadas como parte de otros sistemas. SOA (Service Oriented Architecture) nace como una estrategia de integración, exponiendo

servicios con funcionalidades bien definidas a la aplicación que lo requiera. De esta manera, una aplicación final simplemente se provee de un conjunto de estos servicios, añade su lógica particular y le presenta una interfaz al usuario final. Una visión interna de los servicios es que los mismos funcionan como aplicaciones independientes, teniendo sus propias reglas de negocio, datos, procedimientos, operaciones y administración. Exponen toda su funcionalidad basada en una interfaz a través de mensajes, lo que implica la carencia de una interfaz de usuario.

A continuación se relacionan algunos beneficios de SOA: [11]

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

- ✓ **Interoperatibilidad de aplicaciones:** Disminución de la complejidad en el proceso de integración, pues se interactúa con elementos que se abstraen de la tecnología y ubicación de los servicios.

Hay un buen número de definiciones de componente pero una de las que es bastante operativa es la que plantea que un componente de software es una unidad de composición con interfaces especificadas contractualmente y dependencias del contexto explícitas. Por otra parte, el desarrollo de software basado en componentes se centra en el desarrollo de aplicaciones complejas mediante el ensamblado de módulos, que han sido previamente diseñados por otras personas a fin de ser rehusados en múltiples aplicaciones. Cada componente debe describir de forma completa la interfaz que ofrece, así como las interfaces que requiere para su operación y debe funcionar correctamente con independencia de los mecanismos internos que utilice para la funcionalidad de la interfaz. [12]

Algunas de las ventajas de desarrollar software basado en componentes son las que se comentan a continuación: [13]

- ✓ **Reutilización de software:** Nos lleva a alcanzar un mayor nivel de reutilización.

- ✓ **Simplifica las pruebas:** Permite que las pruebas sean ejecutadas probando cada uno de los componentes antes de probar el conjunto completo de componentes ensamblados.

- ✓ **Simplifica el mantenimiento del sistema:** Cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar las otras partes del sistema.

- ✓ **Mayor calidad:** Dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará continuamente.

#### 1.6.2.1.1 Tecnología Servicios Web XML

La comunicación hacia y desde un servicio, es realizada utilizando mensajes y no llamadas a métodos, como se realiza en el paradigma de la Programación Orientada a Objetos. Estos mensajes deben contener o referenciar toda la información necesaria para ser comprendidos internamente por la aplicación que lo ha solicitado. Un Servicio Web es un sistema de software diseñado para soportar interacción máquina a máquina los cuales pueden ser descritos, publicados, localizados e invocados a través de la red utilizando protocolos estándar.

Los Servicios Web utilizan SOAP (Simple Object Access Protocol) como protocolo para invocar llamadas remotas por su simplicidad, se puede identificar un mensaje SOAP como un documento XML (eXtensible Markup Language) conformado por una envoltura "envolpe" obligatoria, un encabezamiento "header", opcional y un cuerpo ""body" también obligatorio. SOAP permite la comunicación entre aplicaciones

heterogéneas, de modo que clientes de diferentes plataformas o lenguajes de programación pueden comunicarse entre sí de manera satisfactoria.

Paralelamente, alrededor de los Servicios Web existen una serie de protocolos y mecanismos adicionales para facilitar tareas como el descubrimiento de servicios distribuidos a lo largo de la red o UDDI (Universal Description, Discovery and Integration of Servicio Web), una descripción del contenido de los mensajes o WSLD (Servicio Web Description Language) el cual define donde está disponible el servicio y qué protocolo de comunicaciones utilizar para consumir este, lo que significa que un archivo WSDL define todo lo necesario para escribir un programa que interactúe con un Servicio Web. Además se utiliza el protocolo HTTP (Hypertext Transfer Protocol) para el transporte de la mensajería, fundamentalmente utilizando el puerto 80 ya que el mismo siempre se encuentra accesible.

### 1.6.2.2 Alta cohesión y bajo acoplamiento

La alta cohesión significa que la información que gestione un servicio determinado, debe ser coherente y estar en la mayor medida de lo posible relacionada con la información proporcionada por este. Cada elemento de nuestro diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos. Un ejemplo de baja cohesión son servicios que realizan demasiadas tareas. En todas las metodologías de desarrollo se considera la factorización, proceso que permite la creación de los denominados paquetes de servicio. El bajo acoplamiento es la idea de tener los componentes y los servicios que estos brindan lo menos ligados entre si, de tal forma que en caso de producirse una modificación en alguno, tenga la mínima repercusión en el resto de los servicios que conforman el componente, potenciándose así la reutilización y disminuyéndose la interdependencia. [14]

### 1.6.2.3 Arquitectura de 3 capas

La arquitectura de tres niveles es la generalización de la arquitectura Cliente-Servidor, donde la carga se divide en tres partes con un reparto claro de funciones: una Capa de Presentación, otra parte para las reglas lógicas del negocio, denominada Capa de Negocio y la Capa de Datos para el almacenamiento de los mismos. Es decir, está soportado sobre un nivel de abstracción creciente, lo cual permite a los

desarrolladores la fragmentación de un problema complejo en una secuencia de pasos incrementales. También proporciona una amplia reutilización, pues los datos abstractos pueden ser utilizados por diferentes implementaciones o versiones de una misma capa en la medida que soporten las mismas interfaces. [15]

Las capas de presentación (clientes), de negocio y datos pueden residir en un único servidor, aunque lo más común es que haya una multitud de servidores, donde resida este modelo de arquitectura en dependencia de la complejidad de estas capas. Por ejemplo las capas de negocio y datos pueden coexistir simultáneamente en un mismo servidor, ahora se debe tener en cuenta un grupo de elementos que de presentarse, obligarían a realizar la separación de estas; uno de ellos está dado por el volumen y tamaño de la base de datos, considerándose que la misma pueda seguir creciendo en el tiempo una vez generalizado el sistema informático. Por el contrario si la complejidad fuese en el negocio este es subdividido, realizando peticiones sobre una misma base de datos. Este modelo es el que se encuentra implementado para el Registro Informatizado de Salud y será generalizado durante el proceso de desarrollo del Sistema de Información para la Salud. (Ver Anexo III)

### 1.6.2.4 Middleware

Es una capa de software intermedio que se encuentra entre el cliente y el servidor, encargada de gestionar la conexión entre aplicaciones separadas, además proveen las interfaces de servidor y los protocolos estándares de comunicación para enlazar estas. Existe una gran variedad de paquetes middleware, pero todos ellos tienen en común la capacidad de ocultar la complejidad y diferencias entre los protocolos de red y sistemas operativos, además se basan normalmente en los mecanismos básicos de mensajería o llamada a procedimientos remotos (RPC).

Es interesante considerar el papel del middleware desde un punto de vista lógico más que desde la implementación. Esta capa permite cumplir los requerimientos del proceso distribuido, viéndose este como un conjunto de aplicaciones y recursos disponibles para los usuarios, no teniendo estos que preocuparse por la ubicación física de los datos y aplicaciones, pues el middleware es el responsable de encaminar las peticiones desde los clientes al servidor.

### 1.6.2.5 Modelo Vista Controlador (MVC)

Este patrón de arquitectura de software permite separar los datos de una aplicación, la interfaz de usuario y la lógica de negocio en tres componentes distintos, esto proporciona múltiples vistas sobre un mismo modelo de datos. El patrón MVC se usa frecuentemente en aplicaciones Web donde se utilicen diferentes interfaces de usuario y el código que provee los datos a la página es dinámico. Los tres elementos esenciales de este patrón son los siguientes: [16]

- ✓ **Modelo:** Administra el comportamiento y los datos del dominio de la aplicación, responde a requerimientos de información sobre su estado, usualmente formulados desde la vista, respondiendo a instrucciones de cambio para cambiar el estado de estos datos, habitualmente desde el controlador.

- ✓ **Vista:** Este presenta el modelo en un formato adecuado para interactuar, usualmente un elemento de interfaz de usuario.

- ✓ **Controlador:** Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

Tanto la vista como el controlador dependen del modelo, el cual no depende de otros conceptos o clases. Esta separación permite construir y probar el modelo independientemente de la representación visual. En aplicaciones Web por ejemplo, la separación entre la vista (navegador) y el controlador (componentes del lado del servidor que manejan los requerimientos a través de HTTP) está muy claramente definida. Entre las ventajas del estilo señaladas por Microsoft 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 datos de manera simultánea.

- ✓ **Adaptación al cambio:** Los requerimientos no funcionales de interfaz de usuario tienden a cambiar con mayor rapidez que las reglas del negocio. Los clientes pueden preferir distintas opciones de representación pero dado que el modelo no depende de la vista, agregar nuevas opciones o modificar las ya existentes generalmente no afecta al modelo.

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 pueden llegar a ser difíciles 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.

### 1.6.2.6 Single Sign On (SSO)

Los usuarios para identificarse ante varios sistemas se ven obligados a recordar numerosas contraseñas y enfrentarse a distintas interfaces, esta es la razón por la que eligen contraseñas sencillas poniendo potencialmente en riesgo la seguridad del sistema. El Enterprise Single Sing On es un servicio que permite el mapeo de credenciales de seguridad entre Windows y otros sistemas heterogéneos, permitiendo que los usuarios puedan acceder a diferentes aplicaciones con un sólo conjunto de credenciales, principalmente es utilizado en escenarios distribuidos. La autenticación se puede desarrollar en un ámbito amplio, fuera del dominio de seguridad particular, mientras que las credenciales de seguridad permanecen dentro de un dominio de seguridad particular.

Esto es un elemento de mucha importancia a tener en cuenta en el proceso de desarrollo del Centro de Control, pues permitirá el encapsulamiento de la infraestructura de seguridad subyacente, trayendo consigo además que los procesos de implementación, despliegue y mantenimiento sean más fáciles ya que ninguna de las partes comunicantes en el sistema distribuido, necesita implementar individualmente todos los mecanismos de seguridad. Este tipo de proceso permitirá a los desarrolladores y organizaciones

centrarse en el desarrollo de la lógica de negocio asociada a un componente en particular, obviando los mecanismos de seguridad, debido a que en el momento de ser registrados en SISalud a través del Centro de Control estos serán proporcionados de manera automática.

### 1.6.2.7 Observador – Observable

Este patrón es muy utilizado en la implementación de una arquitectura MVC, el modelo es un subtipo de Observable y la vista un subtipo de Observador. Normalmente se implementan como dos clases que manejan adecuadamente la función de notificación de cambios que necesita MVC, pues proporciona el mecanismo por el cual las vistas pueden ser notificadas automáticamente de los cambios producidos en el modelo. Permite gestionar la relación entre un componente observado y sus observadores, pudiendo un componente determinado comportarse de ambas formas, es decir cada componente observado posee una referencia a todos sus observadores, así como los servicios que están siendo consumidos por estos.

El patrón de diseño Observador – Observable es aplicable cuando: [17]

- ✓ Existe una relación fuerte entre datos y vistas, de manera que conviene separar el control de los datos de su representación final.
- ✓ Un cambio en el estado de un determinado componente afecte a muchos otros.
- ✓ Se necesite notificar a otros componentes sin hacer presunciones sobre la identidad y ubicación física de estos, evitando un fuerte acoplamiento lo cual ayuda a diseñar sistemas con diferentes capas de abstracción claramente separadas.
- ✓ Se necesitan sincronizar las acciones de varios componentes sobre un estado común y la coherencia de ellos.

Como se ha comentado en otras secciones, las aplicaciones destinadas a salud pública deben proporcionar una vista única para los datos que se gestionan, ya sea desde componentes centralizados o

distribuidos, pero se deben garantizar los principios intrínsecos a todo sistema informático, entiéndase esto como la integridad y confidencialidad de la información que sea gestionada, en el caso que ocupa a SISalud, la asociada con los servicios de salud que se brindan a una población determinada. Es por ello, que uno de los objetivos bien marcados de la nueva versión del componente de Seguridad SAAA, es garantizar la integridad referencial en el flujo de información que sea gestionada por los componentes integrados a SISalud.

### 1.6.3 Plataforma de Servicios PlaSer

PlaSer es una librería para el desarrollo de componentes utilizando el paradigma de los Servicios Web basados en XML. Esta fue creada para facilitar y agilizar el proceso de implementación y la homogeneidad de los módulos del RIS. La versión actual de PlaSer sólo soporta como llamada RPC el protocolo SOAP. Está implementada completamente con PHP y entre los componentes que la conforman pueden ser mencionados: Plaser_Client (Manipula las llamadas mediante SOAP), Plaser_XML (Manipula el documento XML que devuelve el Plaser_Client e interconecta este con Plaser_XSLT), Plaser_XSLT (Transforma el XML hacia HTML aplicando para ello el XSLT correspondiente), Plaser_DBz (Manipula las operaciones con la Base de Datos). Este grupo de componentes será utilizado para el desarrollo del Centro de Control. (Ver Anexos V)

### 1.6.4 Lenguajes utilizados en el proceso de desarrollo

En esta sección se realizará un análisis de los lenguajes que se utilizan para el proceso de implementación de la Capa de Negocio y Presentación del Componente Centro de Control. En todos los casos se abordará la definición de estos y se expondrán algunas de sus características, así como las ventajas que influyeron en su elección.

### 1.6.4.1 JavaScript

Es un lenguaje de programación interpretado, es decir no requiere compilación. El código JavaScript es embebido directamente en el código HTML o similares, haciendo fácil la creación de páginas Web con contenido dinámico. Está diseñado para controlar la apariencia y manipular los eventos de los objetos contenidos en un formulario, además de ser soportado por la mayoría de los navegadores existentes. El mismo comparte muchos elementos con otros lenguajes de alto nivel siendo muy semejante a C, Java o PHP, tanto en su formato como en su sintaxis. También se debe tener en cuenta que los programas escritos en este lenguaje tienden a ser pequeños y compactos, lo cual influye en que no se requiera ni memoria ni tiempo adicional para su transmisión. [18]

### 1.6.4.2 PHP

PHP (Hypertext Pre-processor) es un lenguaje de programación interpretado, creado en 1994 por Rasmus Lerdof. Es utilizado habitualmente para la creación de sitios, contenido dinámico para aplicaciones Web y aplicaciones para servidores. Con frecuencia los scripts PHP se embeben en otros códigos como HTML ampliando las posibilidades del diseñador de páginas Web. La interpretación y ejecución de estos scripts se hacen en el servidor, el cliente (solicitud realizada desde un navegador Web) sólo recibe el resultado y jamás ve el código PHP. Permite conexión con todo tipo de bases de datos como MySQL, Postgre SQL, Oracle, DB2, Microsoft SQL Server, Firebird y SQLite. PHP corre sobre 7 plataformas, funciona en 11 tipos de servidores, ofrece soporte para varios Sistemas Gestores de Bases de Datos (SGBD) y contiene unas 40 extensiones estables, actualmente se encuentra en la versión 5.

Algunas de las más importantes capacidades de PHP son: [19]

- ✓ Integración con varias bibliotecas externas, permitiendo generar documentos Portable Document Format (PDF) y Microsoft Office Excel (XLS).

- ✓ Ofrece una solución simple y universal para las paginaciones dinámicas de fácil programación.

- ✓ Soportado por una gran comunidad de desarrolladores, como producto de código abierto, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente, implicando menos costos.

- ✓ Gran número de funciones predefinidas. A diferencia de otros lenguajes de programación, PHP fue diseñado especialmente para el desarrollo de páginas Web dinámicas. Por ello, está dotado de un gran número de funciones que simplificará enormemente tareas habituales como descargar documentos, envío de correo electrónico, creación dinámica de imágenes y gráficos en el servidor, procesamiento de información en formularios, manipulación de cookies y sesiones, transporte de información mediante HTTP y análisis de documentos XML.

- ✓ Análisis léxico para reconocer el tipo de dato almacenado en una variable haciéndose automáticamente, permitiéndole al usuario no tener que separar las variables de sus valores.

- ✓ Posee un conjunto de funciones de seguridad que previenen la inserción de órdenes dentro de una solicitud de datos desde el cliente evitando por ejemplo, la ocurrencia de la conocida inyección de código SQL.

Debe tenerse en cuenta que este potente lenguaje de programación también posee algunas desventajas entre ellas se pueden mencionar que todo el trabajo se realiza en el lado del servidor, no delegando responsabilidades al cliente, puede que en un momento determinado la capacidad de respuesta sea ineficiente en la medida en que las solicitudes al servidor aumenten considerablemente. Además si el código PHP se incluye en código HTML, puede que la legibilidad de este se vea afectada. La Programación Orientada a Objetos (POO) en PHP es aún deficiente para aplicaciones de gran envergadura influyendo sobre el rendimiento de estos sistemas. Por tal motivo la versión 4.3.4 será la utilizada para el proceso de implementación del Centro de Control.

### 1.6.4.3 Lenguajes de marcas

El intercambio de información siempre ha sido un grave problema cuando se utilizan lenguajes y sistemas operativos incongruentes y heterogéneos, este problema sólo se agudizó con la aparición de Internet; inclusive previo a su aparición, fueron creados mecanismos para lograr el intercambio fluido de información entre diferentes sistemas, los lenguajes de marcas.

En sentido general, estos son una forma de codificar un documento que, junto con el texto, incorpora marcas que contienen información adicional acerca de la estructura del texto o su presentación. Proporcionan un conjunto de etiquetas o tags, que se entremezclan con el propio contenido en un único archivo o flujo de datos. Se suele diferenciar entre tres clases de lenguajes de marcado, aunque en la práctica pueden combinarse varias clases en un mismo documento siendo estas el marcado de presentación, el marcado procedimental y el marcado descriptivo o semántico, los cuales se comentan a continuación:

- ✓ **Marcado de presentación:** Es aquel que indica el formato del texto. Este tipo de marcado es útil para maquetar la presentación de un documento para su lectura, pero resulta insuficiente para el procesamiento automático de la información. El marcado de presentación resulta más fácil de elaborar, sobre todo para cantidades pequeñas de información. Sin embargo resulta complicado de mantener o modificar, por lo que su uso se ha ido reduciendo en proyectos grandes en favor de otros tipos de marcado más estructurados.

- ✓ **Marcado de procedimientos:** Está enfocado hacia la presentación del texto, sin embargo, también es visible para el usuario que edita el texto. El programa que representa el documento debe interpretar el código en el mismo orden en que aparece. Por ejemplo, para formatear un título, debe haber una serie de directivas inmediatamente antes del texto en cuestión, indicándole al software instrucciones tales como centrar, aumentar el tamaño de la fuente, o cambiar a negrita. Inmediatamente después del título deberá haber etiquetas inversas que reviertan estos efectos. En sistemas más avanzados se utilizan macros o pilas que facilitan el trabajo.

- ✓ **Marcado descriptivo o semántico:** Utiliza etiquetas para describir los fragmentos de texto, pero sin especificar cómo deben ser representados o en qué orden, lo cual le da al mismo una gran flexibilidad. El marcado descriptivo también simplifica la tarea de reformatear un texto, debido a que la información del formato está separada del propio contenido.

#### 1.6.4.3.1 HTML

HTML es el acrónimo de Hypertext Markup Language (Lenguaje de marcado de hipertexto), creado en 1989 por Tim Berners-Lee. Es originalmente un subconjunto de SGML (Standard Generalized Markup Language), especializado en la descripción de documentos en pantalla. El proyecto inicial se basaba en una colección de etiquetas que permitían describir documentos de texto y vínculos de hipertexto que hacían posible el desplazamiento en forma jerárquica entre diferentes documentos. La facilidad de su uso y la particularidad de no ser propiedad de nadie, hizo de HTML el sistema idóneo para compartir información a través de Internet. Inicialmente su intención era que las etiquetas fueran capaces de marcar la información de acuerdo a su significado, pero por diversos motivos los creadores de los navegadores Web fueron añadiendo más etiquetas HTML, dirigidas a controlar la representación de la información contenida en el documento. [20]

#### 1.6.4.3.2 XML

eXtensible Markup Language (XML), establecido en febrero 1998, no es más que un metalenguaje de marcado descriptivo, que posee un grupo de reglas que le permite crear sus propios elementos de marcación, los cuales pueden usarse después para describir su contenido. XML se desarrolló porque HTML no estaba diseñado para describir algunos tipos de datos que las personas querían enviar a través de la red, pues este último tenía sus etiquetas predefinidas. XML proporciona la flexibilidad y consistencia que no se podía alcanzar con HTML, pues no se tiene que forzar al contenido para que se ajuste al grupo limitado de elementos proporcionados por este. En pocas palabras, XML proporciona un mecanismo estándar para describir cualquier tipo de datos en documentos que son altamente transportables y reutilizables. [21]

#### 1.6.4.3.3 XHTML

HTML carece de grandes cantidades de marcas semánticas por lo que dificulta la descripción y el intercambio de datos y la mayoría de los documentos especificados en este lenguaje, en la actualidad, están plegados de código innecesario lo que hace que la velocidad de descarga aumente considerablemente. Desde finales de 1998, se produce una migración hacia tecnologías relacionadas con XML. Su éxito no fue una sorpresa, pues este lenguaje permite a los desarrolladores la capacidad de crear un vocabulario personalizado, a la vez que separa la presentación de la estructura, lo que hace que el mantenimiento y el análisis gramatical sea más fácil.

XHTML es el acrónimo de eXtensible Hypertext Markup Language (Lenguaje de Marcado de Hipertexto Extensible). No es más que una reformulación de HTML 4.01 de acuerdo a las reglas de XML 1.0, en realidad está diseñado para que HTML vuelva a sus raíces y lo convierta en un grupo de marcación que describa una clase en particular de contenido sin importar cómo se visualizará este. Debido a que está especificado de acuerdo a XML, puede beneficiarse de muchas tecnologías y herramientas que están desarrolladas para trabajar con XML. XHTML 1.0 se convirtió en la recomendación de World Wide Web Consortium (W3C) en enero del 2000. Aunque XHTML es compatible regresivamente con los navegadores Web actuales y no es muy diferente a HTML 4.01, muchos desarrolladores todavía se resisten a cambiar porque no ven el valor agregado que ofrece una herramienta con un grupo más estricto de especificaciones de sintaxis. [22]

#### 1.6.4.3.4 XSL y XSLT

Como se analizó en el epígrafe anterior, XHTML podrá de una manera natural beneficiarse con infraestructuras desarrolladas para XML, como por ejemplo XSL (eXtensible Stylesheet o Language Lenguaje de Hojas de Estilo Extensible) y XSLT (XSL Transformations o Transformaciones XSL). XSL es al igual que CSS (Cascading Style Sheets Hojas de estilo en cascada) un lenguaje de hojas de estilo, pero a diferencia de CSS permite realizar transformaciones complejas o simples de documentos, a la vez que les da formato. Es decir, XSL consta de dos etapas, la primera de ellas en un proceso de transformación,

esbozado en la especificación de XSLT y en segundo lugar, un proceso para dar formato, definido en la especificación principal de XSL.

### 1.6.5 Sistemas de Gestión de Bases de Datos

Un Sistema de Gestión o Manejador de Bases de Datos (SGBD) es un conjunto de programas que permite a los usuarios crear y mantener una Base de Datos, por lo tanto, el SGBD es un software de propósito general que facilita el proceso de definir, construir y manipular Bases de Datos para diversas aplicaciones. Existen muchas formas de organizar las bases de datos, pero hay un conjunto de objetivos generales que deben cumplir todos los SGBD, de modo que faciliten el proceso de diseño de aplicaciones y que los tratamientos sean más eficientes y rápidos, dando la mayor flexibilidad posible a los usuarios. Los objetivos fundamentales de los SGBD son:

- ✓ Independencia de los datos y los programas de aplicación.
- ✓ Minimización de la redundancia.
- ✓ Integración y sincronización de Bases de Datos.
- ✓ Integridad de los datos.
- ✓ Seguridad y protección de los datos.
- ✓ Facilidad de manipulación de la información.

#### 1.6.5.1 MySQL

MySQL es uno de los SGBD más populares desarrolladas bajo la filosofía de código abierto. Fue implementado y le brinda soporte la empresa MySql AB, pero puede utilizarse gratuitamente y su código fuente está disponible. Su ingeniosa arquitectura lo hace extremadamente rápido y fácil de personalizar. Posee una extensiva reutilización del código dentro del software que ha dado lugar a un sistema de administración de bases de datos con una importante velocidad de respuesta, compactación, estabilidad y

facilidad de despliegue. Entre las principales características de este gestor de datos se pueden mencionar algunas como por ejemplo, aprovechamiento de la potencia de sistemas multiprocesador, dado por su implementación multihilo; soporta gran cantidad de tipos de datos para las columnas; versatilidad pues trabaja en sistemas operativos basados en UNIX como el Windows de Microsoft, así como que posee una gestión muy eficiente de requerimientos de seguridad. [23]

### 1.6.6 Servidor Web Apache

Comenzó a desarrollarse en 2005 pero hoy en día es el servidor Web más utilizado del mundo, encontrándose muy por encima de todos sus competidores, tanto gratuitos como comerciales. Representa el complemento perfecto para el desarrollo de páginas dinámicas con PHP y MySQL, pues comparte con estos muchas de sus características, como gratuidad, popularidad, su sencillez de manejo y versatilidad, ya que puede ser instalado tanto sobre Linux o sobre Windows, de hecho en marzo del 2007 se utilizaba por el 58.62 % de los sitios activos en el mundo. (Ver Anexo V).

### 1.6.7 Proceso Unificado de Desarrollo (RUP)

Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un software. Sin embargo, RUP es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, tipos de organizaciones, niveles de aptitud y tamaños de proyecto. El Proceso Unificado está basado en componentes, lo cual quiere decir que el sistema o software en construcción está formado por componentes interconectados a través de interfaces. [24]

RUP utiliza el UML (Unified Modeling Language Lenguaje o Unificado de Modelado) para representar todos los esquemas del sistema de software a desarrollar, de hecho UML es una parte esencial de RUP, pero sus verdaderos aspectos definitorios se resumen en tres ideas claves, dirigido por casos de uso, centrado en la arquitectura e iterativo e incremental. Es decir, un sistema de software se define para dar servicios a sus usuarios, por lo tanto para construirlo con éxito se debe conocer qué es lo que los futuros usuarios necesitan así como que para llevar a cabo el proceso de desarrollo es conveniente dividir el

trabajo en partes pequeñas o miniproyectos, representando cada uno de ellos lo que se conoce como iteración.

### 1.6.7.1 Lenguaje Unificado de Modelado (UML)

UML se define como un lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de software. Es un sistema notacional, que entre otras cosas incluye el significado de estas, destinado a los sistemas de modelado que utilizan conceptos orientados a objetos. Nació en 1994 por iniciativa de Grady Booch y Jim Rumbaug.

Es un lenguaje para construir modelos, no guía al desarrollador en la forma de realizar análisis y diseño orientado a objetos ni le indica cuál proceso de desarrollo adoptar, está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. La finalidad de los diagramas es presentar diversas perspectivas de un sistema, a las cuales se les conoce como modelo. [25]

## 1.6.8 Herramientas a utilizar

Con los argumentos dados en los epígrafes anteriores y con un conocimiento del problema en cuestión, se pueden definir las herramientas que se utilizan para el desarrollo del Centro de Control. En primer lugar se utilizara el Rational Rose Enterprise Edition 2003 como herramienta CASE (Computer-Aided Software Engineering) basada en UML, que permite crear los diagramas que se generen como parte de la documentación. Para la creación de las páginas Web, Dreamweaver 8 debido a que soporta el lenguaje de marcas XHTML 1.0, para la edición de código PHP, se utiliza el Zend Studio 5 y para administrar la base de datos MySQL el MySQL-Front, debido a que es un ambiente gratuito de sencilla utilización que posee una interfaz muy intuitiva. Finalmente se escoge el Stylus Studio 5.1 siendo este un completo entorno de desarrollo integrado que incluye un potente editor de XML y un "debugger" XSLT.

## 1.7 Conclusiones

En este capítulo, se profundizó en los conceptos y definiciones necesarias para comprender el proceso de desarrollo del Centro de Control. Este en un sistema de administración y seguridad que se encuentra estrechamente relacionado con la arquitectura definida por el MINSAP para dar cumplimiento al programa general para informatizar el Sistema Nacional de Salud. Además se realizó, un análisis de las tecnologías, patrones, lenguajes y herramientas que se tendrán en cuenta para dar cumplimiento al proceso de desarrollo, que estará basado completamente en software no propietario, obedeciendo a una de las políticas definidas por el MINSAP.

# CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

## 2.1 Introducción

En este capítulo se realiza una descripción de la propuesta de solución para el Centro de Control. Debido a la poca estructuración de los procesos de negocio y para poder comprender el contexto en el cual se desarrolla el sistema se determina desarrollar un Modelo de Dominio, donde se expone un marco conceptual y las relaciones entre estas definiciones. Por otra parte, se enumeran los requerimientos funcionales y no funcionales, agrupándose los primeros en Casos de Uso, con el fin de estructurar el Diagrama de Casos de Uso del Sistema. Asimismo se justifican los usuarios que interactuarán con el Centro de Control.

## 2.2 Modelo de Dominio

El Modelo de Dominio o Modelo Conceptual, permite de manera visual mostrar al usuario los principales conceptos que se manejan en el dominio del sistema en desarrollo. Un Modelo del Dominio es una representación de las clases conceptuales del mundo real, no de componentes software. El modelo desarrollado no se trata de un conjunto de diagramas que describen clases de software u objetos de software con responsabilidades, sino que puede considerarse como un diccionario visual de las abstracciones relevantes, vocabulario e información del dominio. Aprovechando las bondades de los diagramas UML para representar conceptos, el Modelo del Dominio se presenta en forma de diagrama de clases donde figuran los principales conceptos y roles del sistema en cuestión.

En el desarrollo del Centro de Control, se decide realizar un Modelo de Dominio debido a que no pueden ser identificados los procesos de negocio asociados al contexto del sistema, como consecuencia de la poca estructuración y fronteras de los mismos. Además las definiciones o conceptos que se interrelacionan fueron obtenidos sobre la base del conocimiento de expertos de la Empresa SOFTEL y de la experiencia asociada a la versión actual del componente SAAA, desarrollada para el RIS en diciembre del año 2003 por estos mismos expertos. También fue recomendable realizar un Modelo de Dominio

debido que los conceptos que serán expuestos representan objetos del mundo real a los cuales el sistema debe darle un estricto seguimiento.

### 2.2.1 Conceptos fundamentales

Para una mejor comprensión del Diagrama del Modelo de Dominio estructurado en el siguiente epígrafe, a continuación se proporciona un marco conceptual con las definiciones identificadas en el contexto del proceso de desarrollo del Centro de Control, estas son:

**Administrador de Configuración:** Único encargado de la gestión de componentes, tipos de usuarios, métodos y módulos. Además se encarga de la gestión de la integridad referencial entre los componentes.

**Administrador General:** Usuario que en dependencia de su nivel (Nacional, Provincial, Municipal, Unidades de Salud) tiene los permisos para gestionar la información de los usuarios en su mismo nivel, así como de los administradores del nivel inmediato inferior en su misma ubicación y unidades de salud subordinadas a su nivel. Es el encargado de realizar auditorías sobre la trazabilidad de estos usuarios.

**Administrador Unidad de Salud:** Es la persona encargada de realizar la gestión de los usuarios en el Nivel de Unidades de Salud, pudiendo realizar labores de auditoría sobre las operaciones en las que intervienen los usuarios creados por él. Puede ser el Vicedirector Primero u otra persona designada por el Director de la unidad.

**Administrador Municipal:** Es la persona encargada de realizar la gestión de los usuarios en el Nivel Municipal, así como de los administradores de Unidades de Salud subordinadas a su municipio, pudiendo realizar labores de auditoría sobre las operaciones en las que intervienen estos usuarios. Puede ser el Vicedirector Primero u otra persona designada por el Director municipal.

**Administrador Provincial:** Es la persona encargada de realizar la gestión de los usuarios en el Nivel Provincial, así como de los administradores municipales y administradores de Unidades de Salud subordinadas a esta provincia, pudiendo realizar labores de auditoría sobre las operaciones en las que

intervienen estos usuarios. Puede ser el Vicedirector Primero u otra persona designada por el Director Provincial.

**Administrador Nacional:** Es la persona encargada de realizar la gestión de los usuarios en el Nivel Nacional, así como de los administradores provinciales y administradores de Unidades de Salud de subordinación nacional, pudiendo realizar labores de auditoría sobre las operaciones en las que intervienen estos usuarios. Es el Director Nacional de Informática del MINSAP.

**Certificado:** Conjunto de datos y privilegios de un usuario determinado que se crea durante los procesos de autenticación y autorización, el mismo contendrá un identificador único (token) de 32 caracteres que se genera de manera aleatoria, tiene el nivel de acceso del usuario, el identificador del nivel de acceso y un listado de los módulos a los que tiene acceso el usuario y los privilegios que tiene el mismo en estos módulos.

**Componente:** Es un subsistema que representa el núcleo de la aplicación donde se encuentran los métodos y provee los servicios que pueden ser consumidos por otros componentes.

**Nivel:** Niveles de dirección administrativa del Sistema Nacional de Salud cubano siendo estos Nacional, Provincial, Municipal o Unidad de Salud.

**Método:** Secuencia de operaciones lógicas que pueden modificar el estado de una información o simplemente acceder a ella.

**Módulo:** Conjunto de componentes que se integran para mostrar los datos gestionados por ellos de forma unificada en una misma presentación.

**Traza:** Historial que guarda los eventos realizados por el usuario al interactuar con la información perteneciente al sistema.

**Tipo de usuario:** Clasificación dada a los usuarios de un determinado componente en dependencia del negocio gestionado por este.

**Usuario:** Personal que tiene acceso a los diferentes módulos que forman parte del Sistema de Información para la Salud.

## 2.2.2 Diagrama del Modelo de Dominio

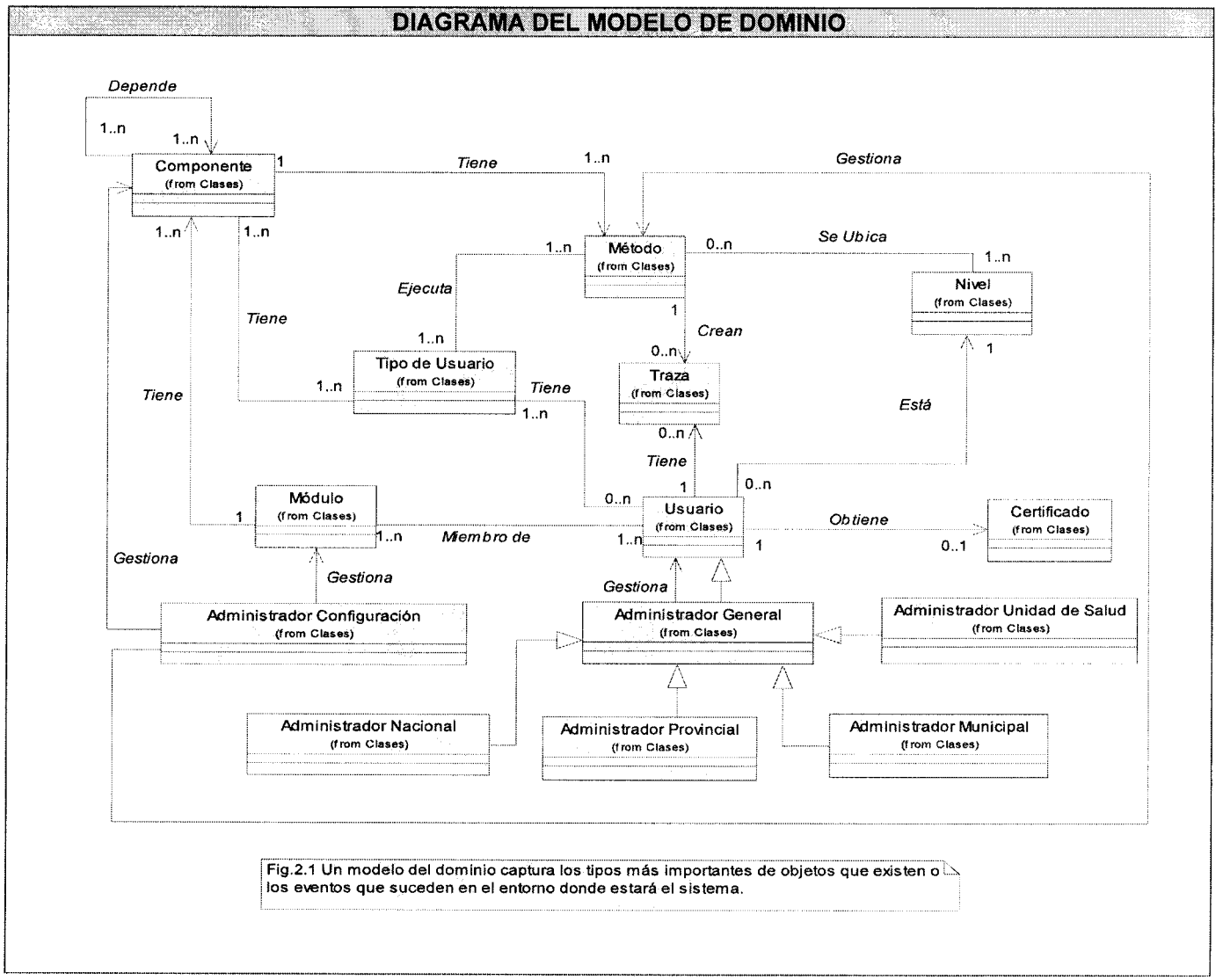



Figura 2.1 Diagrama del Modelo de Dominio

## 2.3 Propuesta de Sistema

Un sistema está comprendido por varios elementos: hardware, software y personas; la ingeniería de sistemas ayuda a traducir las necesidades del cliente en un modelo de sistema que utiliza estos elementos. Este proceso comienza tomando la visión global definida en el Modelo de Dominio realizándose un análisis del mismo con el fin de establecer todos los requerimientos básicos, cada uno de los conceptos o definiciones identificados serán objeto de estudio individualmente. La ingeniería de sistemas demanda una intensa comunicación entre el cliente y los desarrolladores, realizándose un grupo de actividades denominadas ingeniería de requisitos. Una vez que los requisitos hayan sido identificados el Modelo del Sistema puede ser realizado.

### 2.3.1 Especificación de los Requerimientos de Software

El propósito fundamental del proceso de especificación de requerimientos de software es guiar su desarrollo correctamente. Esto se consigue, mediante una descripción de los requisitos del sistema suficientemente buena para que pueda llegarse a un acuerdo entre el cliente, incluyendo a los usuarios finales y los desarrolladores sobre lo que debe y no hacer el sistema. El esfuerzo principal en el flujo de requerimientos, es desarrollar un Modelo de Sistema que se va a construir y la agrupación correcta de estos en casos de uso para estructurar este modelo. Según la IEEE Standard Glossary of Software Engineering Terminology, los requerimientos de software, son condiciones o capacidades que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente.

#### 2.3.1.1 Requerimientos Funcionales

Los requerimientos funcionales son condiciones o capacidades que el sistema debe cumplir. Son las acciones que el sistema debe permitir realizar para que se cumplan los objetivos y de esta manera dar solución a los problemas identificados. Estas acciones son ejecutadas por los usuarios o el sistema internamente las realiza. Una vez analizados los conceptos descritos en el Modelo de Dominio se obtuvo

el siguiente levantamiento de requerimientos funcionales los cuales han sido enumerados en orden de prioridad:

| LISTADO DE REQUERIMIENTOS FUNCIONALES | |
|---|---|
| **RF1-** Autenticar | **RF20-** Buscar Usuario Externo |
| **RF2-** Autorizar | **RF21-** Buscar Componentes |
| **RF3-** Crear certificado digital | **RF22-** Buscar Métodos |
| **RF4-** Obtener Privilegios | **RF23-** Buscar Usuario |
| **RF5-** Crear Trazas | **RF24-** Cambiar Contraseña |
| **RF6-** Gestionar Componentes | **RF25-** Listar Componentes |
| **RF7-** Gestionar Tipo de Usuario | **RF26-** Listar Métodos |
| **RF8-** Gestionar Métodos | **RF27-** Listar Tipo de Usuario |
| **RF9-** Asignar Tipo de Usuario | **RF28-** Listar Niveles |
| **RF10-** Asignar Módulo | **RF29-** Listar Módulos |
| **RF11-** Asignar Nivel de Acceso | **RF30-** Buscar Trazas |
| **RF12-** Asignar Ubicación | **RF31-** Exportar Listado Usuarios |
| **RF13-** Asignar Usuario Externo | **RF32-** Exportar Listado Trazas |
| **RF14-** Gestionar Módulo | **RF33-** Enviar Confirmación Automática |
| **RF15-** Asignar Componente | **RF34-** Buscar Unidades de Salud |
| **RF16-** Asignar Componente Base | **RF35-** Listar Ubicaciones |
| **RF17-** Asignar Método | **RF36-** Listar Configuración Total |
| **RF18-**Gestionar Dependencia entre Componentes | **RF37-** Exportar Listado Módulos |
| **RF19-** Gestionar Usuarios | **RF38-** Generar Ficheros de Configuración |

Tabla 2.1 Listado de Requerimientos Funcionales

Seguidamente los requerimientos funcionales serán agrupados en casos de uso del sistema. Téngase en cuenta que un caso de uso es la forma en que los actores usan el sistema. Es decir son fragmentos de funcionalidades que el sistema ofrece para aportar un resultado de valor a sus actores o una secuencia de acciones que el sistema puede llevar a cabo interactuando con estos, incluyendo alternativas dentro de la secuencia, en tal sentido los casos de uso del sistema identificados son los siguientes:

| DEFINICIÓN DE CASOS DE USO | |
|---|---|
| **CUS-1: Autenticar**<br>**RF1-** Autenticar<br>**RF2-** Autorizar<br>**RF3-** Crear certificado digital<br>**RF4-** Obtener Privilegios<br>**RF5-** Crear Trazas | **CUS-6: Gestionar_Componentes**<br>**RF6-** Gestionar Componentes<br>Insertar Componente<br>Modificar Componente<br>Eliminar Componente<br>**RF15-** Asignar Componente<br>**RF17-** Asignar Método<br>**RF18-** Gestionar Dependencia entre Componentes<br>Insertar Dependencia entre Componentes<br>Eliminar Dependencia entre Componentes<br>**RF21-** Buscar Componentes<br>**RF25-** Listar Componentes<br>**RF26-** Listar Métodos<br>**RF38-** Generar Ficheros de Configuración |
| **CUS-2: Modificar_Perfil**<br>**RF24-** Cambiar Contraseña | |
| **CUS-3: Gestionar_Usuarios**<br>**RF9-** Asignar Tipo de Usuario<br>**RF12-** Asignar Ubicación<br>**RF13-** Asignar Usuario Externo<br>**RF19-** Gestionar los Usuarios<br>Insertar Usuarios<br>Modificar Usuarios<br>Eliminar Usuarios<br>**RF21-** Buscar Usuario Externo<br>**RF23-** Buscar Usuario<br>**RF27-** Listar Tipo de Usuario<br>**RF29-** Listar Módulos<br>**RF31-** Exportar Listado Usuarios<br>**RF33-** Enviar Confirmación Automática<br>**RF34-** Buscar Unidades de Salud<br>**RF35-** Listar Ubicaciones | **CUS-7: Gestionar_Métodos**<br>**RF8-** Gestionar Métodos<br>Insertar Métodos<br>Modificar Métodos<br>Eliminar Métodos<br>**RF11-** Asignar Nivel de Acceso<br>**RF15-** Asignar Componente<br>**RF22-**Buscar Métodos<br>**RF25-** Listar Componentes<br>**RF26-** Listar Métodos<br>**RF27-** Listar Tipo de Usuario |

| | |
|---|---|
| **CUS-4: Obtener_Trazas**<br>**RF23-** Listar Componentes<br>**RF26-** Listar Métodos<br>**RF30-** Buscar Trazas<br>**RF32-** Exportar Listado Trazas | **CUS-8: Gestionar_Módulos**<br>**RF11-** Asignar Nivel de Acceso<br>**RF14-** Gestionar Módulo<br>Insertar Módulo<br>Modificar Módulo<br>Eliminar Módulo<br>**RF15-** Asignar Componente<br>**RF16-** Asignar Componente Base<br>**RF25-** Listar Componentes<br>**RF29-** Listar Módulos<br>**RF36-** Listar Configuración Total<br>**RF37-** Exportar Listado Módulos<br>**RF38-** Generar Ficheros de Configuración |
| **CUS-5: Gestionar_Tipo_Usuario**<br>**RF7-** Gestionar Tipo de Usuario<br>Insertar Tipo de Usuario<br>Modificar Tipo de Usuario<br>Eliminar Tipo de Usuario<br>**RF27-** Listar Tipo de Usuario | |

Tabla 2.2 Definición de Casos de Uso del Sistema

### 2.3.1.2 Requerimientos No Funcionales

Los requerimientos no funcionales son propiedades o cualidades que hacen al producto atractivo, usable, rápido o confiable. En muchos casos los requerimientos no funcionales influyen considerablemente en el éxito del producto. Su importancia radica en que tanto clientes como usuarios pueden valorar las características no funcionales del producto. Pues si se conoce que el mismo cumple con la toda la funcionalidad requerida, las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable es, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

#### 2.3.1.2.1 Usabilidad

**RNF1:** El sistema debe garantizar un acceso fácil y rápido, podrá ser usado por cualquier usuario que posea pocos conocimientos informáticos y de un ambiente Web en sentido general.

#### 2.3.1.2.2 Rendimiento

**RNF2:** El sistema debe tener una similitud en sus páginas y estar poco cargado con imágenes, posibilitando que las respuestas sean devueltas en un tiempo mínimo, siendo más sencillo de entender y utilizar por el usuario.

#### 2.3.1.2.3 Soporte

**RNF3:** Una vez terminado el desarrollo del Centro de Control se llevarán a cabo los procesos de despliegue, capacitación y mantenimiento software con motivos de asistir a los clientes del producto, así como lograr su mejoramiento progresivo, evolución y sostenibilidad en el tiempo. Además se contará con un grupo de trabajo designado para el mantenimiento y actualización continua del sistema.

#### 2.3.1.2.4 Portabilidad

**RNF4:** Permitir que el sistema se ejecute sobre el Sistema Operativo Linux, Windows 98 o superior.

#### 2.3.1.2.5 Seguridad

**RNF5:** Disponer de un mecanismo de seguridad basado en el modelo de Autenticación, Autorización y Auditoría (AAA).

**Confiabilidad:** La información manejada por el sistema está protegida de acceso no autorizado. El sistema debe prevenir posibles fallos y/o errores y presentar facilidades para una rápida recuperación en dichos casos.

**Integridad:** Que la información sea modificada (incluyendo su creación y borrado) sólo por personal autorizado. Se implementarán políticas de resguardo de información, así como la realización de copias periódicas de seguridad.

**Disponibilidad:** Los usuarios autorizados tendrán acceso a la información en todo momento.

**RNF5.1:** La autenticación será la primera acción del usuario en el sistema y consiste en suministrar un nombre único de usuario y una contraseña que debe ser de conocimiento exclusivo de la persona que se está involucrada en esta operación.

**RNF5.2:** Cada petición de usuario autorizada o no será registrada, almacenándose día, mes, año, hora, minuto y segundo en que fue solicitada dicha petición.

**RNF5.3:** Una persona solo podrá realizar labores en un nivel de dirección, en caso de que por necesidades suceda esto deberá tener una cuenta de usuario con los privilegios asignados, para cada nivel.

**RNF5.4:** Si un usuario realiza laborares de gestión en diferentes niveles de dirección tendrá una cuenta para cada nivel.

**RNF5.5:** Los usuarios de administración sólo tendrán estos privilegios, por lo que no podrán acceder a la información de salud gestionada por SISalud.

**RNF5.6:** La administración del sistema será jerárquica, donde un administrador podrá crear usuarios en su mismo nivel y administradores del nivel inmediato inferior o de unidades de salud subordinadas a su nivel, para cada uno de los módulos y con permisos diferenciados.

**RNF5.7:** El sistema controlará la cantidad de caracteres mínimo y máximo para el login y contraseña, registro de las cuentas creadas por cada administrador y no permitir conexiones concurrentes con un mismo conjunto de credenciales.

**RNF5.8:** Se utilizará el protocolo HTTPS para el transporte de información a través de la Web.

**RNF5.9:** Aplicar políticas de seguridad y acceso a los servidores sea desplegado el Centro de Control.

#### 2.3.1.2.6 Apariencia o Interfaz Externa

**RNF6:** La interfaz debe ser sencilla y amigable ya que el usuario no es experto en el uso de aplicaciones Web. Diseño sencillo, con pocas entradas, permitiendo que no sea necesario mucho entrenamiento para utilizar el sistema, además que exista paginación de reportes de búsqueda y listados. Se utilizan los mismos colores definidos para los módulos desplegados actualmente en el RIS.

#### 2.3.1.2.7 Ayuda y documentación en línea

**RNF7:** Se contará con una Ayuda en línea así como un Manual de Usuario en formato PDF que indicará como interactuar con las funcionalidades del sistema. Además se tendrá leyendas en aquellas páginas que lo requieran, por la utilización de conceptos o abreviaturas desconocidas para el usuario.

#### 2.3.1.2.8 Software

**RNF8.1 Para clientes:** Se tendrá acceso al Centro de Control a través de cualquier navegador Web. Recomendados Firefox Mozilla 1.5, Internet Explorer 5.0 o superior que soporte DHTML y CSS2.

**RNF8.2 Para Capa de Presentación:** El servidor debe tener PHP Versión 4.3.4 (aunque debe trabajar también con el 4.3.2), Biblioteca PEAR-SOAP 0.8RC3, Módulo XSLT Sablotrón en PHP. Servidor HTTP preferiblemente Apache. Sistema Operativo LINUX distribución White Box.

**RNF8.3 Para Capa de Negocio:** El servidor debe tener PHP Versión 4.3.4 (aunque debe trabajar también con el 4.3.2), Módulo DBX en PHP, Biblioteca PEAR-SOAP 0.8RC3. Servidor HTTP preferiblemente Apache. Sistema Operativo LINUX distribución White Box.

**RNF8.3 Para Capa de Datos:** Servidor de Base de Datos MySQL Versión 4. Sistema Operativo LINUX distribución White Box.

#### 2.3.1.2.9 Hardware

**RNF9.1 Requerimientos mínimos para clientes:**

- Ordenador Pentium o superior.
- 64 MB de Memoria RAM.
- Monitor VGA o superior.
- Teclado y Mouse.
- Procesador 486DX / 66 MHZ o superior.
- Disco duro de 20 GB.
- Impresora de puntos.
- Insumos. (Disquetes, CD-RW, Papel continuo y cintas de impresora).
- La PC de trabajo debe estar conectada a una Red de Área Local (LAN).
- Conectividad con el nodo central de INFOMED.

**RNF9.2 Requerimientos para servidores:** Desplegar el Centro de Control en el cluster centralizado de servidores de INFOMED.

Esta propuesta es realizada debido a que el mantenimiento y la actualización del hardware utilizado en estos servidores, son definidos por el Grupo de Arquitectura MINSAP–MIC, en dependencia de los requerimientos de rendimiento y disponibilidad que posean las soluciones informáticas a desplegar en los mismos.

### 2.3.2 Modelo de Casos de Uso del Sistema

El Modelo de Casos de Uso del Sistema permite que los desarrolladores de software y los clientes lleguen a un entendimiento sobre los requisitos, es decir, sobre las condiciones y posibilidades que debe cumplir el sistema. Este modelo proporciona la entrada fundamental para el análisis, diseño e implementación, por lo que mientras mejor estructurado se encuentre el mismo, mucho más sencillo será el proceso de construcción de sistema de software. Este modelo contiene los actores, casos de uso y las relaciones que se establecen entre estos elementos.

### 2.3.2.1 Definición de los actores

El Modelo de Casos de Uso describe, además, lo que hace el sistema para cada tipo de usuario. Estos actores pueden estar representados por un usuario en concreto (un humano u otro sistema) que interactúe de alguna forma con el sistema en desarrollo. A continuación se definen los cinco actores del sistema en desarrollo:

| ACTORES | JUSTIFICACIÓN |
|---|---|
| Administración de Configuración | Actor que tiene permiso total sobre la configuración del sistema. Único encargado de la gestión de componentes, tipos de usuario métodos y módulos. Así como de la gestión de dependencias y restricciones de integridad entre componentes que conforman un módulo determinado. |
| Administrador General | Actor que en dependencia de su nivel (Nacional, Provincial, Municipal o Unidades de Salud), tiene los permisos para gestionar la información de los usuarios que están en su mismo nivel y ubicación, así como de los administradores que pertenecen al nivel inmediato inferior o a unidades subordinadas a su nivel y ubicación. Esta gestión incluye crear usuarios, modificar sus datos y privilegios, eliminación y búsqueda. Por otra parte puede llevar a cabo un nivel de auditoría estricta a través de las trazas del sistema, conociendo qué usuario ha participado en cada transacción. |
| Usuario | Trabajadores de las instituciones del Ministerio de Salud Pública, los cuales pueden fungir como un determinado tipo de usuario dentro del sistema y corresponder a cualquiera de los niveles de dirección del sector de la salud. |
| Registro de Ciudadanos | Registro Externo que posee de forma única los datos personales de cualquier ciudadano que trabaje en salud y/o reciba sus servicios. Los datos fundamentales contenidos en son los mismos que recoge el Carné de Identidad. Del mismo se obtiene el Nombre, Primer |

| | |
|---|---|
| | los usuarios de SISalud. |
| Registro de Unidades de Salud | Registro Externo que gestiona la información de todas las Unidades de Salud del país, entiéndase como unidad de salud a cualquier centro laboral del MINSAP. De estas se conoce: nombre, código, tipo de unidad, subordinación, etc. El mismo es utilizado para realizar búsquedas de las unidades para posteriormente poder crear los Administradores Generales de las mismas. |
| Registro de Ubicación Geográfica | Registro Externo que gestiona las Provincias, Municipios, Localidades, Calles y Manzanas del país. El mismo es utilizado para listar las provincias y municipios con el fin de crear los Administradores Generales de estas ubicaciones en todo el país. |

Tabla 2.3 Justificación de Actores del Sistema

### 2.3.2.1.1 Vista global de los Actores del Sistema

En el siguiente diagrama, se puede observar como han sido separadas en dos usuarios diferentes, la administración de usuarios y privilegios de la configuración del sistema. En el actual módulo de Administración, el Administrador Nacional debe realizar ambos procesos, cuestión esta que es compleja. En la propuesta de actores del Centro de Control, se propone separar estas labores en actores diferentes, el Administrador General y el Administrador de Configuración. El Administrador General será el encargado de la gestión de usuarios y privilegios el cual se encuentra especializado en administradores de los niveles de dirección del MINSAP, lo que permitirá que sólo puedan manipular la información asociada a su nivel. Ambos actores se generalizan el actor Usuario debido que las personas a las que se le asignen estos roles también constituirán usuarios de SISalud.

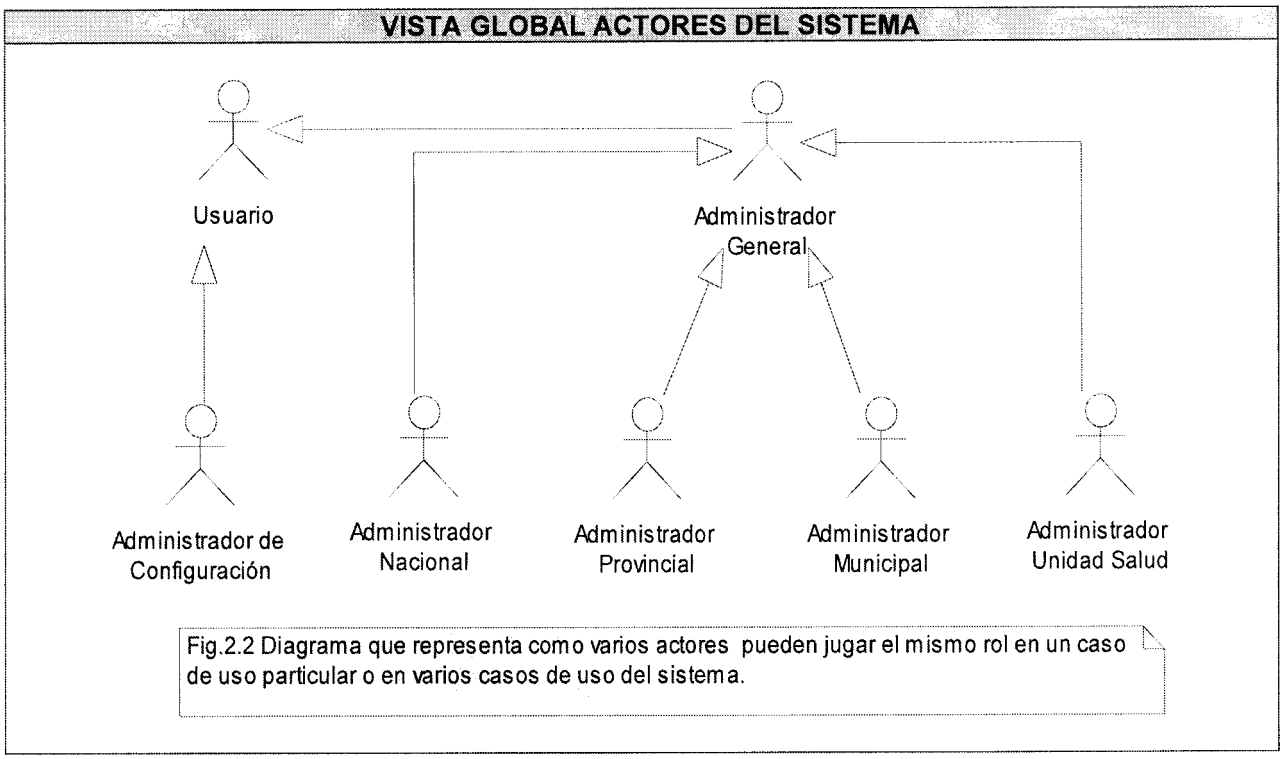



Figura 2.2 Vista Global Actores del Sistema

### 2.3.2.2 Diagrama de Casos de Uso

A continuación se muestra el Diagrama de Casos de Uso observándose las relaciones que se establecen entre actores y casos de uso, es decir son descritas las propiedades estructurales del sistema.

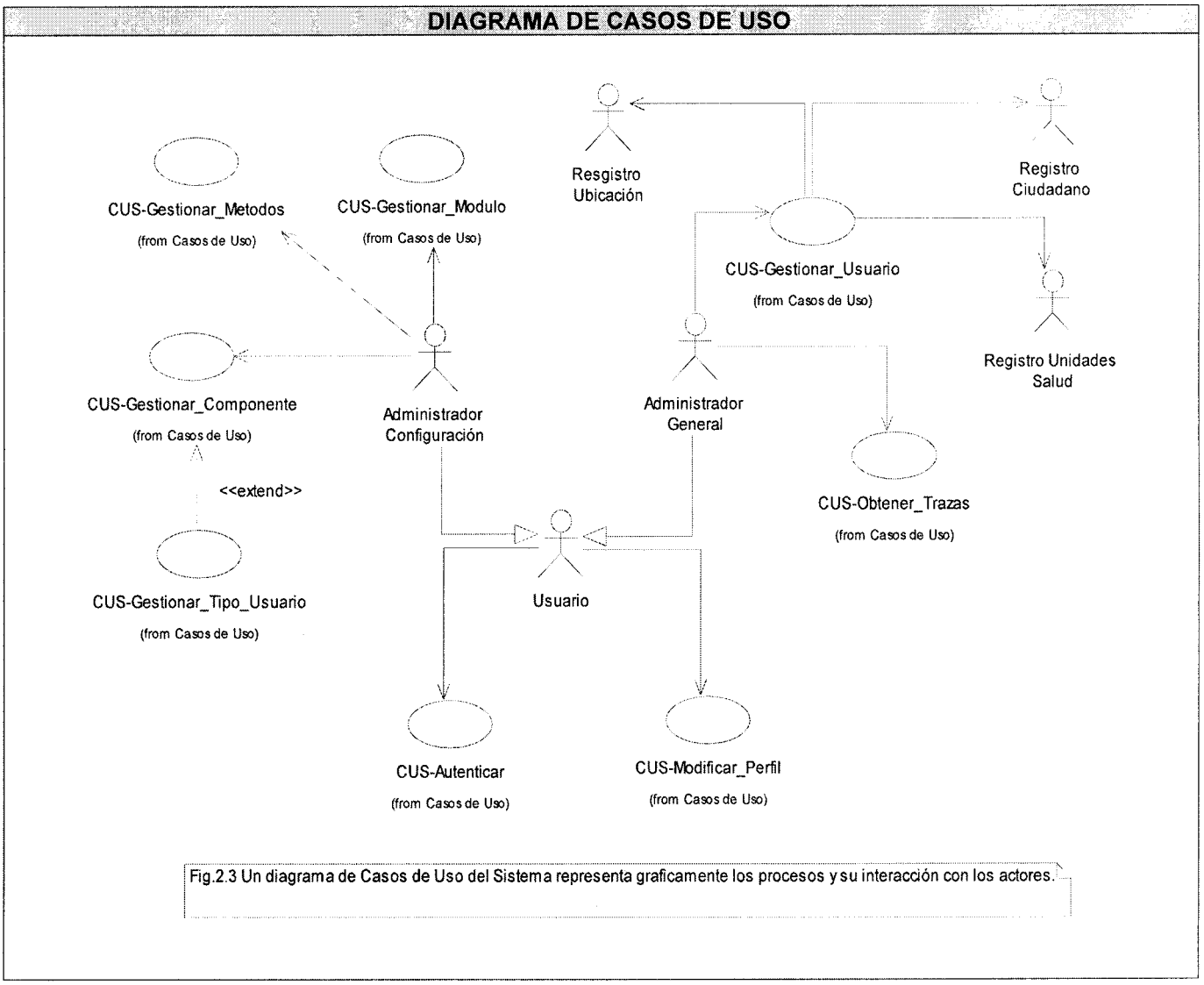



Figura 2.3 Diagrama de Casos de Uso del Sistema

### 2.3.2.3 Descripción textual de los Casos de Uso

La descripción textual de un caso de uso, representa los procesos o flujos de funcionalidades que son objeto de automatización mediante el mismo, incluyendo cómo comienza, termina e interactúan los actores con este. A continuación se proporciona una descripción breve de los casos de uso identificados para el Centro de Control.

| ***CUS-1: Autenticar*** | |
|---|---|
| **Propósito:** | Permitir la entrada al sistema. |
| **Actores:** | **Usuario** (Inicia) |
| **Resumen:** | El caso de uso inicia cuando un usuario intenta acceder al Sistema de Información para la Salud, este introduce su nombre de usuario y contraseña, accediendo al sistema con los privilegios que le fueron otorgados. El caso de uso termina cuando el actor haya accedido al sistema. |
| **Precondición:** | El usuario debe conocer el nombre de usuario y contraseña que le fue asignado en el momento en que el Administrador General haya registrado a dicho usuario. |
| **Poscondición:** | Permite la entrada del usuario al Centro de Control. Si no se realiza la operación, el sistema muestra un mensaje al actor. |
| **Referencias:** | **RF1, RF2, RF3, RF4, RF5** |
| **Prioridad:** | Crítico |

Tabla 2.4 Descripción textual *CUS-1: Autenticar*

| ***CUS-2: Modificar_Perfil*** | |
|---|---|
| **Propósito:** | Permitir a un usuario determinado cambiar su contraseña. |
| **Actores:** | **Usuario** (Inicia) |
| **Resumen:** | El caso de uso inicia cuando un usuario desea cambiar su contraseña, este escribe la nueva contraseña y la rectifica, el sistema muestra un mensaje de confirmación de la realización de la operación, terminado de esta forma el caso de uso. |
| **Precondición:** | El usuario debe haber sido autenticado en el sistema. |
| **Poscondición:** | Contraseña modificada. Si no se realiza la operación, el sistema presenta un mensaje al actor. |
| **Referencias:** | **RF24** |
| **Prioridad:** | Auxiliar |

Tabla 2.5 Descripción textual *CUS-2: Modificar_Perfil*

| ***CUS-3: Gestionar_Usuarios*** | |
|---|---|
| **Propósito:** | Gestionar la información relacionada con los usuarios del Sistema de Información para la Salud. |
| **Actores:** | **Administrador General** (Inicia) |
| **Resumen:** | El caso de uso inicia cuando el actor accede al Centro de Control para realizar acciones sobre un usuario, como insertarlo, buscarlo, eliminarlo o modificarlo; si desea insertar uno nuevo debe seleccionar el ciudadano correspondiente para homologar los datos del usuario con los de la persona asociada, una vez obtenido la información necesaria de la persona son capturados los datos restantes del usuario, como el período de actividad, correo electrónico, nombre de usuario (login) y especificándose si este usuario es administrador o no, si no es administrador se le asignan los privilegios desglosados por módulos y tipo de usuario en estos módulos, si se crea un Administrador General se especifica si es del nivel inmediato inferior o de una Unidad de Salud subordinada a su nivel de dirección y creándose el usuario en dependencia de lo seleccionado, si el actor desea eliminar o modificar un usuario primeramente tiene que buscarlo, una vez localizado, es realizada la acción deseada. Si el actor lo desea puede exportar los resultados de las búsquedas realizadas. El caso de uso termina cuando el actor termine las opciones de gestión deseadas sobre los usuarios. |
| **Precondición:** | El usuario debe haber sido autenticado en el sistema. |
| **Poscondición:** | El sistema deja actualizada la información perteneciente al usuario. Si no se realiza la operación, el sistema muestra un mensaje al actor. |
| **Referencias:** | **RF5, RF9, RF12, RF13, RF19, RF21, RF23, RF27, RF29, RF31, RF33, RF34, RF35** |
| **Prioridad:** | Crítico |

Tabla 2.6 Descripción textual *CUS-3: Gestionar_Usuarios*

| *CUS-4: Obtener_Trazas* | |
|---|---|
| **Propósito:** | Obtener las trazas registradas en el sistema. |
| **Actores:** | **Administrador General** (Inicia) |
| **Resumen:** | El caso de uso inicia cuando el actor desea obtener listados de las trazas registradas por el sistema, la búsqueda el actor la puede realizar por varios parámetros según lo que desee obtener, una vez llenado estos parámetros, se le mostrará las trazas que tiene registradas el sistema que coinciden con estos parámetros, si el actor desea puede exportar el resultado obtenido. Al terminar las labores de auditoría en las trazas concluye el caso de uso. |
| **Precondición:** | El usuario debe haber sido autenticado en el sistema. |
| **Poscondición:** | El actor obtiene las trazas. Si no se encontró ningún resultado el sistema le muestra un mensaje al actor. |
| **Referencias:** | **RF23, RF26, RF30, RF32** |
| **Prioridad:** | Secundario |

Tabla 2.7 Descripción textual *CUS-4: Obtener_Trazas*

| *CUS-5: Gestionar Tipo_Usuario [Extendido: CUS-6: Gestionar_Componentes]* | |
|---|---|
| **Propósito:** | Gestionar un tipo de usuario. |
| **Actores:** | **Administrador de Configuración** (Inicia) |
| **Resumen:** | El caso de uso se inicia cuando se desea gestionar los tipos de usuario de un componente determinado. En caso de asignar los tipos de usuario al componente involucrado se proporciona un listado con los que ya se encuentran registrados en el sistema, seleccionándose aquellos que estén asociados a este componente, en caso de no encontrarse alguno, este nuevo tipo puede ser registrado para posteriormente ser seleccionado. En el caso de eliminar o modificar un tipo de usuario se selecciona para realizar alguna de estas acciones. Concluyendo así el caso de uso. |
| **Precondición:** | El usuario debe haber sido autenticado en el sistema. |
| **Poscondición:** | El sistema deja actualizada la información perteneciente al tipo de usuario, así como su modificación o eliminación física en caso de que sea posible. Si no se realiza la operación, el sistema muestra un mensaje al actor. |
| **Referencias:** | **RF7, RF27** |
| **Prioridad:** | Secundario |

Tabla 2.8 Descripción textual *CUS-5: Gestionar_Tipo_Usuario*

| ***CUS-6: Gestionar_Componentes*** | |
|---|---|
| **Propósito:** | Gestionar la información relacionada con los componentes desplegados en SISalud. |
| **Actores:** | **Administrador de Configuración** (Inicia) |
| **Resumen:** | El caso de uso inicia cuando el actor intenta realizar todas las operaciones relacionadas con los componentes como crearlos, eliminarlos, buscarlos y listarlos, si el actor desea crear un componente, llena todos los datos generales que lleva el componente en su creación, además de configurar la integración de este componente con otros componentes, especificando por qué método y restricción de integridad dependerán los mismos, si desea eliminar o modificar un componente ya insertado, primeramente tiene que buscarlo para realizar la acción deseada, concluyendo así el caso de uso. |
| **Precondición:** | El usuario debe haber sido autenticado en el sistema. |
| **Poscondición:** | El sistema deja actualizada la información perteneciente al componente además de asignarle el tipo de usuario y método. Si no se realiza la operación, el sistema muestra un mensaje al actor. |
| **Referencias:** | **RF6, RF15, RF17, RF18, RF21, RF25, RF26** |
| **Prioridad:** | Crítico |

Tabla 2.9 Descripción textual *CUS-6: Gestionar_Componentes*

| ***CUS-7: Gestionar_Métodos*** | |
|---|---|
| **Propósito:** | Gestionar los métodos de los componentes desplegados en el Sistema de Información para la Salud. |
| **Actores:** | **Administrador de Configuración** (Inicia) |
| **Resumen:** | El caso de uso inicia cuando el actor intenta insertar, modificar o eliminar un método, si el actor desea crear un método inserta la información específica que este necesita en su creación como puede ser el componente al que pertenece y los niveles en los cuáles puede ser invocado el mismo, si el actor desea buscar un método, llena los parámetros de búsqueda, obteniendo el método para poder eliminarlo o modificarlo según desee terminando de este modo el caso de uso. |
| **Precondición:** | El usuario debe haber sido autenticado en el sistema. |
| **Poscondición:** | El sistema deja actualizada la información perteneciente a los métodos. Si no se realiza la operación, el sistema muestra un mensaje al actor. |
| **Referencias:** | **RF8, RF11, RF15, RF22, RF25, RF26, RF27** |
| **Prioridad:** | Crítico |

Tabla 2.10 Descripción textual *CUS-7: Gestionar_Métodos*

| ***CUS-8: Gestionar_Módulos*** | |
|---|---|
| **Propósito:** | Gestionar la información relacionada con los módulos del Sistema de Información para la Salud. |
| **Actores:** | **Administrador de Configuración** (Inicia) |
| **Resumen:** | El caso de uso inicia cuando un actor intenta crear, eliminar, buscar o listar los módulos registrados, si desea registrar un nuevo módulo debe llenar los datos necesarios, además de especificar los niveles en los cuáles se tendrá acceso a este módulo, además se debe especificar cuales son los componentes que lo conforman indicando cuál de ellos es el componente base, si el actor desea eliminar o modificar un módulo debe realizar la búsqueda de este primeramente, así podrá modificar los datos que se encuentren registrado o eliminarlo en caso de que sea posible. Si el actor lo desea, puede exportar los resultados de las búsquedas realizadas. Terminando de esta forma el caso de uso. |
| **Precondición:** | El usuario debe haber sido autenticado en el Sistema. |
| **Poscondición:** | El sistema deja actualizada la información perteneciente a los módulos, así como su modificación o eliminación física en caso de que sea posible. Si no se realiza la operación, el sistema presenta un mensaje al actor |
| **Referencias:** | **RF11, RF14, RF15, RF16, RF25, RF29, RF36, RF37, RF38** |
| **Prioridad:** | Crítico |

Tabla 2.11 Descripción textual *CUS-8: Gestionar_Módulos*

## 2.4 Conclusiones

Con el desarrollo del presente capítulo se obtuvo una perspectiva del sistema que se desea construir, en términos de requerimientos funcionales y no funcionales. Además, se identificaron las personas y sistemas que tendrán un comportamiento de emisor o receptor de la información que es manipulada por el Centro de Control. Finalmente, fueron descritos los flujos de eventos asociados a los Casos de Uso definidos, priorizándose y estableciendo la referencia existente entre estos y las operaciones que el sistema debe proveerle a los usuarios finales. Este segundo capítulo, constituye la base para el desarrollo exitoso de los subsiguientes Flujos de Trabajo, del Proceso Unificado de Desarrollo, que son objetivo de la presente investigación, siendo estos: Diseño e Implementación.

# CAPÍTULO 3: DISEÑO DEL SISTEMA

## 3.1 Introducción

En el Diseño se modela y adquiere forma el sistema para que soporte todos los requisitos funcionales o no funcionales e inclusive cualquier otro tipo de restricción, contribuyendo a obtener una arquitectura sólida y estable para la futura implementación del sistema de software. En tal sentido el propósito de este capítulo está encaminado a adquirir una comprensión de los aspectos relacionados con los requerimientos, lenguajes de programación, componentes reutilizables, tecnologías de distribución y concurrencia. Entre los artefactos que serán mostrados en el presente capítulo se encuentran: el Modelo de Diseño, especificándose la estructura y la definición de los elementos que este posee, Diagramas de Clases y descripción de las clases de diseño.

## 3.2 Modelo de Diseño

Un Modelo de Diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema. Este artefacto constituye la entrada fundamental utilizada para el correcto desarrollo de las entradas de implementación. En el caso que ocupa al Centro de Control alguno de los elementos de este Modelo como el diagrama de Clases Persistentes, Modelo de Datos y Diagramas de Interacción, no se encuentran por constituir información confidencial y sensible, relacionada con la seguridad de las soluciones desarrolladas para el MINSAP.

### 3.2.1 Estructuración

Antes de comenzar a desarrollar el Modelo de Diseño, es necesario definir la descomposición de este en subsistemas, con sus interfaces y las dependencias. Esta descomposición es muy significativa para la arquitectura en general, debido a que los subsistemas y sus interfaces constituyen la estructura fundamental del producto de software. En el caso que ocupa al desarrollo del Centro de Control, se ha

modelado un Diagrama de Subsistemas de Servicios, entiéndase este diagrama como un conjunto de clases agrupadas por compartir funcionalidades similares y las relaciones que se establecen entre sí. Del mismo modo, una de las características primordiales de estos subsistemas es su capacidad de reutilización, lo cual se ajusta a la arquitectura definida para el Centro de Control y el Sistema de Información para la Salud, SOA-CBA. A continuación se muestran estos subsistemas los cuales serán detallados posteriormente:

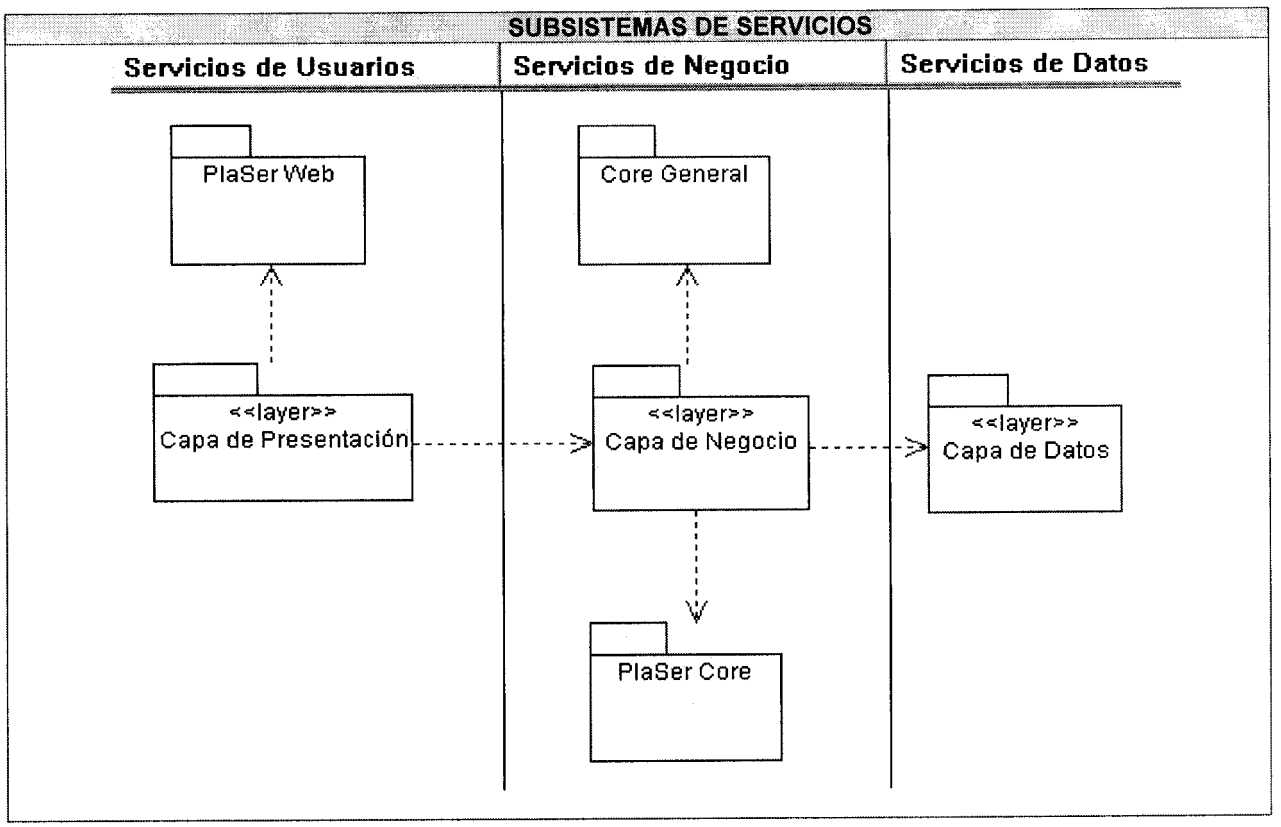



Figura 3.1 Diagrama de Subsistemas de Servicios

A continuación, se describirá el contenido de cada uno de estos subsistemas de servicios, posibilitando una adecuada comprensión de los Diagramas de Clases del Diseño.

**PlaSer Web:** En este subsistema se encuentran las librerías Plaser_XML y PlaSer_XSLT, permiten la manipulación de los documentos XML y el formateo de estos para ser mostrados a los clientes a través de páginas XHTML respectivamente.

**Capa de Presentación:** Las funcionalidades de la capa de presentación consisten en intercambiar información con los actores, por lo tanto es la única capa con la que interactúa directamente un usuario. Contiene los ficheros mediante los cuales son invocados los métodos de la Capa de Negocio, entre ellos pueden ser mencionados los documentos XSL utilizados para la transformación de los XML y ficheros JavaScript utilizados para las validaciones y tratamiento de errores o excepciones.

**Plaser Core:** Este subsistema está representado por la clase de acceso a datos Plaser_DBz y los ficheros confplaser_client y confplaser_server, utilizados como ficheros de configuración de un componente determinado, el primero contiene la URL para la comunicación con los componentes dependientes y el segundo de estos los métodos pertenecientes al componente desplegado.

**Core General:** Representa los ficheros utilizados en la Capa de Negocio para las validaciones y generación de ficheros en formato Portable Document Format (PDF) o Microsoft Office Excel (XLS).

**Capa de Negocio:** Esta capa también se conoce como Capa de Dominio o Capa de Lógica de Negocio. Establece la comunicación entre la capa de presentación y la capa de datos, encargada de recibir y responder cada petición de los usuarios. Los ficheros que la conforman reciben las solicitudes de los clientes, se comunican con las capa de datos, actualizando o recuperando información emitiendo una respuesta. Constituye la parte del sistema donde se establecen todas las reglas de negocio que deben cumplirse.

**Capa de Datos:** Representa las tablas de la Base de Datos del Centro de Control, es el repositorio físico de la información gestionada por el sistema, la cual puede ser recuperada, actualizada o eliminada a través de la Capa de Negocio.

### 3.2.2 Definición de elementos de diseño

Para la realización de los Diagramas de Clases del Diseño será utilizada la extensión de UML para el modelado de aplicaciones Web, publicada por Jim Conallen a fines de 1999 mediante el artículo "*Modelling Web Applications Architectures with UML*". Así esta extensión presenta como elementos más significativos a tres clases UML estereotipadas "Server Page", "Client Page" y "Form" empleadas para el código servidor, código cliente y formularios respectivamente, permitiendo además representar ficheros contenedores de sentencias script como por ejemplo PHP y JavaScript. Entre estos elementos de diseño se establece un conjunto de relaciones las cuales son especificadas en la siguiente tabla: [26]

| Hasta / Desde | Client Page | Form | Server Page |
|---|---|---|---|
| **Client Page** | <<link>> , <<redirect>> | aggregation | <<link>> , <<redirect>> |
| **Form** | aggregated by | | <<submit>> |
| **Server Page** | <<build>>, <<redirect>> | | <<redirect>>,<<include>> |

Tabla 3.1 Relaciones entre las clases principales que conforman la extensión de UML para Web.

El código servidor se encarga de construir o generar el resultado XHTML que conforma el código cliente (*<<build>>*), los formularios envían sus datos al código servidor para ser procesados los pedidos (*<<submit>>*), además forman parte del código cliente o resultado XHTML, es por esto que la relación entre la clase empleada para el código cliente y la clase empleada para el formulario es de agregación. Entre páginas clientes pueden existir vínculos (*<<link>>*) o redireccionamientos (*<<redirect>>*). Es importante destacar que una página cliente es construida por una sola página servidora. Esta a su vez, puede completar su funcionamiento incluyendo código existente en otra página de este mismo tipo, utilizando la relación de inclusión (*<<include>>*), que aunque no es propia de la extensión de UML, las herramientas para el modelado la consideran en aras de representar todas las relaciones existentes en el modelo. [27]

Para la nomenclatura de estas clases en los Diagramas de Clases del Diseño del Centro de Control, se siguió la siguiente estructura: CI_<NombreClaseCliente>, FR_<NombreFormulario>,

SP_<NombreClaseServidora> y <NombreMétodo>.php, para las páginas clientes, formularios, clases servidoras ubicadas en la Capa de Presentación y métodos de la Lógica de Negocio respectivamente.

### 3.2.3 Diagramas de Clases del Diseño

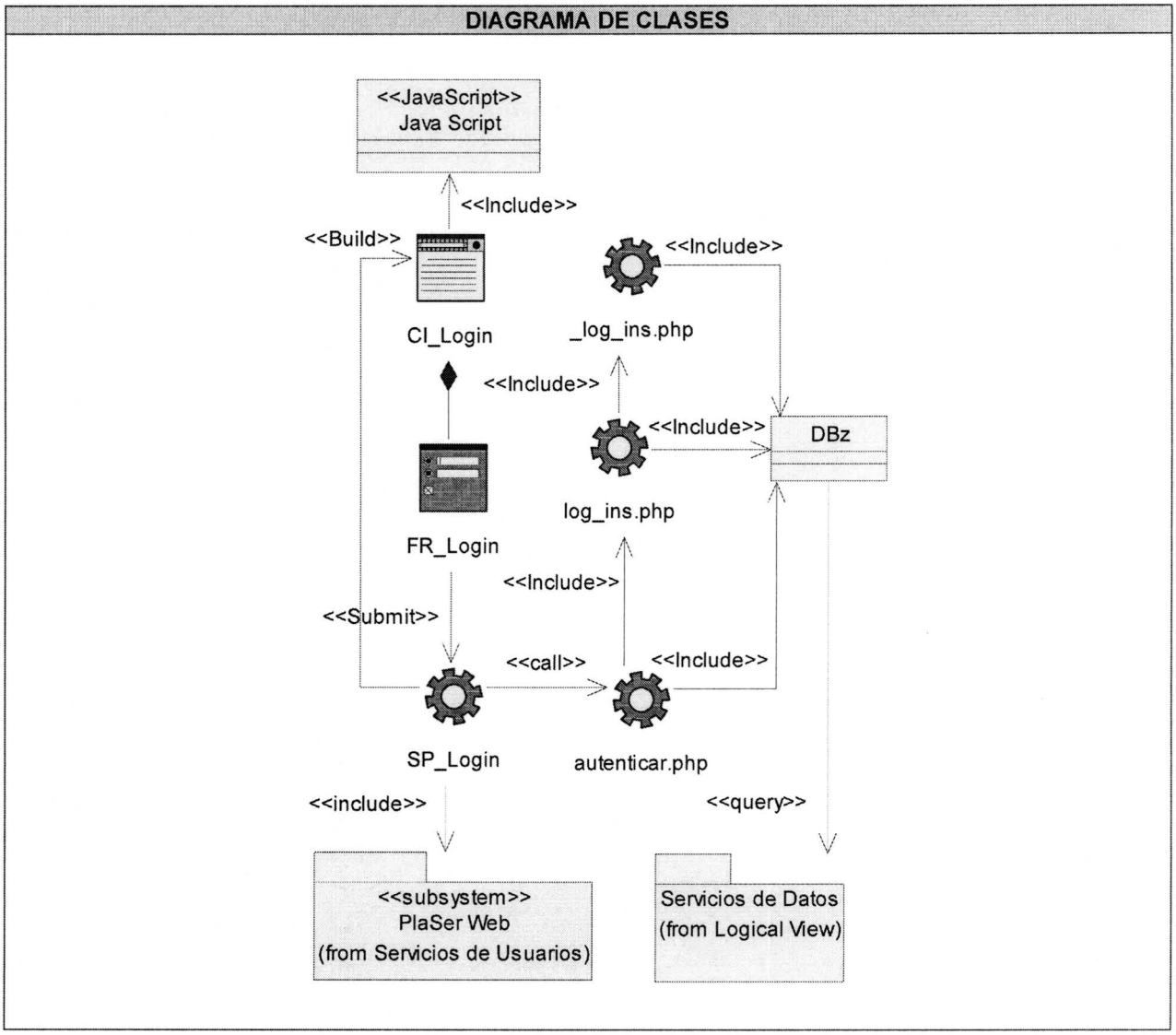



Figura 3.1 Diagrama de Clases del Diseño *CUS-1: Autenticar*

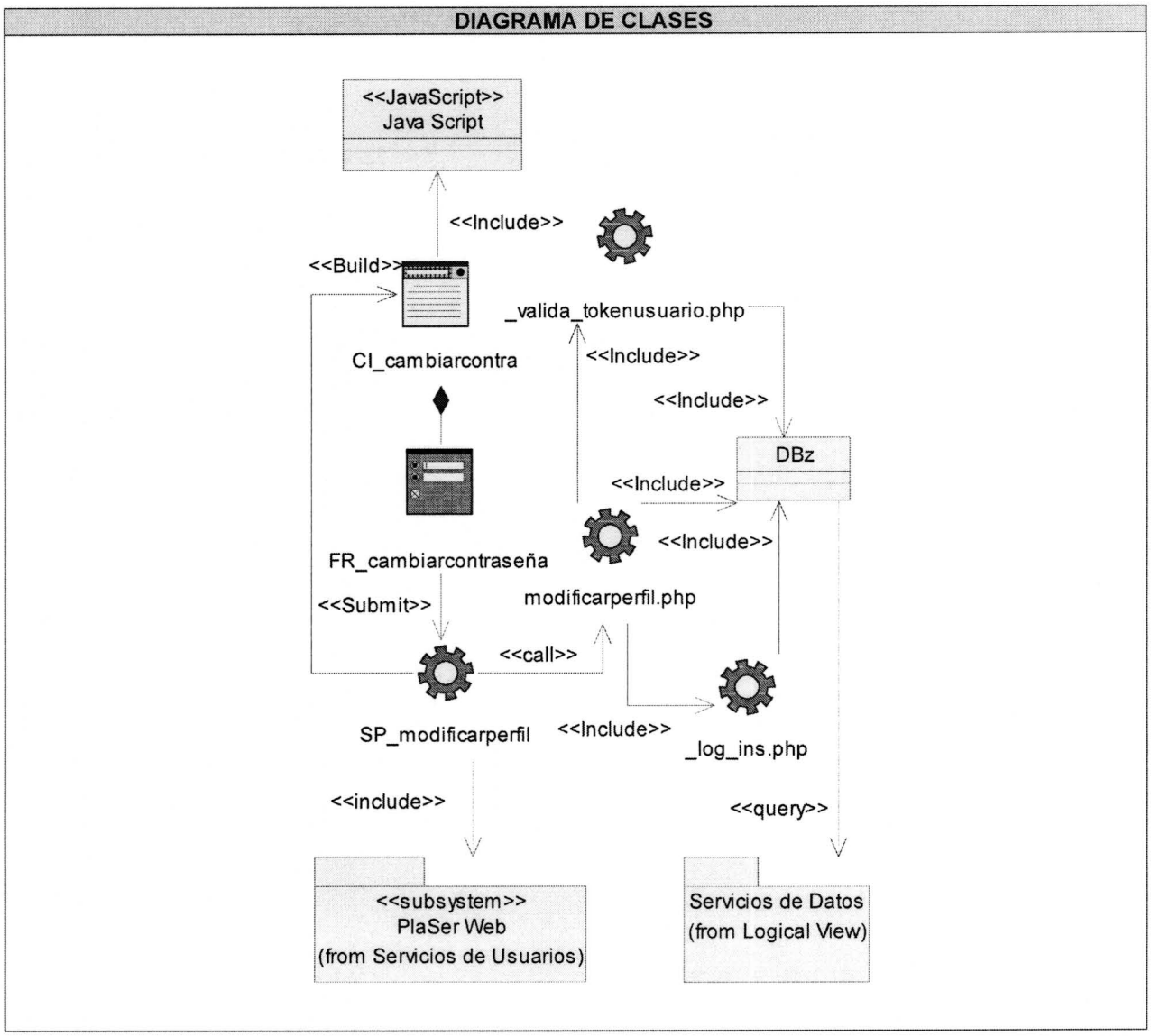



Figura 3.2 Diagrama de Clases del Diseño *CUS-2: Modificar_Perfil*

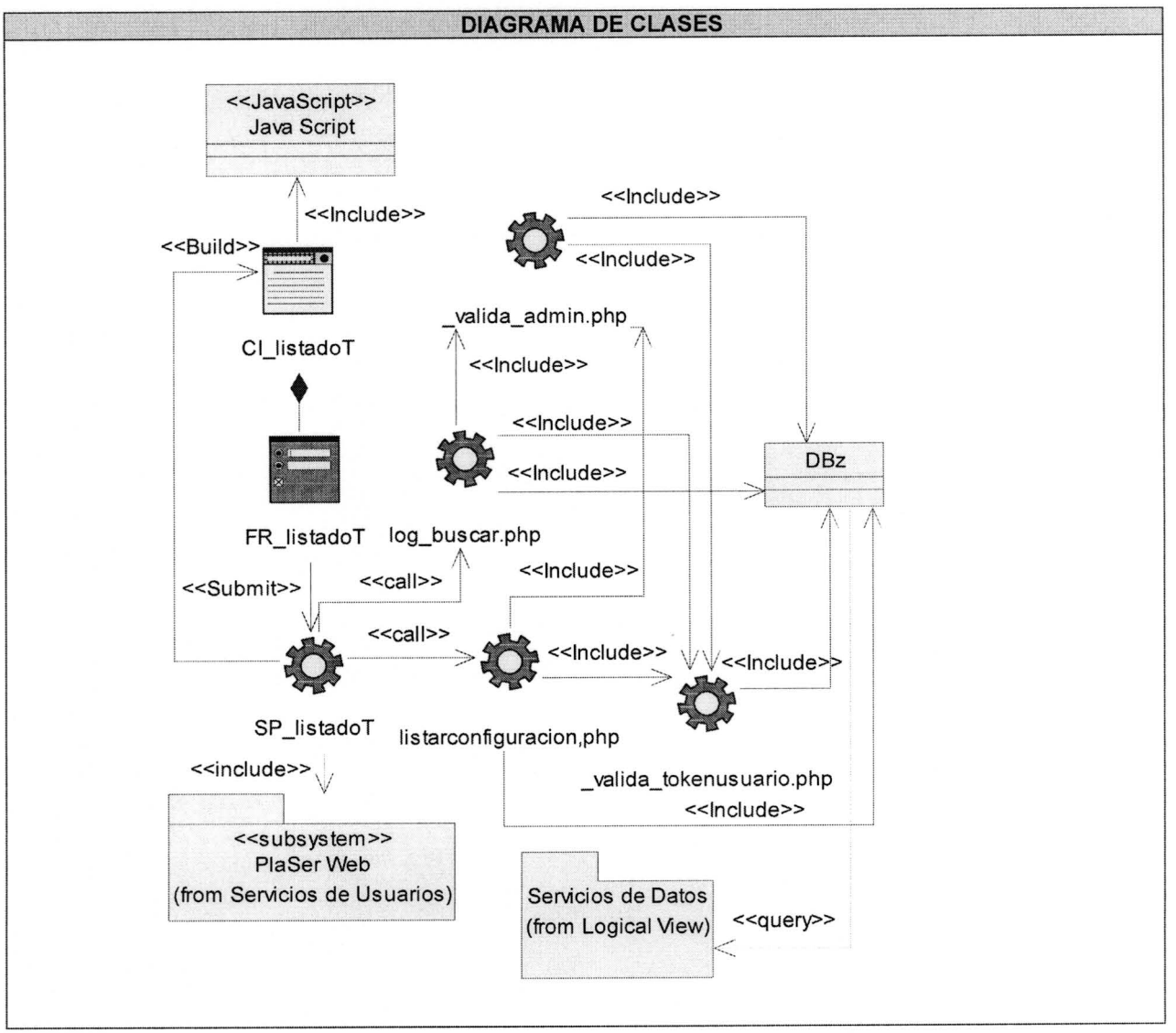



Figura 3.3 Diagrama de Clases del Diseño *CUS-4: Obtener_Trazas*

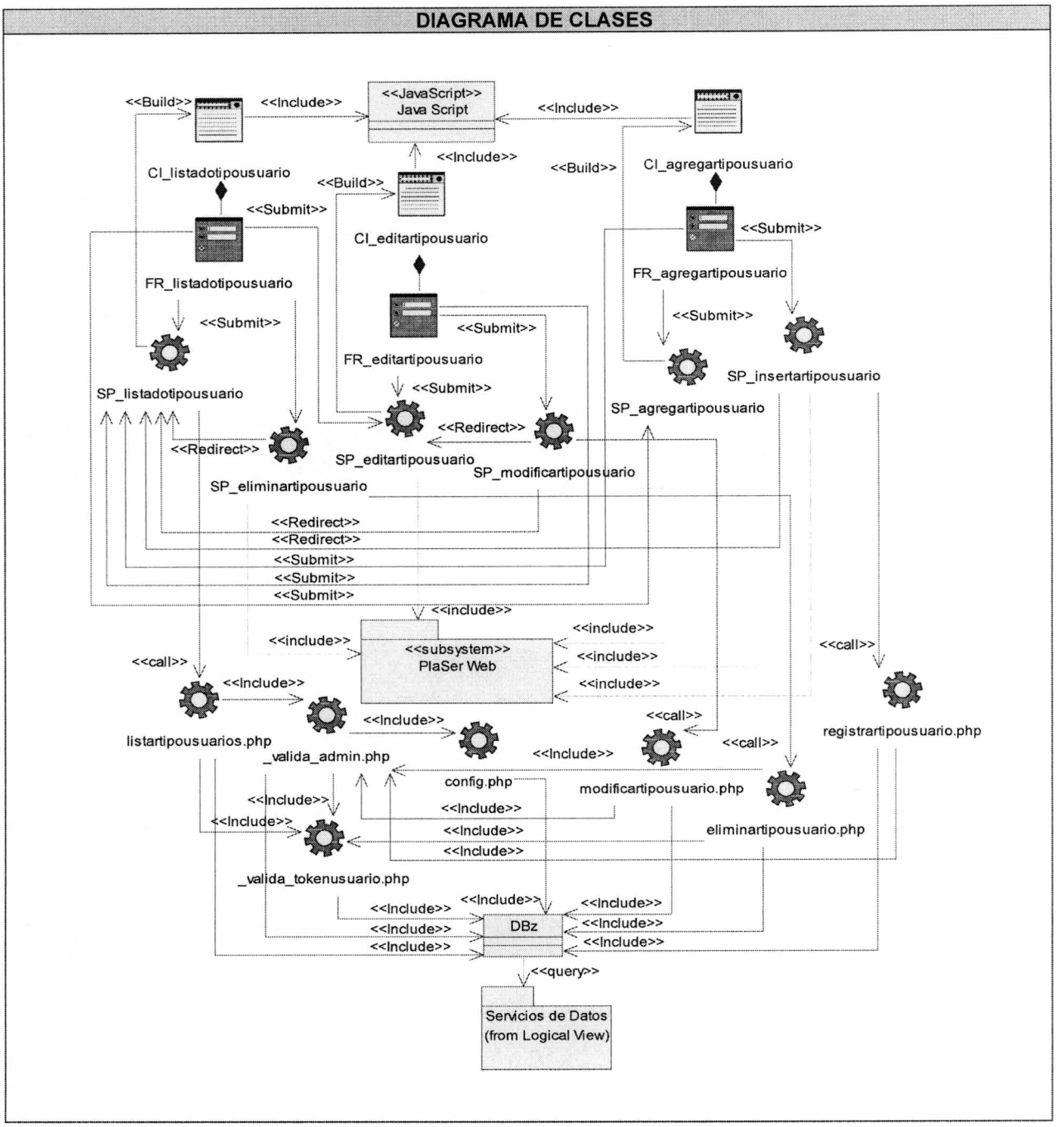



Figura 3.4 Diagrama de Clases del Diseño *CUS-5: Gestionar_Tipo_Usuario*

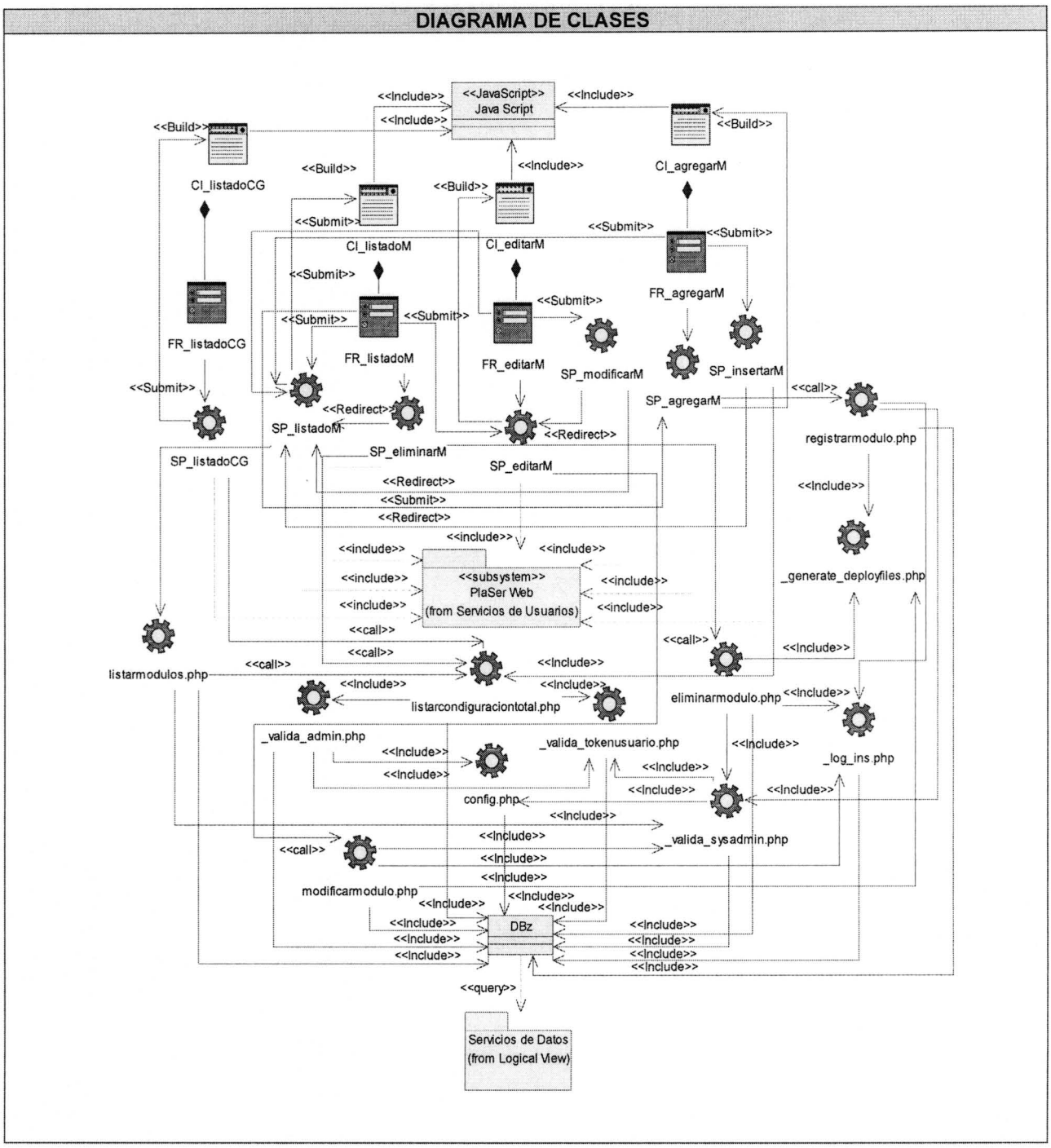



Figura 3.5 Diagrama de Clases del Diseño *CUS-8: Gestionar_Módulos*

### 3.2.4 Descripción de clases y atributos

Seguidamente serán explicadas algunas de las clases que han sido identificadas para su futura implementación. La estructura general de la Capa de Presentación para cada uno de los casos de uso identificados en el Centro de Control es similar contando con las mismas páginas clientes servidoras, del mismo modo serán descritas algunas de las responsabilidades que realizarán las páginas servidoras que responden a la Lógica de Negocio. De esta manera se tendrá una comprensión mayor del funcionamiento que tendrá el sistema en desarrollo.

#### 3.2.4.1 Descripción de páginas clientes

| **Nombre:** ***Cl_Login*** |
|---|
| **Tipo de clase:** Client Page |
| **Descripción General:** Constituye la primera página Web con la que interactúa un usuario del Sistema de Información para la Salud, la misma tiene el objetivo de capturar el nombre de usuario y contraseña para poder acceder al sistema, constituyendo la página de bienvenida al mismo. Es utilizada en el caso de uso autenticar. |

Tabla 3.1 Descripción página cliente *Cl_Login*

| **Nombre:** ***Cl_cambiarcontra*** |
|---|
| **Tipo de clase:** Client Page |
| **Descripción General:** Esta página le brinda la posibilidad al usuario de cambiar su contraseña del sistema, la cual inicialmente es creada por un Administrador General, consta de tres textfield utilizados para el nombre de usuario, el cual aparece por defecto sin poder ser modificado y otros dos para la nueva contraseña y la confirmación de esta. |

Tabla 3.2 Descripción página cliente *Cl_cambiarcontra*

| **Nombre:** ***Cl_listado*** |
|---|
| **Tipo de clase:** Client Page |
| **Descripción General:** La clase *Cl_listado* es una página Web que se ejecuta del lado del cliente sobre un navegador Web. Permite a los administradores la realización de listados y búsqueda de información |

| |
|---|
| específica, permitiendo generar documentos Portable Document Format (PDF) y Microsoft Office Excel (XLS). A través de esta página se puede registrar, modificar o eliminar la información almacenada o simplemente visualizar información. El resultado de las búsquedas es paginada cada 10 elementos, permitiendo la movilidad por tales resultados inclusive ir directamente a la última página. Es utilizada en los siguientes Casos de Uso del Sistema:<br>o Gestionar_Usuarios<br>o Obtener_Trazas<br>o Gestionar_Tipo_Usuario<br>o Gestionar_Componentes<br>o Gestionar_Métodos<br>o Gestionar_Módulos |

Tabla 3.3 Descripción página cliente *Cl_listado*

| |
|---|
| **Nombre: *Cl_agregar*** |
| **Tipo de clase:** Client Page |
| **Descripción General:** La clase *Cl_agregar* es una página Web que se ejecuta del lado del cliente sobre un navegador Web. Permite capturar los datos que serán registrados para su posterior almacenamiento en la base de datos del Centro de Control, posee un conjunto de validaciones JavaScript que permita no realizar peticiones innecesarias y por lo tanto se incrementa la usabilidad. Cuenta además con una leyenda en aquellas funcionalidades que lo requieran por manipular abreviaturas desconocidas para los clientes. Es utilizada en los siguientes Casos de Uso del Sistema:<br>o Gestionar_Usuarios<br>o Gestionar_Tipo_Usuario<br>o Gestionar_Componentes<br>o Gestionar_Métodos<br>o Gestionar_Módulos |

Tabla 3.4 Descripción página cliente *Cl_agregar*

| **Nombre:** ***Cl_editar*** |
|---|
| **Tipo de clase:** Client Page |
| **Descripción General:** La clase *Cl_editar* es una página Web que se ejecuta del lado del cliente. Muestra los datos del elemento que ha sido previamente seleccionado para editar desde la página cliente *Cl_listado*. Permite modificar la información que se visualiza en la misma. Es utilizada en los siguientes Casos de Uso del Sistema:<br>o Gestionar_Usuarios<br>o Gestionar_Tipo_Usuario<br>o Gestionar_Componentes<br>o Gestionar_Métodos<br>o Gestionar_Módulos |

Tabla 3.5 Descripción página cliente *Cl_editar*

### 3.2.4.2 Descripción de páginas servidoras

| **Nombre:** ***SP_Login*** |
|---|
| **Tipo de clase:** Server Page |
| **Descripción General:** Página servidora ubicada en la Capa de Presentación. Su tarea fundamental es construir la página cliente *Cl_Login*. Aplica un fichero XSLT (login.xsl) a un documento XML para transformarlo y mostrarlo al usuario en formato XHTML. Invoca al método del negocio autenticar.php para obtener la Lista de derechos de los usuarios desglosados por módulos, además a través de la invocación del método log_ins.php se registra el acceso del usuario al sistema. |

Tabla 3.6 Descripción página servidora *SP_Login*

| **Nombre:** ***SP_modificarperfil*** |
|---|
| **Tipo de clase:** Server Page |
| **Descripción General:** Clase servidora ubicada en la Capa de Presentación. Su tarea fundamental es construir la página cliente *Cl_cambiarcontra*. Aplica un fichero XSLT (modificar_perfil.xsl) a un documento XML para transformarlo y mostrarlo al usuario en formato XHTML. |

Tabla 3.6 Descripción página servidora *SP_modificarperfil*

| **Nombre: SP_listado** |
| --- |
| **Tipo de clase:** Server Page |
| **Descripción General:** La clase *SP_listado* es una clase que se ejecuta del lado del servidor en la Capa de Presentación. Su actividad fundamental es construir la página cliente *Cl_listado*. Aplica un documento XSLT (listado.xsl) a otro documento XML para transformarlo y mostrarlo al usuario en formato XHTML. Es utilizada en los siguientes Casos de Uso del Sistema:<br>o Gestionar_Usuarios<br>o Gestionar_Tipo_Usuario<br>o Gestionar_Componentes<br>o Gestionar_Métodos<br>o Gestionar_Módulos |

Tabla 3.7 Descripción página servidora *SP_listado*

| **Nombre: *SP_agregar*** |
| --- |
| **Tipo de clase:** Server Page |
| **Descripción General:** Clase servidora ubicada en la Capa de Presentación. Su tarea fundamental es construir la página cliente *Cl_agregar*. Aplica un fichero XSLT (nuevo.xsl) a un documento XML para transformarlo y mostrarlo al usuario en formato XHTML. Es utilizada en los siguientes Casos de Uso del Sistema:<br>o Gestionar_Usuarios<br>o Gestionar_Tipo_Usuario<br>o Gestionar_Componentes<br>o Gestionar_Métodos<br>o Gestionar_Módulos |

Tabla 3.8 Descripción página servidora *SP_agregar*

| **Nombre: *SP_editar*** |
| --- |
| **Tipo de clase:** Server Page |
| **Descripción General:** Clase servidora ubicada en la Capa de Presentación. Su tarea fundamental es construir la página cliente *Cl_editar*. Aplica un fichero XSLT (editar.xsl) a un documento XML para transformarlo y mostrarlo al usuario en formato XHTML. Es utilizada en los siguientes Casos de Uso |

| |
|---|
| del Sistema:<br>o Gestionar_Usuarios<br>o Gestionar_Tipo_Usuario<br>o Gestionar_Componentes<br>o Gestionar_Métodos<br>o Gestionar_Módulos |

Tabla 3.9 Descripción página servidora *SP_editar*

| **Nombre: *SP_insertar*** |
|---|
| **Tipo de clase:** Server Page |
| **Descripción General:** La clase *SP_insertar* es una página que se ejecuta del lado del servidor en la Capa de Presentación. Recibe y valida los datos que se envían desde la página cliente *Cl_nuevo*. Codifica las descripciones y construye las estructuras que serán enviadas a la Capa de Negocio, se invoca al método del negocio para el registro de los nuevos datos y una vez concluida la ejecución de sus responsabilidades redirecciona su ejecución a la clase *SP_listado*. Es utilizada en los siguientes casos de uso.<br>o Gestionar_Usuarios<br>o Gestionar_Tipo_Usuario<br>o Gestionar_Componentes<br>o Gestionar_Métodos<br>o Gestionar_Módulos |

Tabla 3.10 Descripción página servidora *SP_insertar*

| **Nombre: *SP_modificar*** |
|---|
| **Tipo de clase:** Server Page |
| **Descripción General:** La clase *SP_modificar* es una página que se ejecuta del lado del servidor en la Capa de Presentación. Recibe y valida los datos que se envían desde la página cliente *Cl_editar*. Codifica las descripciones y construye las estructuras que serán enviadas a la Capa de Negocio, se invoca al método del negocio para la modificación del los datos involucrados y una vez concluida la ejecución de sus responsabilidades redirecciona su ejecución a la clase *SP_listado*. Es utilizada en los siguientes casos de uso. |

| |
|---|
| ○ Gestionar_Usuarios<br>○ Gestionar_Tipo_Usuario<br>○ Gestionar_Componentes<br>○ Gestionar_Métodos<br>○ Gestionar_Módulos |

Tabla 3.11 Descripción página servidora *SP_modificar*

| **Nombre: *SP_eliminar*** |
|---|
| **Tipo de clase:** Server Page |
| **Descripción General:** La clase *SP_eliminar* es una página que se ejecuta del lado del servidor en la Capa de Presentación. Recibe el identificador del elemento que se desea eliminar desde la página *SP_listado*. Invoca al método del negocio necesario y una vez concluida la ejecución de sus responsabilidades redirecciona su ejecución a la clase *SP_listado*. Es utilizada en los siguientes casos de uso.<br>○ Gestionar_Usuarios<br>○ Gestionar_Tipo_Usuario<br>○ Gestionar_Componentes<br>○ Gestionar_Métodos<br>○ Gestionar_Módulos |

Tabla 3.12 Descripción página servidora *SP_eliminar*

### 3.2.4.3 Descripción de métodos del negocio

| **Nombre: *autenticar.php*** |
|---|
| **Tipo de clase:** Server Page |
| **Caso de Uso:** Autenticar |
| **Descripción General:** Permite el acceso de los usuarios al sistema, comprueba si el usuario que está solicitando el acceso se encuentra registrado en el Centro de Control y si su cuenta de usuario está habilitada, es decir si es un usuario activo. Si no se cumple alguna de estas condiciones se reporta un error de acceso. En el caso contrario es generado su token de acceso de manera aleatoria, registrándose la fecha y hora en que se autorizó el acceso al usuario, almacenándose mediante la |

| página log_ins.php una traza de esta operación. La misma devuelve una estructura necesaria para la ejecución de las funcionalidades del sistema y una Lista de Derechos que contiene sus privilegios desglosados por módulos y tipos de usuario. | |
|---|---|
| **PARÁMETROS DE ENTRADA** | |
| **Descripción** | **Tipo** |
| login | String |
| password | String |

Tabla 3.13 Descripción del método *autenticar.php*

| **Nombre: *log_ins.php*** | |
|---|---|
| **Tipo de clase:** Server Page | |
| **Caso de Uso:** Obtener_Trazas | |
| **Descripción General:** Esta es una página que se ejecuta del lado del servidor formando parte de la lógica de negocio del sistema. Permite el registro de las trazas en el sistema. | |
| **PARÁMETROS DE ENTRADA** | |
| **Descripción** | **Tipo** |
| code | String |
| desc | String |
| componente | String |
| metodo | String |

Tabla 3.14 Descripción del método *log_ins.php*

| **Nombre: *modificarperfil.php*** | |
|---|---|
| **Tipo de clase:** Server Page | |
| **Caso de Uso:** Modificar_Perfil | |
| **Descripción General:** Esta es una página que se ejecuta del lado del servidor formando parte de la lógica de negocio del sistema. Es invocada por la página servidora *SP_modificar perfil*, permite al usuario cambiar su contraseña. | |
| **PARÁMETROS DE ENTRADA** | |
| **Descripción** | **Tipo** |
| tokenuser | String |
| password | String |

Tabla 3.15 Descripción del método *modificarperfil.php*

| **Nombre:** ***log_buscar.php*** | |
|---|---|
| **Tipo de clase:** Server Page | |
| **Caso de Uso:** Obtener_Trazas | |
| **Descripción General:** Permite realizar búsquedas de las trazas registradas en el sistema. Los administradores Generales sólo podrán visualizar las operaciones en las que han intervenido los usuarios que han sido creados por su nivel y ubicación, facilitando el proceso de auditoría y control. | |
| **PARÁMETROS DE ENTRADA** | |
| **Descripción** | **Tipo** |
| login | String |
| codigo | String |
| descripcion | String |
| idcomponente | Int |
| idmetodo | Int |
| datetime_ini | String |
| datetime_fin | String |
| niveladmin | Int |
| idniveladmin | Int |
| orden | String |
| ordenar | String |
| offset | Int |
| cantidad | Int |

Tabla 3.16 Descripción del método *log_buscar.php*

| **Nombre:** ***registrarmodulo.php*** | |
|---|---|
| **Tipo de clase:** Server Page | |
| **Caso de Uso:** Gestionar_Módulos | |
| **Descripción General:** Página servidora que es invocada por *SP_insertar*. Permite registrar un nuevo módulo en el Sistema de Información para la Salud. Del mismo son recogidos un conjunto de datos generales así como los componentes que se integran en este módulo y cuál de ellos es el base. | |
| **PARÁMETROS DE ENTRADA** | |
| **Descripción** | **Tipo** |
| nombre | String |
| url | String |
| alias | String |
| imagen | String |
| descripcion | String |
| componente_base | Int |

| | |
|---|---|
| composicion | Array |
| niveles | Array |

Tabla 3.17 Descripción del método *registrarmodulo.php*

| **Nombre: *modificarmodulo.php*** | |
|---|---|
| **Tipo de clase:** Server Page | |
| **Caso de Uso:** Gestionar_Módulos | |
| **Descripción General:** Página servidora que es invocada por *SP_modificar.* Permite actualizar un módulo existente en el Sistema de Información para la Salud. | |
| **PARÁMETROS DE ENTRADA** | |
| **Descripción** | **Tipo** |
| idmodulo | Int |
| nombre | String |
| url | String |
| alias | String |
| imagen | String |
| descripcion | String |
| componente_base | Int |
| composicion | Array |
| niveles | Array |

Tabla 3.18 Descripción del método *modificarmodulo.php*

| **Nombre: *eliminarmodulo.php*** | |
|---|---|
| **Tipo de clase:** Server Page | |
| **Caso de Uso:** Gestionar_Módulos | |
| **Descripción General:** Página servidora que es invocada por *SP_eliminar.* Permite eliminar un módulo existente en el Sistema de Información para la Salud, siempre y cuando no tenga usuarios asociados. | |
| **PARÁMETROS DE ENTRADA** | |
| **Descripción** | **Tipo** |
| idmodulo | Int |

Tabla 3.19 Descripción del método *eliminarmodulo.php*

## 3.3 Conclusiones

Al concluir este capítulo donde fue obtenido el Modelo de Diseño, se describió la realización física de los casos de uso, centrándose en cómo los requerimientos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a desarrollar. Además, constituyó una abstracción para la implementación y es de este modo utilizado como una entrada fundamental en las actividades que se realizan en el Flujo de Trabajo de Implementación.

# CAPÍTULO 4: IMPLEMENTACIÓN

## 4.1 Introducción

El presente capítulo constituye la secuencia lógica del Diseño, a través del mismo se implementarán las clases y subsistemas encontrados durante este en términos de componentes. Se realizará una fundamentación de la necesidad de la integración del Centro de Control con otros sistemas o componentes externos, proveedores de servicios de los cuales el Centro de Control es consumidor. Además, se obtendrá el Diagrama de Despliegue, como parte del Modelo de Implementación, que indicará cómo la solución implementada estará distribuida físicamente. Finalmente se proporcionará una detallada explicación de la solución dada al mantenimiento de la integridad referencial en una Arquitectura Orientada a Servicios utilizando PlaSer.

## 4.2 Integración con otros sistemas

Considerando que la arquitectura a utilizar es Orientada a Servicios y Basada en Componentes este proceso de reutilización de componentes es un hecho inevitable, téngase en cuenta que en el caso del Registro de Ciudadanos (RC) y el Registro de Ubicación Geográfica (RU), no pertenecen al dominio de Salud Pública, por lo que en un momento determinado serán sustituidos por sus equivalentes desarrollados por las entidades encargadas de estos temas en Cuba en el marco de la informatización de la sociedad, lo cual simplificará el proceso de mantenimiento y soporte del Centro de Control.

### 4.2.1 Registro de Ciudadanos (RC)

Este es un componente que almacena la misma información de las personas que son registradas por la Oficina del Carné de Identidad, entre la que puede ser mencionada datos generales, datos de nacimiento, dirección y datos de los padres. La integración con el Registro de Ciudadanos se realiza con el fin de homologar los datos de los usuarios del Sistema de Información para la Salud con los datos de un registro externo para validar los mismos. En el actual componente SAAA se encuentran registrados usuarios con

datos erróneos, lo que atenta contra el proceso de auditoría de la información y la localización de los usuarios en caso de alguna irregularidad. Este proceso de homologación además se convirtió en un requerimiento de los nuevos módulos en desarrollo, como por ejemplo el Registro de Población (RPOB) y el Registro de Enfermedades de Declaración Obligatoria (REDO), los cuales debían identificar a la persona autenticada, para asignarle sus privilegios y derechos en estos registros. Es utilizado para este proceso el método BuscarCiudadano el cual devuelve la información de los ciudadanos del país, teniendo en cuenta los criterios de búsqueda que reciba este método como parámetro.

### 4.2.2 Registro de Unidades de Salud (RUS)

Los servicios el Registro de Unidades de Salud son consumidos con el objetivo de asignar a estas los usuarios necesarios para los módulos de SISalud ya sean los Administradores Generales o cualquier otro tipo de usuario definido para aquellos módulos que puedan ser accedidos en este nivel de dirección. Utilizando el método Buscar_Total perteneciente a este componente se obtienen las unidades de salud que permitirán al Administrador General seleccionar la deseada.

### 4.2.3 Registro de Ubicación Geográfica (RU)

La necesidad de integración con el Registro de Ubicación Geográfica es similar a la del Registro de Unidades de Salud pero para los niveles de dirección provincial y municipal. Este registro gestiona el flujo de información relacionado con la creación, modificación o eliminación de las provincias, municipios, localidades, calles y manzanas. Haciendo uso del método ListarProvincias se obtiene un listado de todas las provincias con sus correspondientes municipios el cual será utilizado para el proceso de ubicación de los usuarios.

## 4.3 Modelo de Implementación

La implementación es el centro de las iteraciones durante la fase de construcción, aunque también se lleva a cabo la implementación durante la fase de elaboración, para crear la línea base de la arquitectura y durante la transición, para tratar los defectos tardíos como los encontrados en su distribución. El Modelo de Implementación describe cómo los elementos del Modelo de Diseño serán implementados en términos

de componentes describiendo además la organización de los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y el lenguaje o lenguajes de programación utilizados así como la dependencia que se establece entre estos componentes.

### 4.3.1 Diagramas de Componentes

Un componente es el empaquetamiento físico de los elementos de un modelo, como son las clases en el modelo del diseño. Alguno de sus estereotipos son <<file>>, <<library>> y <<table>>. Los componentes tienen relaciones de traza con los elementos del modelo de diseño que implementan. A continuación serán proporcionados los Diagramas de Componentes asociados a varios de los subsistemas de implementación identificados para el centro de Control.

Como ha sido mencionado anteriormente el Centro de Control es desarrollado sobre una arquitectura en tres capas, en tal sentido su estructuración en subsistemas de implementación es la siguiente:

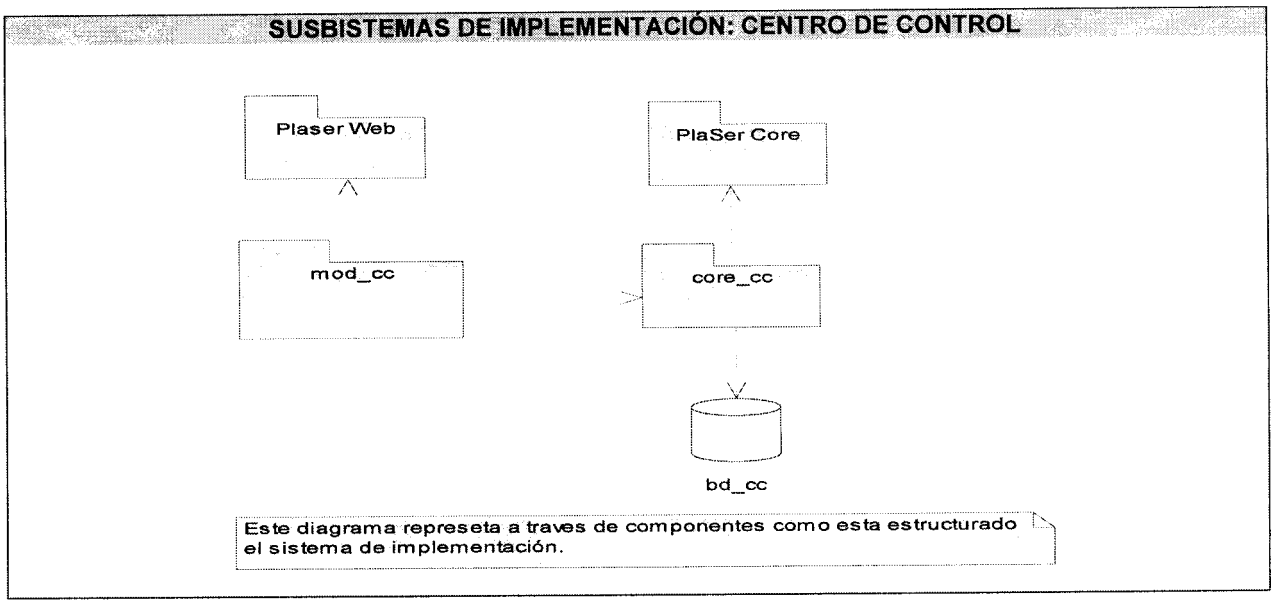



Figura 4.1 Estructura en subsistemas de implementación

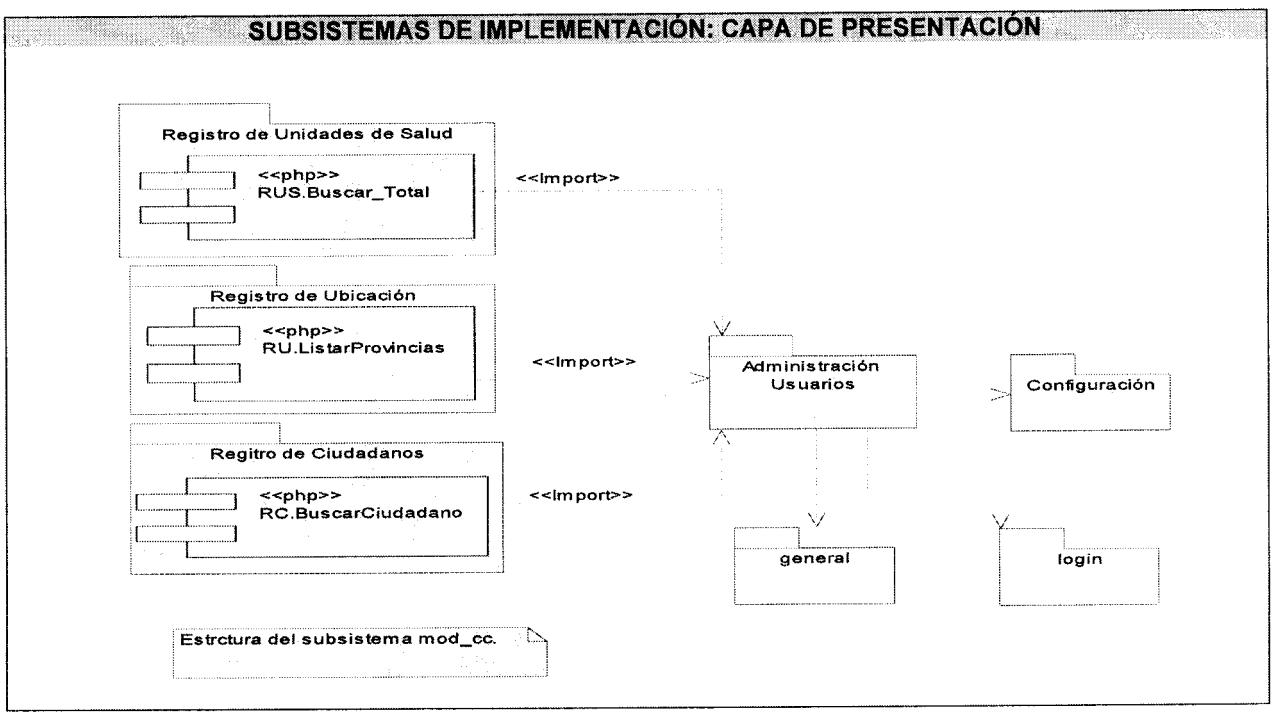



Figura 4.2 Estructura subsistema de implementación: *mod_cc*

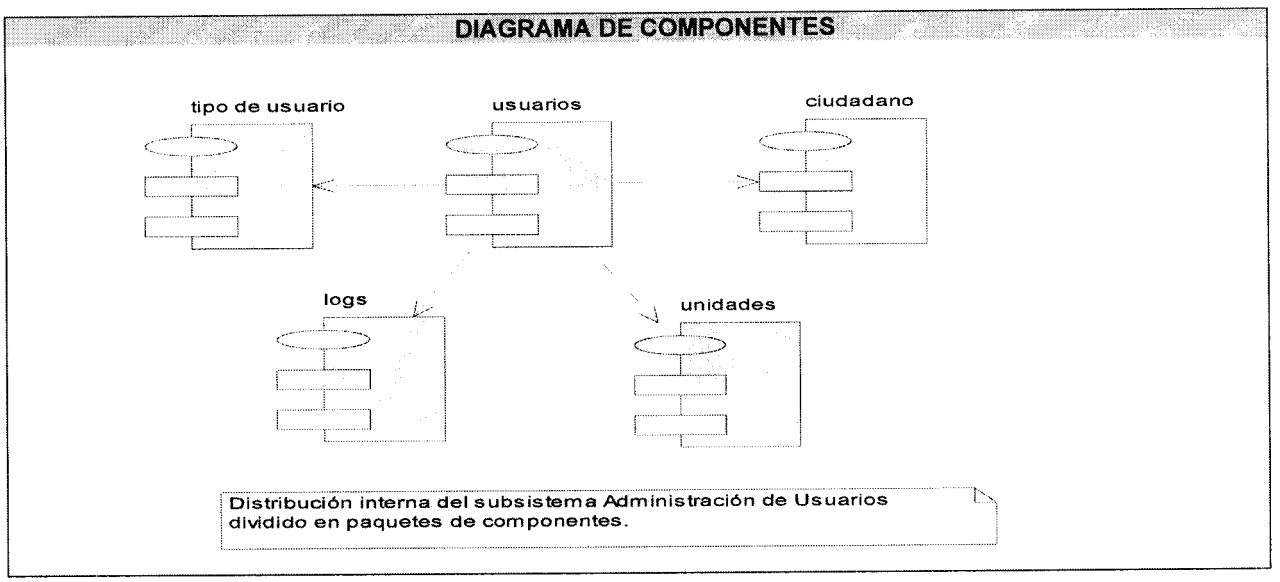



Figura 4.3 Organización en paquetes de componentes del subsistema: *Administración Usuarios (mod_cc)*

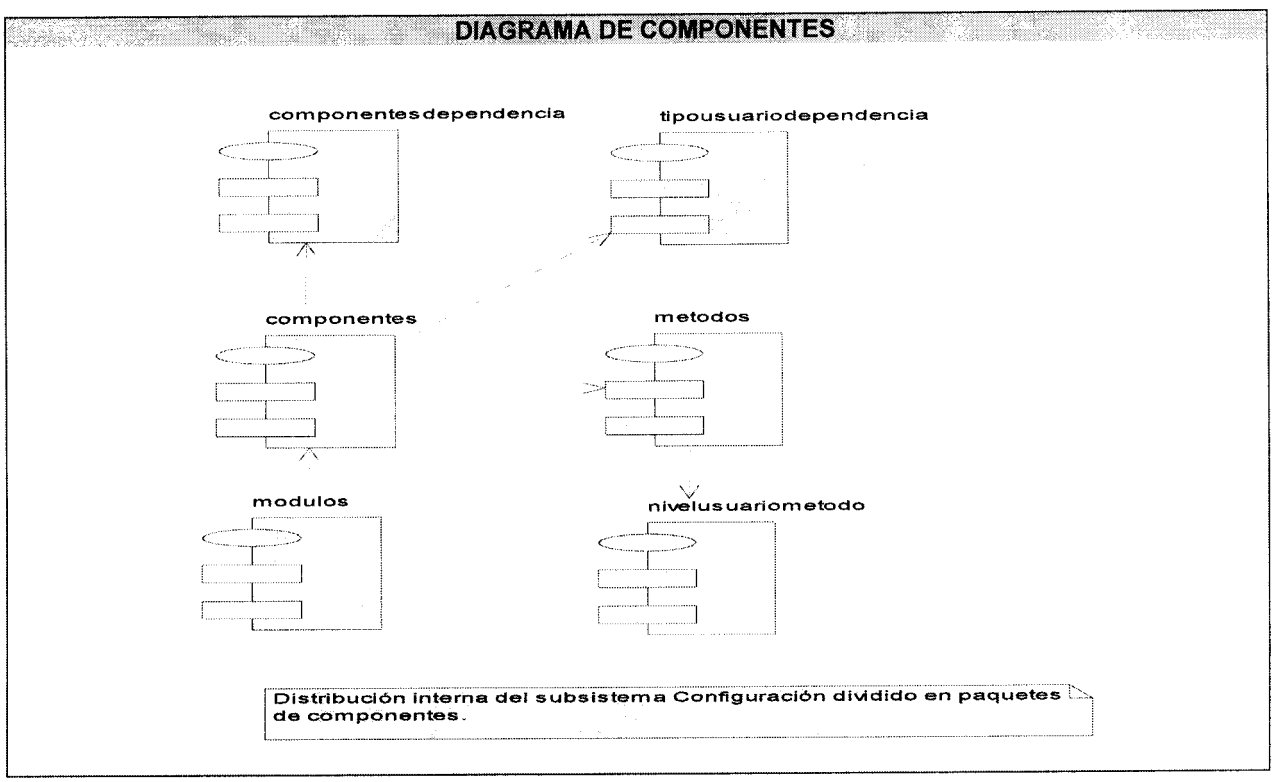



Figura 4.4 Organización en paquetes de componentes de subsistema: *Configuración (mod_cc)*

### 4.3.2 Diagrama de Despliegue

El Modelo de Despliegue es un modelo de objetos que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos de cómputo, por esta razón tal distribución tiene una influencia principal para las actividades de implementación. Cada nodo representa un recurso de cómputo, en el caso del Centro de Control, este será distribuido en tres servidores, el primero de ellos conteniendo la Capa de Presentación conectado a la Red Telemática de Salud INFOMED. El servidor de la Lógica de Negocio tendrá conectividad punto a punto únicamente con el servidor anterior y con el servidor de datos, todos estos nodos de procesamiento funcionarán con sistema operativo LINUX distribución White Box y la comunicación se realizará mediante SOAP utilizando el protocolo de transporte HTTPS.

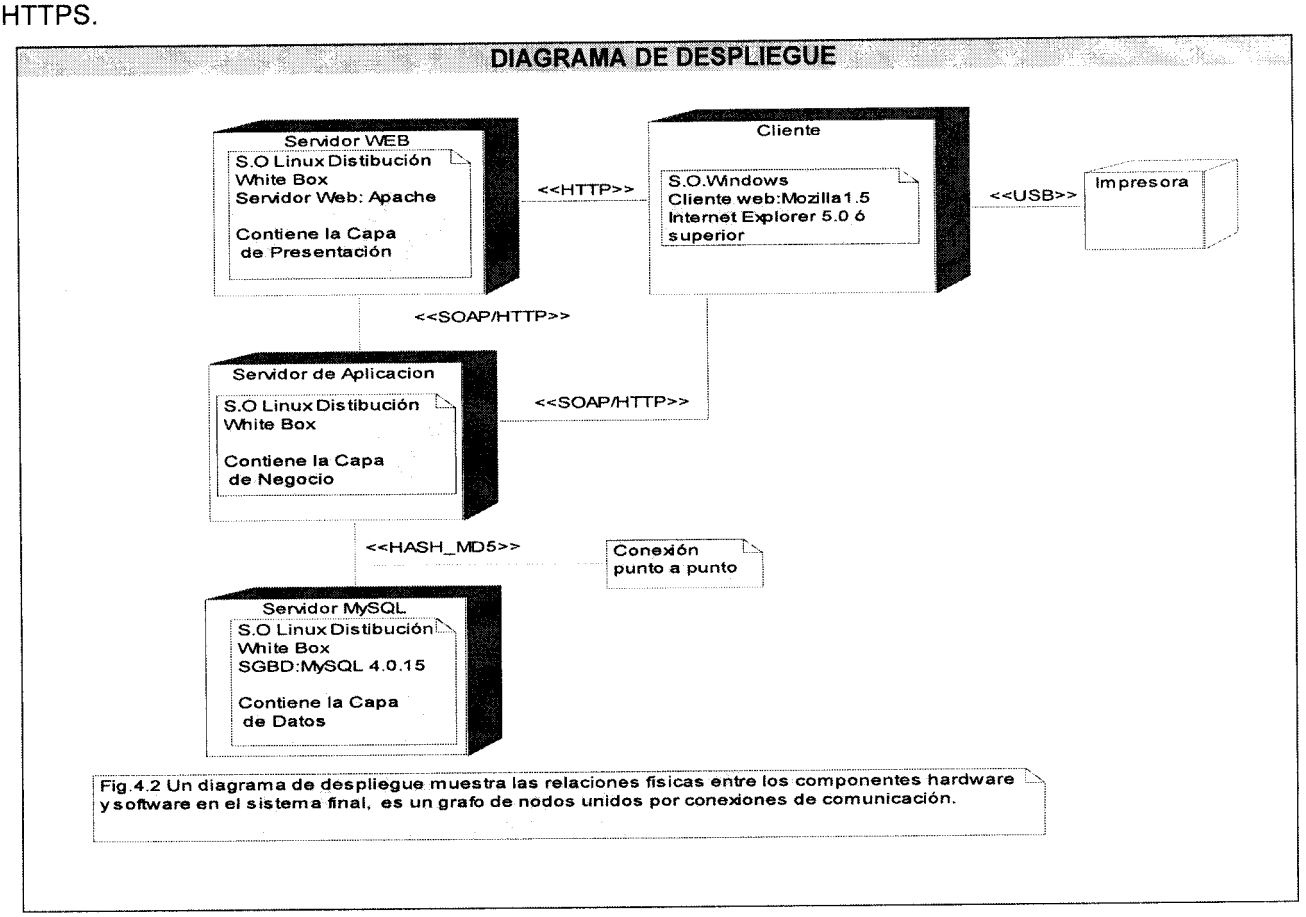



Figura 4.5 Diagrama de Despliegue

## 4.4 Mantenimiento de la integridad. Propuesta de solución

Para darle solución al segundo de los problemas planteados en la investigación, asociado al mantenimiento eficiente de la integridad referencial en el flujo de información distribuida, se identificó que este requerimiento debía ser tratado en el proceso de gestión de componentes. Para ello, se implementó el patrón de diseño Observador-Observable aprovechando las facilidades proporcionadas por la librería PlaSer. En tal sentido, fueron configuradas las restricciones de integridad que se establecían entre algunos de los componentes actualmente desplegados en el RIS, teniendo en cuenta que tales dependencias pueden ser restrictivas o en cascada. Esta configuración es utilizada para comprobar las dependencias de integridad cada vez que sea realizada una petición desde la Capa de Presentación.

Para acceder a los métodos de la Capa de Negocio desde la Capa de Presentación, en el Registro Informatizado de Salud, se implementa el patrón de diseño Proxy. El Proxy PlaSer o simplemente ProxPla, es un directorio que contiene los ficheros de configuración y despliegue confplaser_server.php y confplaser_client.php, el primero de ellos contiene todos los métodos que conforman los componentes y el segundo las URL de los componentes, hacia donde deben ser redireccionadas las peticiones realizadas desde la Capa de Presentación. A través del ProxPla es la única forma de acceder a los métodos del negocio, constituyendo el primer nivel de seguridad. En el ProxPla son controlados los derechos del usuario que está realizando la petición, autorizándose o no la ejecución de la misma.

Antes de comenzar a describir la solución implementada, es conveniente poner un ejemplo ilustrativo de la problemática asociada al mismo. El Registro de Enfermedades de Declaración Obligatoria (REDO) homologa las enfermedades de riesgo epidemiológico con el Registro de la Clasificación Internacional de Enfermedades y Problemas Relacionados con la Salud (RCIE).

Si se desea eliminar una determinada enfermedad en el componente RCIE la cual tiene asociada una enfermedad de declaración obligatoria en el componente REDO y con la cual fue diagnosticado un paciente y emitida la respectiva tarjeta de notificación, al ser eliminada esta enfermedad traería consigo inconsistencia y corrupción en la información gestionada por el REDO. Lo más lógico hubiese sido denegar esta petición por el ProxPla, es decir el REDO es un observador restrictivo del método

EliminarEnfermedad del RCIE y tal dependencia debía haber sido comprobada antes de redireccionarla hacia el componente RCIE.

Para realizar la implementación del patrón Observador-Observable fue necesario agregar un nuevo fichero de configuración al ProxPla, el confplaser_integrity.php, el cual tiene un conjunto de reglas de integridad referencial descritas en la forma que sigue, ***regla[]=observado.metodo.0/1.observado*** (***observado:*** *Acrónimo del componente Observado,* ***metodo:*** *Método por el cual se observa,* ***0/1:*** *Tipo de dependencia de integridad, 0 si es en cascada o 1 si es restrictiva,* ***observador:*** *Acrónimo del componente observador*). Además fue adicionada la clase *ParseXML* y la función *_integrity()* a la clase PLASER_Server, modificaciones realizadas a la librería PlaSer. *ParseXML* es una clase que recibe como parámetro un documento XML siendo este devuelto como un arreglo asociativo. La segunda modificación es la encargada de realizar los chequeos de integridad necesarios antes de redireccionar la petición al componente involucrado en la misma.

Cuando llega una petición desde la Capa de Presentación, es invocada la función *_integrity,* notificándose a los componentes que estén observando esta operación restrictivamente, siendo invocado un método del negocio nombrado *check,* el cual recibe como parámetros la operación y los datos involucrados en la misma. *Check* es un método booleano que devuelve falso si no se puede ejecutar la operación y verdadero en caso contrario, si el componente observador no tiene los datos a modificar registrados en su base de datos. Este método es implementado en dependencia de los negocios de los componentes observadores. Si al menos uno de los observadores involucrados devuelve falso, el ProxPla envía un mensaje de error hacia la Capa de Presentación, indicándosele al usuario que la petición no fue autorizada, siendo especificado el componente que denegó la misma por contener los datos a manipular.

En caso de que la respuesta de todos los métodos *check* de los componentes observadores sea verdadera, es redireccionada la petición hacia el componente observado y se notifica a los observadores cuya restricción es en cascada, para que actualicen su información en caso de tener registrada la que sufrió modificaciones en el componente observado. De esta manera son comprobadas todas las dependencias de integridad debidamente configuradas al registrar un nuevo componente en el Centro de Control. Esta solución implicará no necesitar tablas caché en los futuros componentes a desarrollar, así actualizar manualmente aquellas que se encuentren en los componentes actuales, proporcionándole a

SISalud un control estricto y total de las interdependencias entre los componentes distribuidos protegiéndose la información clínica contra estados corruptos e inconsistentes.

## 4.5 Impacto del despliegue del Centro de Control

Teniendo en cuenta que el Centro de Control es una versión del SAAA, componente que se encuentra en explotación desde el año 2004, el proceso de transición de este nuevo producto de software debe ser realizado mediante un Plan de Transición con tareas y ejecutores bien definidos. La fase de Transición se centra en implantar el producto en su entorno de operación, es decir proporcionar una versión para los usuarios, de modo tal que sean realizadas las pruebas de aceptación. Para cumplir con éxito los propósitos de esta fase, la cantidad de personas que la ejecutan puede ser similar a las utilizadas en la fase de construcción, aunque los objetivos y esfuerzos para el resultado a obtener sean diferentes.

El Plan de Transición para el Centro de Control debe contemplar una lista de riesgos, secuencia de pasos a seguir e interdependencias entre estos, así como los períodos de tiempo en los que el RIS debe estar de baja o fuera de servicio para la satisfactoria implantación de la nueva versión del SAAA. Teniendo en cuenta que se cuantifican por el actual componente SAAA, hasta el 5 de junio de 2007, 491 usuarios registrados, 8786 autenticaciones y un total de 450882 trazas de las cuales 413339 correspondían a la ejecución de alguno de los métodos de los componentes actualmente desplegados implicaría afectaciones a los usuarios activos al no poder utilizar las funcionalidades proporcionadas por los módulos actuales del RIS, por lo que este proceso complejo debe ser bien planificado y ejecutado.

Estos elementos deben ser considerados para la realización de un análisis y estudio donde sean definidas un conjunto de medidas que minimicen este impacto, algunas ideas o tareas a considerar son las siguientes:

1. Realizar salva del actual módulo de Administración del RIS incluyendo su base de datos, esta información no puede eliminarse debido a que contiene todas las operaciones o eventos de interacción de los usuarios con la información almacenada por los componentes del RIS.

2. Asignar a todos los usuarios existentes su ciudadano asociado, previendo que el mismo se encuentre registrado en el Registro de Ciudadanos (RC).

3. Registrar los métodos, componentes y módulos teniendo en cuenta toda la información adicional que recoge el Centro de Control, garantizando así el correcto funcionamiento de los requerimientos y funcionalidades del nuevo módulo de administración.

4. Realizar levantamiento de las dependencias de integridad que se establecen entre los componentes ya liberados e implementar los métodos check y update, en dependencia de las necesidades de integración, para los componentes liberados que requieran la comprobación de sus interdependencias.

5. Desarrollar un adecuado proceso de capacitación a las personas que interactuarán con el Centro de Control, enfatizando en las nuevas funcionalidades proporcionadas por este.

6. Confeccionar una plantilla que contenga todos los datos y secuencia de pasos necesarios, para que el Administrador de Configuración realice un despliegue exitoso de los módulos a integrarse en SISalud.

7. Borrar físicamente aquellos usuarios y trazas que hayan sido eliminados en forma lógica del sistema. Proceso que será realizado periódicamente en forma manual por los administradores del sistema, proporcionando así al servidor de bases de datos mayor velocidad en los tiempos de respuesta, al no tener almacenada información innecesaria.

## 4.6 Conclusiones

Una vez concluido el proceso de implementación del Centro de Control, se ha obtenido un producto con la totalidad de las funcionalidades previstas para su correcto funcionamiento. Se ha descrito además, la solución al mantenimiento de la integridad referencial en el flujo de información que es intercambiada entre los componentes desplegados en el Sistema de Información para la Salud. Se les brinda a los administradores del sistema una solución mucho más funcional que la anterior versión del Módulo de

Administración, incorporándosele a esta nueva versión un grupo de operaciones fundamentalmente orientadas hacia una eficiente gestión de usuarios.

# CONCLUSIONES

- Ha sido realizado un estudio de las tendencias y tecnologías actuales para el desarrollo del Centro de Control, asimilando la arquitectura definida por el MINSAP, el negocio actual del componente SAAA del RIS y la metodología de desarrollo definida por la empresa SOFTEL.

- Fue desarrollada una base de datos para el Centro Control, capaz de organizar y almacenar de manera eficiente la información que es manipulada por el sistema, soportando integridad referencial y transacciones.

- Se ha desarrollado un producto de software cualitativamente superior para la administración de usuarios y asignación de privilegios que lo diferencia notablemente de su antecesor el componente SAAA del Registro Informatizado de Salud (RIS).

- El Centro de Control proveerá una adecuada comprobación de las condiciones restrictivas de dependencia, que se establezcan entre los componentes distribuidos integrados en el Sistema de Información para la Salud (SISalud), garantizando la integridad en el flujo de información.

- La información clínica gestionada por el Sistema de Información para la Salud (SISalud) será objeto de cuidadosa protección contra estados corruptos e inconsistentes, teniendo acceso a ella sólo aquellas personas con privilegios asignados.

- Durante el proceso de transición del sistema informático obtenido tendrán que llevarse a cabo importantes acciones de capacitación a los usuarios finales para su exitoso despliegue.

# RECOMENDACIONES

- Realizar un análisis para definir medidas que garanticen un menor impacto al desplegar la versión propuesta del SAAA, el cual debe contemplar una lista de riesgos, secuencia de pasos a seguir e interdependencias entre estos; así como períodos de tiempo en los que el RIS debe estar de baja o fuera de servicio para su implantación.

- Utilizar el Centro de Control como un Bus de Servicios Empresariales (ESB) para las aplicaciones que se desarrollen en el dominio de salud pública en Cuba.

- Definir los requerimientos que permitan adicionar funcionalidades al Centro de Control, para realizar auditoría sobre la trazabilidad histórica registrada por el sistema de aquellos usuarios que fueron eliminados lógicamente.

# RFERENCIAS BIBLIOGRÁFICAS

[1]. Valdés Menéndez, Ramiro. Ministro de Informática y las Comunicaciones. *Discurso pronunciado en el Acto Inaugural del XII Convención y Expo Internacional Informática 2007*. La Habana, Cuba.

[2]. Sánchez Rodríguez, Alfredo. [Entrevista con]. *Antecedentes del proceso de integración de sistemas informáticos en el sector de la salud*. 2007.

[3]. Marín Díaz, Miguel E. Fundamentos *del Sistema de Salud Pública en Cuba para estudiantes de Informática*. La Habana. Cuba. 2006. p 3.

[4]. Ramírez Márquez, Abelardo Dr; Castell-Florit Serrate, Pastor Dr; Mesa, Guillermo Dr. *El Sistema Nacional de Salud de Cuba. Escuela Nacional de Salud Pública (ENSAP)*. La Habana, Cuba. 2003. Disponible en: http://www.sld.cu/galerias/doc/sitios/infodir//09_el_sistema_nacional_de_salud.doc

[5]. Delgado Ramos, Ariel. *Presentación Informatización del Sistema Nacional de Salud*. Dirección Nacional de Registros Médicos y Estadísticas de Salud. Ministerio de Salud Pública. La Habana. Cuba. 2006.

[6]. Durand Llamos, Catherine; Reyna Soler, Luis Mariano. *Centro de Control para el Sistema de Información para la Salud*. Trabajo de Diploma para optar por el Título de Ingeniero Informático. Instituto Superior Politécnico José Antonio Echeverría. La Habana, Cuba. Julio 2005. p 9.

[7]. Cabrera Hernández, Mirna et al. *Propuesta de Esquema del Sistema de Información para la Salud (SISalud)*. La Habana. Cuba. 2006.

[8]. Pressman, Roger S. *Ingeniería de Software, un enfoque práctico*. Parte 1. La Habana, Cuba. Editorial Félix Varela. 2005

[9]. Gudiño Fleites, Pedro. *Tutorial de Sistemas Distribuidos I.* Departamento de Sistemas y Computación. Instituto Tecnológico de Colima. México. 2004. Disponible en: http://www.itcolima.edu.mx/profesores/tutoriales/sistemas_distribuidos_I/index.htm.

[10]. Maldonado Segura, José Alberto et al. *Tecnologías de la información al servicio de la historia clínica electrónica*. Sociedad Española de Informática para la Salud. España. 2002. p 154.

[11]. SOFTEL. *Documento sobre la Arquitectura de Software a emplear en los componentes del Sistema de Información para la Salud.* La Habana. Cuba. 2006.

[12]. Parra, José David. *Hacia una arquitectura empresarial basada en componentes*. 2005. Disponible en: http://www.microsoft.com/spanish/msdn/comunidad/mtj.net/voices/art143.asp

[13]. Ídem a la referencia 10.

[14]. Reynoso, Carlos; Kicillof, Nicolás. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* Universidad de Buenos Aires 2004.
Disponible en: http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/style.asp#10

[15]. Larman, Craig. *Patrones Grasp*. 2005. Disponible en: http://jorgesaavedra.wordpress.com.

[16]. Ídem a la referencia 14

[17]. Ídem a la referencia 14

[18].Legido Hernández, Marcos. *Las variables en JavaScript.* 2005. Disponible en: http://www.wikilearning.com/caracteristicas_de_javascript-wkccp-3529-3.htm.

[19]. Rodas Hinostroza, Raúl. *Usuarios y Grupos en Linux*. Octubre 2006. Disponible en: http://www.linuxcentro.net/linux/staticpages/index.php?page=CaracteristicasPHP

[20]. Ayala Montero, Ramón. XML *Iniciación y Referencia*. Madrid, España. McGRAW-HILL/ Interamericana de España. 2001.p 9,10.

[21]. Minnick, Chris; Valentine, Chelsea. *XHTML Serie Práctica*. Nueva York, Estados Unidos. 2000. p 3, 4, 5.

[22]. Minnick, Chris; Valentine, Chelsea. *XHTML Serie Práctica*. Nueva York, Estados Unidos. 2000. p 7.

[23]. Gallego Vázquez, José Antonio. *Desarrollo Web con PHP y MySQL*. Madrid, España. Ediciones Anaya Multimedia. 2003. p 20.

[24]. Jacobson, Ivar; Booch, Grady; Rumbaugh, James. *El Proceso Unificado de Desarrollo de Software*. La Habana, Cuba. Editorial Félix Varela. 2004. p 4, 5, 6, 7.

[25]. Larman, Craig. *UML y Patrones: Introducción al análisis y diseño orientado a objetos*. La Habana, Cuba. Editorial Félix Varela. 2004. p 16,17

[26]. Franco Navarro, José Angel. *UML en acción. Modelando Aplicaciones Web*. Instituto Superior Politécnico José Antonio Echeverría. La Habana, Cuba. Mayo 2005.

[27]. Idem a la referencia 26.

# BIBLIOGRAFÍA

Dirección de Informatización. *Arquitectura para los sistemas que conforman la intranet universitaria.* Universidad de las Ciencias Informáticas. La Habana, Cuba. Mayo 2007.

Durand Llamos, Catherine; Reyna Soler, Luis Mariano. *Centro de Control para el Sistema de Información para la Salud.* Trabajo de Diploma para optar por el Título de Ingeniero Informático. Instituto Superior Politécnico José Antonio Echeverría. La Habana, Cuba. Julio 2005.

Ferm, Fredrik. *The what, why and how is a subsystem. The Rational edge, e-zine for the rational community.* Junio 2003. Disponible en:http://www.therationaledge.com/content/jun_03/t_subsystem_ff.jsp

Franco Navarro, José Angel. *UML en acción. Modelando Aplicaciones Web.* Instituto Superior Politécnico José Antonio Echeverría. La Habana, Cuba. Mayo 2005.

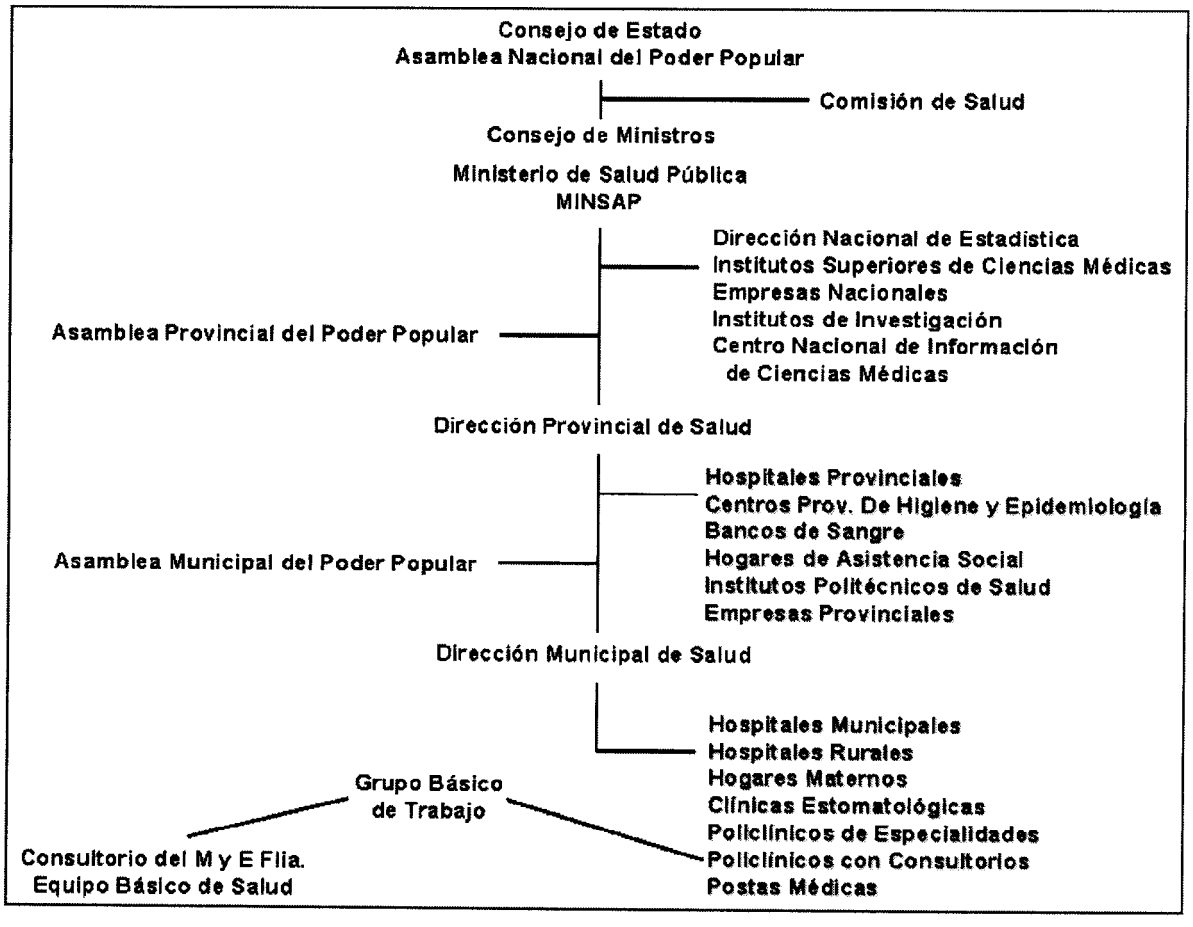
López Medina, Julio E; Mosquera, Luis J. *Arquitectura de integración para el sector financiero. XXV Salón de Informática "Arquitecturas empresariales de software".* Bogotá, Colombia. 28-01-2005.

Pressman, Roger S. *Ingeniería de Software, un enfoque práctico.* Parte 1. La Habana, Cuba. Editorial Félix Varela. 2005.

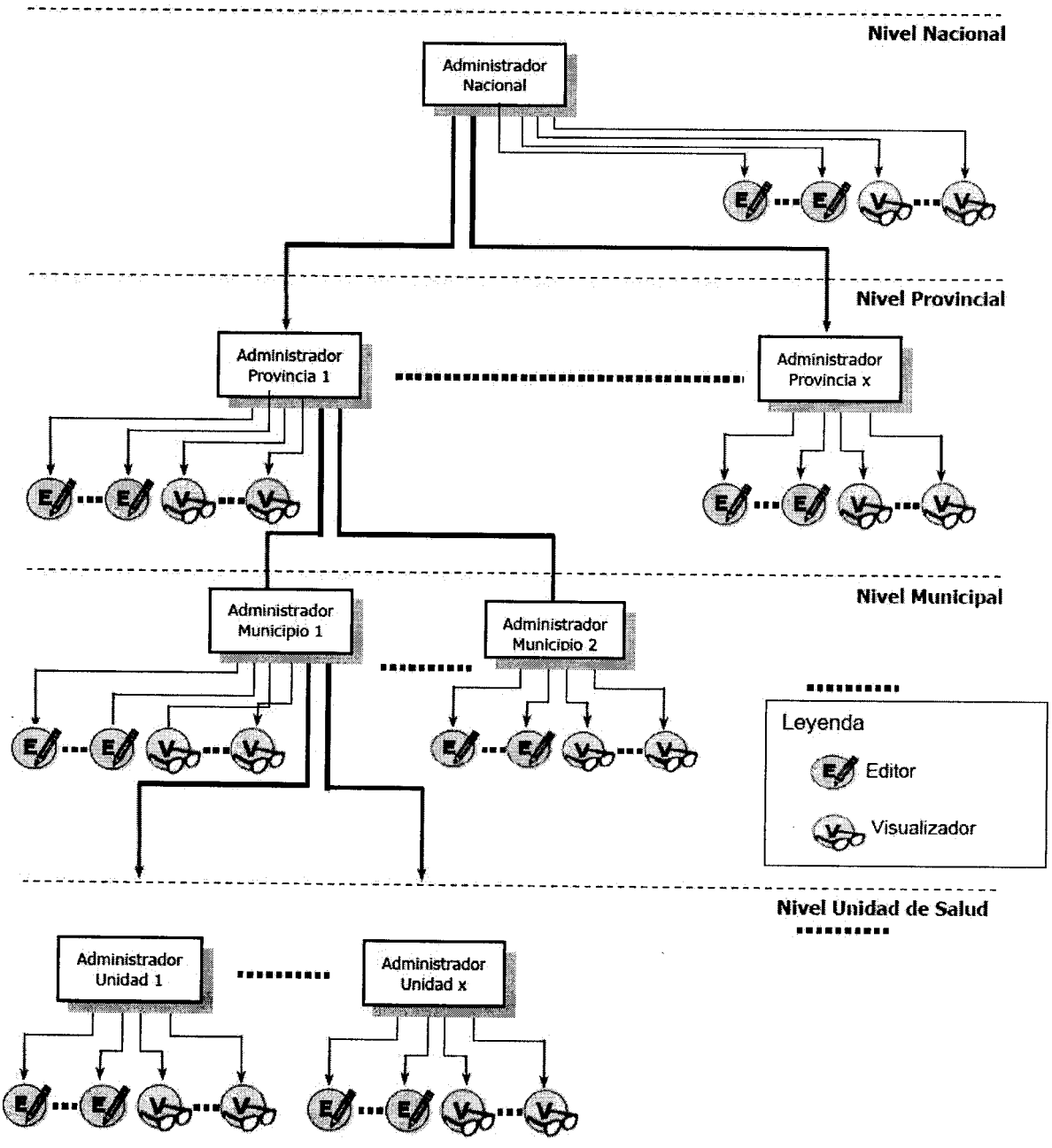
Schmuller, Joseph. *Aprendiendo UML en 24 horas. Macmillan Computer Publishing* .Indiana, Estados Unidos. 2000.

# ANEXOS

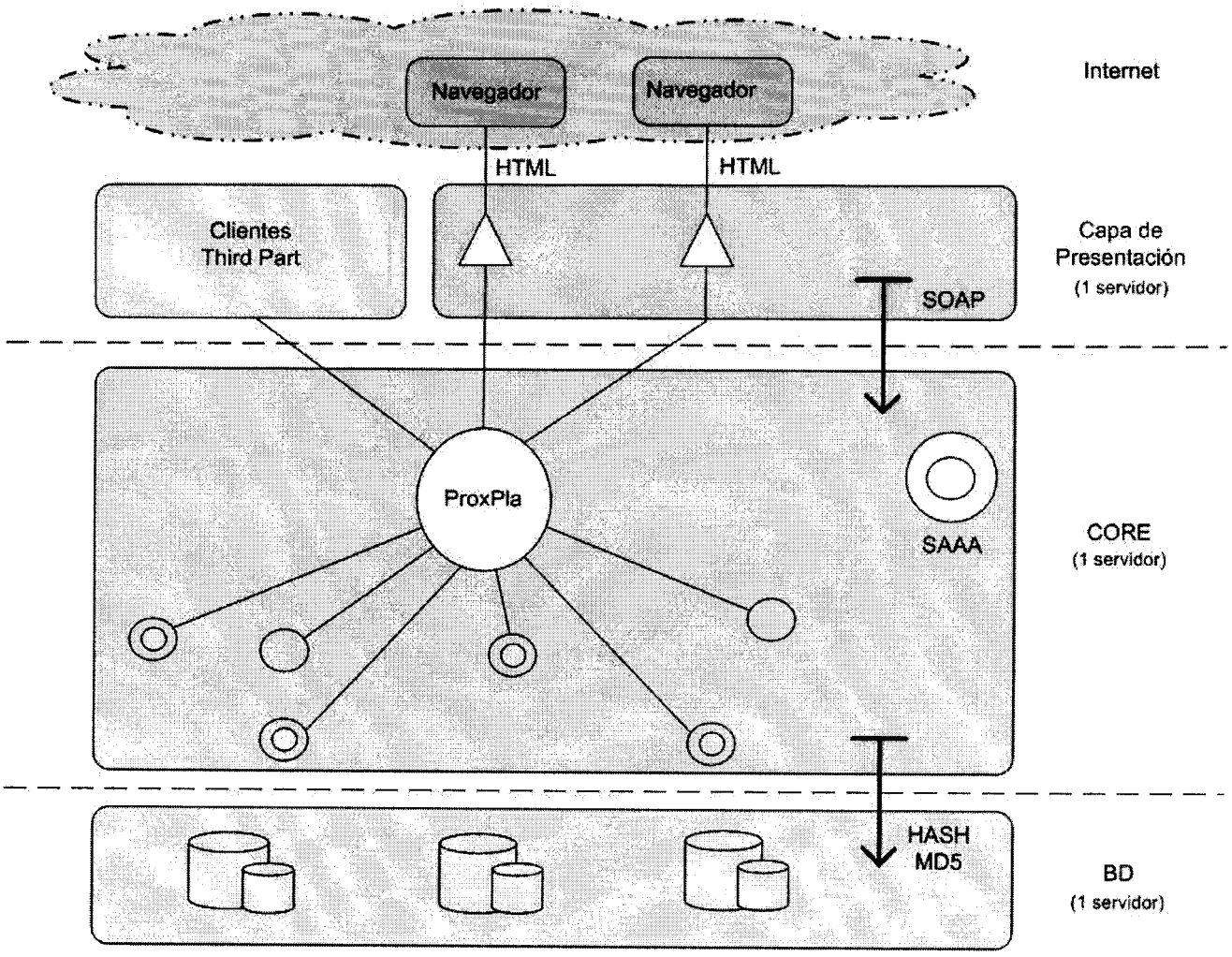
## ANEXO I: ORGANIGRAMA DEL SISTEMA NACIONAL DE SALUD CUBANO



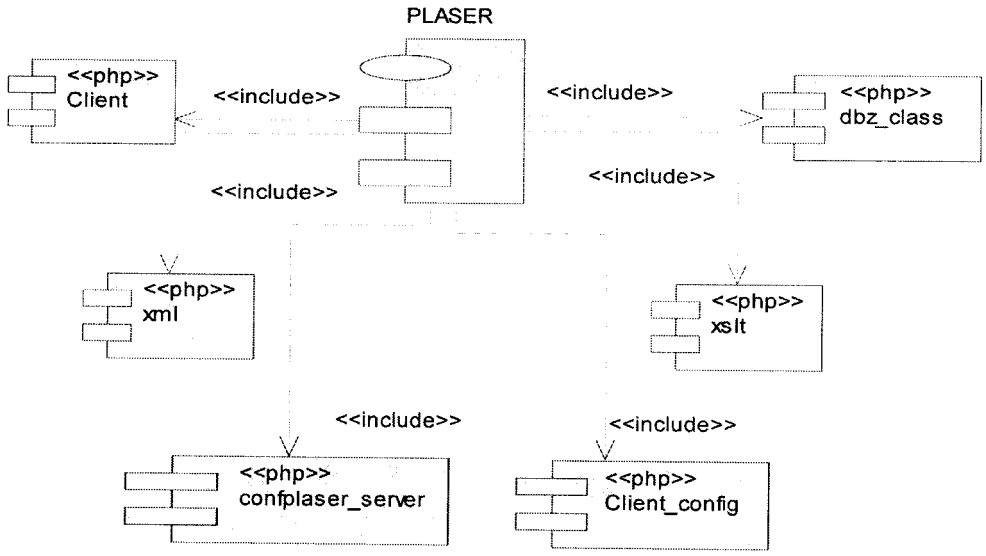
## ANEXO II: ÁRBOL JERÁRQUICO DE ADMINISTRACIÓN DEL RIS



## ANEXO III: ARQUITECTURA DE TRES CAPAS PARA EL RIS

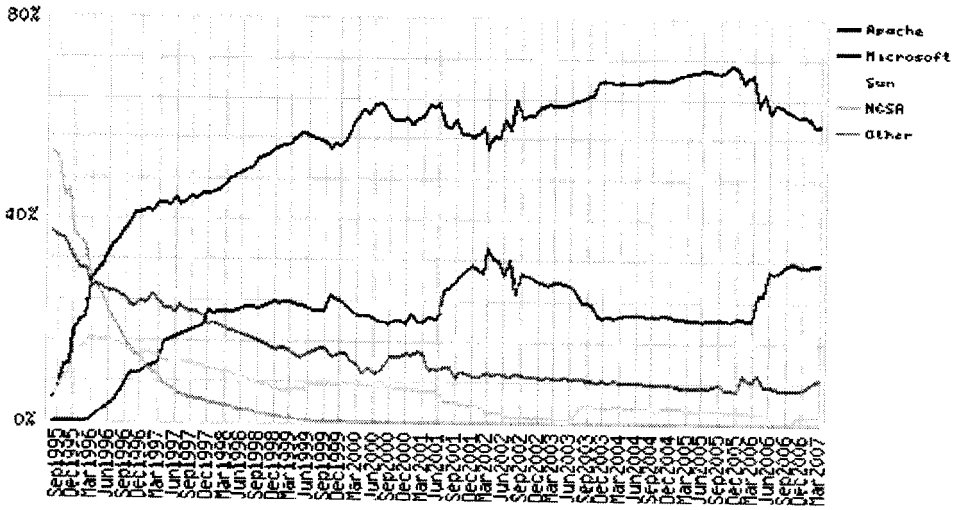


## ANEXO IV: DIAGRAMA DE COMPONENTES LIBRERÍA PlaSer

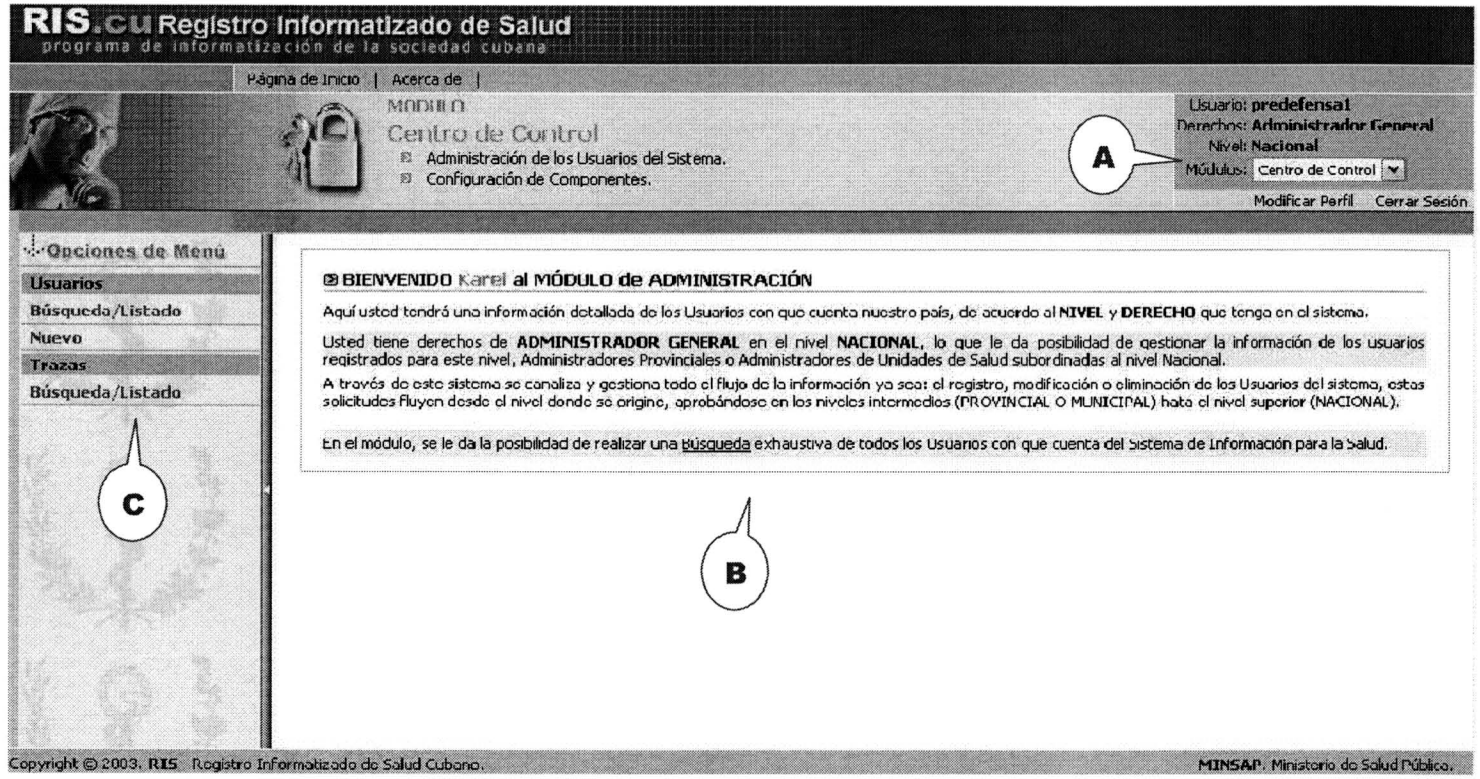



Un diagrama de componentes muestra un conjunto de componentes y sus relaciones, se utilizan para describir la vista de implementación estática de un sistema.

## ANEXO V: POPULARIDAD DEL SERVIDOR APACHE. FUENTE: (www.netcraft.com, Marzo 2007)



## ANEXO VI: PÁGINA PRINCIPAL DEL CENTRO DE CONTROL (Administrador General)

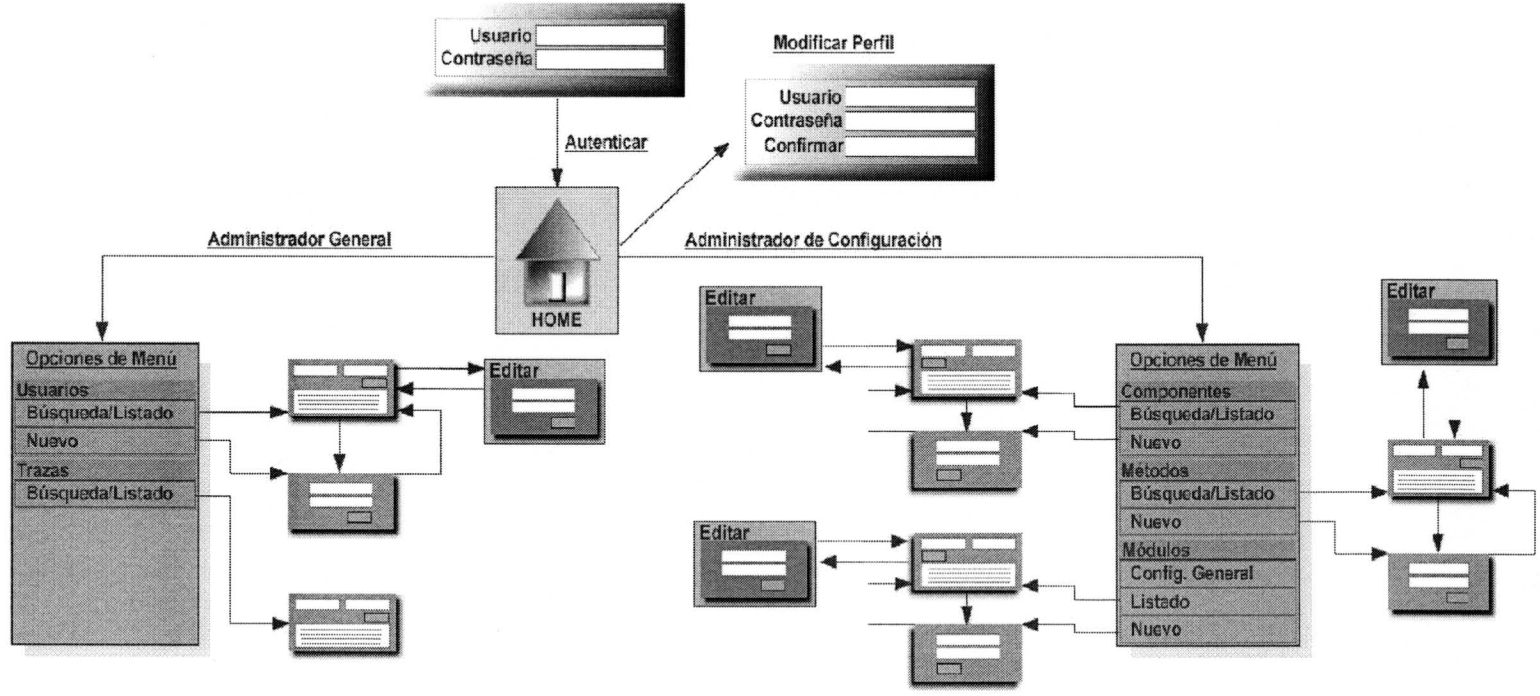



**A:** Región que contiene información del usuario autenticado.

**B:** Región editable, en este caso contiene mensaje de Bienvenida para el usuario, indicándole los privilegios con los cuales fue autenticado.

**C:** Menú XML dinámico que se crea en dependencia de los privilegios del tipo de usuario autenticado.

## ANEXO VII: MAPA DE NAVEGACIÓN DEL CENTRO DE CONTROL



# GLOSARIO DE TÉRMINOS

***Acoplamiento:*** Es una medida de la interdependencia relativa entre los componentes. Minimizando el acoplamiento se evita el "efecto de onda" en la propagación de errores.

***Aplicación Web:*** Es una aplicación informática que los usuarios utilizan accediendo a un servidor Web a través de un navegador o browser. Estas son muy son populares debido a la habilidad para actualizar y mantener la información manipulada sin distribuir e instalar el software en miles de potenciales clientes.

***Autenticación:*** Mecanismos del sistema de información para poder identificar a los usuarios que acceden a sus recursos, asegurando la integridad y autenticidad de los datos.

***Capa de Datos:*** Es donde residen los datos. Está formada por uno o más Sistemas Gestores de Bases de Datos (SGBD) que realiza todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio.

***Capa de Negocio:*** Es donde residen los programas que son ejecutados mediante peticiones del usuario y enviando las respuestas tras el proceso. Se denomina capa de negocio (e incluso de lógica del negocio) pues es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos almacenar o recuperar datos de él.

***Capa de Presentación:*** Es la que ve el usuario, presenta el sistema al usuario, le comunica la información y captura la información del usuario dando un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio.

***CENTERSOFT:*** Empresa cubana que se dedicaba a la exportación y distribución de software fusionada en el año 2003 con la Empresa de Soluciones Informáticas SOFTEL.

***Cohesión:*** Es la medida de la medida de la fuerza relativa funcional de un componente. Un componente con cohesión realiza una sola tarea dentro de un procedimiento de software, requiriendo poca interacción con otros componentes.

***Componente:*** Es un conjunto o bloque de software que proporciona por sí mismo o en conjunción con otros componentes una función o servicio único y puede ser reutilizado para construir diversos sistemas.

***Componente Base:*** Componente de un módulo encargado de la integración de los datos distribuidos y responsable de la creación del documento XML que será enviada a la capa de presentación para ser visualizados al cliente.

***HTTP:*** *HyperText Transfer Protocol* es el protocolo usado en cada transacción de la Web. El hipertexto es el contenido de las páginas Web y este protocolo de transferencia es el sistema mediante el cual se envían las peticiones de acceso a una página y la respuesta con el contenido.

***HTTPS:*** Es la versión segura del protocolo HTTP. El sistema HTTPS utiliza un cifrado basado en las Secure Socket Layers (SSL) para crear un canal cifrado cuyo nivel depende del servidor remoto y del navegador utilizado por el cliente, más apropiado para el tráfico de información sensible que el protocolo HTTP. Cabe mencionar que el uso del protocolo HTTPS no impide que se pueda utilizar HTTP.

***Información Clínica:*** Informe detallado del diagnóstico, tratamiento y seguimiento de un paciente en particular. La información clínica también contiene alguna información demográfica acerca del paciente (por ejemplo, edad, sexo, origen étnico).

***Informatizar:*** Proceso de aplicar sistemas o equipos informáticos al tratamiento de la información.

***INFOMED:*** Red Telemática del Ministerio de Salud Pública (MINSAP) de Cuba.

***Integridad Referencial:*** Es un sistema de reglas deseadas que utilizan la mayoría de las bases de datos relacionales para asegurarse que los registros de tablas relacionadas son válidos y que no se borren o cambien datos relacionados de forma accidental produciendo errores de integridad.

***PlaSer:*** Abreviatura de *Plataforma de Servicios*, librería utilizada para el desarrollo de componentes utilizando el paradigma XML-Web Services. Este fue creado en el año 2003 por la empresa SOFTEL, para facilitar y agilizar el proceso de implementación y la homogeneidad de los módulos del Registro Informatizado de Salud (RIS). La versión actual de PlaSer sólo soporta como llamada RPC el protocolo SOAP.

***Policlínico:*** Unidad de salud donde se brindan servicios médicos a una población geográficamente determinada perteneciente al nivel asistencial de Atención Primaria de Salud.

***Problema de salud:*** Motivo por el cual un paciente acude en busca de atención médica a una unidad asistencial de cualquier nivel de atención médica.

***Rehabilitación:*** Rama de la medicina encargada de ayudar al paciente discapacitado a recuperar o mejorar las funciones perdidas para su reincorporación como miembro útil a la sociedad.

***RPC:*** Remote Procedure Call es un protocolo que permite a un programa de ordenador ejecutar código en otra máquina remota sin tener que preocuparse por las comunicaciones entre ambos. El protocolo fue propuesto inicialmente por Sun Microsystems como un gran avance sobre los socket usados hasta ese momento. De esta manera el programador no tiene que estar pendiente de las comunicaciones, estando éstas encapsuladas dentro de RPC.

***Seguridad:*** Mecanismo que previene algún riesgo o asegura el buen funcionamiento de algún ente, previniendo que la misma falle.

***Servicio:*** Unidad de software que encapsula alguna funcionalidad de negocio y proporciona estas a otros servicios a través de interfaces públicas bien definidas.

***Servidor:*** Son ordenadores que actúan como "almacenes" de información. Esta información es solicitada por los ordenadores cliente y el servidor responde a tales peticiones devolviendo los datos solicitados.

***SOAP:*** Siglas de Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. SOAP es uno de los protocolos utilizados en los Servicios Web.

***Transacción:*** Unidad lógica de procesamiento y ejecución de un programa que incluye operaciones de acceso a la base de datos. Si falla una de las operaciones se deshacen las restantes

***Token:*** Es un identificador o palabra reservada que constituye una unidad léxica, en la cual se agrupan un conjunto de lexemas que coincidan con un mismo patrón.

***Unidad de Salud:*** Centro de trabajo del Ministerio de Salud Pública (MINSAP).

***URL*:** Siglas de *Uniform Resource Locator*. Representa el sistema de direcciones usado en la Web y otros recursos de Internet. La URL contiene información sobre el método de acceso, el servidor al que se accede y la dirección del fichero.