Universidad de las Ciencias Informáticas
Facultad 7

Título: Diseño y Servicios Web para el Registro de Áreas de Salud de la Atención Primaria del Sistema de Información para la Salud.

Trabajo de Diploma para optar por el título de
Ingeniero Informático

Autores: Danieski Rodriguez Osoria
Alexander Luisovich Paneque Meschenkov
Yadisnier Roberto Sarmiento Borrero

Tutores: Lic. Caridad Guzmán Vitón
Lic. Yamilka Gómez León

Junio, 2007

Declaración de Autoría

Declaramos que somos los únicos autores de este trabajo en el cual he utilizado información y documentación que es propiedad de la empresa SOFTEL lo cual está sujeto a un acuerdo de confidencialidad. Pongo a disposición de la Universidad (UCI) todo aquello que no comprometa dicho acuerdo.

Para que así conste firmamos la presente a los _____ días del mes de_____ del año_____

_____ _____

Danieski Rodriguez Osoria Yadisnier Roberto Sarmiento Borrero

_____

Alexander Luisovich Paneque Meschenkov

_____ _____

Lic. Caridad Guzmán Vitón Lic. Yamilka Gómez León

Datos de Contacto

Yamilka Gómez León - yamilkagl@uci.cu:

Profesor graduado de Licenciatura en Ciencias de la Computación en el año 2004 en la UCLV. Ha impartido las asignaturas Programación 2, Sistemas de Bases de Datos y Gestión de Software. Posee la categoría docente de Instructor y cursa la maestría de Ciencias de la Computación en la UCI. Forma parte del proyecto de Atención Primaria de Salud de la facultad 7.

Caridad Guzmán Vitón - **cary@softel.cu:**
Especialista A en Sistemas Organizativos e Informativos de la empresa Softel. Graduada en Cibernética Matemática en el año 1986 en la Universidad de La Habana. Posee 19 años de experiencia en el desarrollo de software desempeñando diferentes roles. Ha trabajado en las empresas Textilera Rubén Martínez Villena (1987 – 1993) y Softel (1993 hasta la actualidad). Trabajó en la empresa mixta BC BIOCON Internacional S.A. España (1996-2001). Ha pasado varios cursos de superación y postgrado. Ha sido tutora de tesis.

El hombre que vive siempre en torno de la verdad, posee la mayor de las virtudes y la más grande de las riquezas

Ligia García

El futuro tiene muchos nombres. Para los débiles es lo inalcanzable. Para los temerosos, lo desconocido. Para los valientes es la oportunidad.

Victor Hugo

Agradecimientos

Agradecemos a todas las personas que nos han brindado su apoyo en la realización de esta investigación.

A nuestras tutoras Cary y Yamilka, a nuestra jefa de proyecto Mirna, por apoyarnos y haber depositado toda su confianza en nuestras manos para desarrollar esta tarea.

A nuestra familia, por su apoyo y dedicación.

A nuestros profesores, por contribuir a nuestra formación profesional a lo largo de estos años.

A nuestros amigos, por enseñarnos el valor de la confianza y la necesidad de crecerse día tras día.

Dedicatoria

A mi padre por ser mi ejemplo a seguir y tener la confianza en mí para salir adelante, especialmente le agradezco a Mirna, Cary, Idesis, Danieski, Yadisnier, Karel, a mis compañeros de grupo, a mi familia, a los profesores de la UCI y a todos los que me han guiado para llegar a ser Ingeniero a todos muchísimas gracias.

Alexander Paneque

A mi madre por darme su apoyo y confianza para superar los obstáculos, agradecerle a Cary, Mirna, Danieski, Paneque, a mi familia, a mis compañeros de grupo y a todos los que me han alentado para llegar a ser ingeniero a todos gracias.

Yadisnier Sarmiento

Dedico este trabajo a mi madre, a mi tío Agustín, a Vidalina, a mi tutora Cary, a mi jefa de proyecto Mirna, a mis compañeros de equipo Yadisnier y Alexander, a toda mi familia, a mis profesores y amigos por su apoyo de toda la vida y por la confianza que siempre han tenido en mí.

Danieski Rodriguez

Resumen

El presente trabajo, persigue diseñar y desarrollar la aplicación Web, Registro de Áreas de Salud (RAS), que gestione la información correspondiente a las Áreas de Salud. Surge porque en el país no existe un registro centralizado que gestione la información de las Áreas de Salud y que pueda ser consultado en los diferentes niveles de dirección de salud, facilitando el control de la información y la toma de decisiones.
Es un módulo del Registro Informatizado de Salud (RIS) que pertenece al Proyecto Sistema de Información para la Salud (SISalud) que informatizará los procesos del Sistema Nacional de Salud.

En el sistema se modelaron los flujos de trabajo diseño e implementación utilizando la metodología del Proceso Unificado de Desarrollo (RUP) y se implementó el RAS, siguiendo la arquitectura y tecnologías propuestas por el MINSAP.

El RAS informatizará los procesos de gestión de la información de las áreas: Grupos Básicos de Trabajo (GBT) y Equipos Básicos de Salud (EBS) que la conforman; poblaciones que atienden, formadas por viviendas o centros laborales y los locales de consultas o de viviendas. Además; se configura para cada área, las localidades que atienden, sus hospitales base, departamentos y servicios que brinda a la población. Permitiendo así que se disponga de un registro que permita el control de la información en los diferentes niveles de dirección y brinde información a otros componentes de SISalud.

Tabla de Contenidos

Introducción

Cuba enfrenta hoy el enorme reto de informatizar la sociedad cubana con vistas a integrarse a la infraestructura global de la información, así como optimizar el uso de las Nuevas Tecnologías de la Información y las Comunicaciones (TIC). El Ministerio de Salud Pública de Cuba (MINSAP) ha definido la informatización del sector de la salud, como los procesos cuyos procedimientos se enmarcan en el concepto de la informatización social, en busca de la optimización de los servicios de salud que se brindan a la población; mayor productividad y competencia en el desempeño de sus profesionales y técnicos, control en la administración de sus recursos, eficacia y eficiencia en su gerencia.[1]

El Sistema Nacional de Salud (SNS) de Cuba está organizado por tres niveles de atención. El nivel de Atención Primaria de Salud (APS) es el que debe dar solución a la mayor cantidad de problemas de salud de la población, sus actividades se realizan en cualquier unidad del Sistema Nacional de Salud y están relacionados fundamentalmente con las que se realizan en el Policlínico y en los Consultorios del Médico de la Familia, los Hospitales Rurales, los Dispensarios y las Postas Médicas. La Atención Secundaria que es la brindada por los Hospitales principalmente en cada una de las regiones, y la Atención Terciaria que es la que brindan los Hospitales Especializados.

La Salud Pública en nuestro país no está informatizada de forma integrada para los procesos actuales de recogida, procesamiento y transmisión de información en el Sistema Nacional de Salud. Existe una demora a la hora de contar con esta información a nivel nacional en tiempo real, ya que el proceso de la captura de información se lleva en papel o en computadoras aisladas, por lo que el proceso actual no es óptimo. La Atención Primaria de Salud no cuenta con aplicaciones que la conviertan en una parte informatizada dentro del SNS, por lo que se hace necesario crear un sistema que realice la gestión de la información a nivel Nacional, Provincial, Municipal y de Unidades de Salud y que permita la toma de decisiones a estos niveles de dirección.

[1] Soñora Cruz, William (2005). Registro de Actividades Diarias del Equipo Básico de Salud Para el Sistema Informatizado de Atención Primaria (Tesis de Pregrado, Instituto Superior Politécnico "José Antonio Echeverría", Universidad De Holguín "Oscar Lucero Moya" Universidad de las Ciencias Informáticas).

Introducción

Las Áreas de Salud en Cuba están constituidas por el área geográfica a la que presta sus servicios una unidad de salud que contemple el Programa de Trabajo del Médico y la Enfermera de la Familia. En estas unidades (pueden ser policlínicos u hospitales rurales) se atiende a la mayoría de la población en el nivel de atención primaria.
Actualmente no se tiene un control efectivo de la información que se gestiona en las Áreas de Salud relacionada con las Localidades que atienden, y en dependencia de su ubicación geográfica con los Hospitales Base que sirven de apoyo a la hora de remitir un paciente a un nivel de atención de salud secundario, con las Poblaciones que se atienden en los Equipos Básicos de Salud (EBS) y que forman parte de los diferentes Grupos Básicos de Trabajo (GBT) del Área de Salud y con la composición o plantilla definida para los GBT y sus EBS, según la estructura organizativa propuesta por la Atención Primara de Salud. No se cuenta con un buen control de los Departamentos que existen, así como los Servicios Médicos que se brindan a la población en cada departamento del Área de Salud y con los locales de viviendas o de consultas asignados al personal de salud del Área.

Esta información se gestiona en las Áreas de Salud de forma local y a medida que sube por los diferentes niveles de dirección de salud va disminuyendo en precisión cualitativa, solamente algunos de estos datos (cuantitativos) se recogen en modelos estadísticos que forman parte del Sistema de Información Estadístico Complementaria (SIE-C) del (MINSAP).

El Sistema de Información para la Salud (SiSalud) es la unión de varios componentes que tienen como objetivo la informatización de la sociedad cubana en el campo de la salud. El Registro Informatizado de Salud (RIS) constituye la materialización de la estrategia metodológica de Informatización del Sistema Nacional de Salud. La misma cuenta con el modulo Registro de Unidades de Salud (RUS) que gestiona la información de las Unidades de Salud del país y que define cuándo una Unidad de Salud es un Área de Salud, pero el resto de la información de las Áreas de Salud no forma parte del negocio a resolver por dicho módulo. Por esta razón no se dispone de un registro nacional de las Áreas de Salud que permita gestionar y controlar de forma centralizada y en cada nivel de dirección la información de las Áreas de Salud u obtener las estadísticas que permitan establecer políticas de administración en salud.

Existen sistemas informáticos desarrollados a nivel internacional relacionados con las Áreas de Salud, pero sus funcionalidades están dirigidas a la gestión hospitalaria y al trabajo que realizan los médicos para

prevenir y dar solución a los problemas de salud que afectan a sus poblaciones. En Cuba se han hecho algunos intentos de mostrar información de las Áreas de Salud, pero no pasan de ser Sitios Web estáticos o trabajos investigativos que muestran mediante imágenes la estructura organizativa que puede tener un Área de Salud, resaltando siempre los Consultorios del Médico de la Familia y la ubicación de las poblaciones que atiende un Equipo Básico de Salud, así como algunos datos estadísticos que se relacionan con las investigaciones y estudios que realizan los médicos con sus poblaciones.

Luego de un profundo análisis de lo expuesto anteriormente y de la necesidad de encontrar una solución para dar respuestas a estas dificultades, que de alguna forma debilitan la prevención de enfermedades y la prestación de los servicios médicos a la población, todos los esfuerzos estarán encaminados a resolver el siguiente **problema**: ¿Cómo gestionar de manera automatizada la información de las Áreas de Salud en Cuba?

Como **objeto de estudio** se ha identificado el proceso de gestión de la información de la Atención Primaria en el Sistema Nacional de Salud. Además tenemos en cuenta que nuestra investigación se centra fundamentalmente en el Proceso de gestión de la información de las Áreas de Salud de la Atención Primaria en el Sistema Nacional de Salud siendo este nuestro **campo de acción**,

Se propone como **Objetivo General** diseñar y desarrollar una aplicación Web que gestione la información correspondiente a las Áreas de Salud, haciendo uso de la tecnología XML Web Services y la Arquitectura Orientada a Servicios y Basada en Componentes (SOA-CBA) para aplicaciones distribuidas.

Se plantea como **idea a defender** la siguiente:

- ✓ Con el desarrollo de la aplicación se permitirá un mejor control de la información y de forma eficiente la toma de decisiones en las áreas de salud y en los diferentes niveles de dirección de salud.

Para lograr el objetivo planteado anteriormente se trazaron las siguientes **tareas de investigación**:

- ✓ Realizar un análisis de las tendencias, herramientas y tecnologías actuales para llevar a cabo el proceso de implementación de dicha solución automatizada como PHP, XML, XSLT, Web Servicies, MySQL, XHTML, Java Script.
- ✓ Analizar la integración con otros componentes o partes del SNS y fuera de este.
- ✓ Modelar, siguiendo el proceso Unificado de Rational (RUP), el flujo de trabajo “Diseño” a partir del resultado del flujo de trabajo del “Análisis”; utilizando la arquitectura definida por el MINSAP y estándares de diseño.

- ✓ Modelar el flujo de trabajo “Implementación”, utilizando RUP.
- ✓ Implementar el módulo RAS, utilizando estándares de codificación y tratamiento de errores o excepciones.

El Registro de Áreas de Salud proporcionará los siguientes beneficios a la sociedad y al Sistema Nacional de Salud:

- ✓ Permitirá gestionar y controlar de forma centralizada en los diferentes niveles de dirección, la información de las Áreas de Salud.
- ✓ El RAS es la base para la gestión de la Historia de Salud Familiar en el Registro de Población y brinda información a otros componentes del Sistema de Información para la Salud.
- ✓ Mejor funcionamiento de las Áreas de Salud, lo cual repercute en un mejor servicio a la población.

Este documento está compuesto por tres capítulos que incluyen todo el trabajo investigativo, así como también los temas referentes al diseño e implementación de la solución que se propone.

En el Capítulo 1: Fundamentación Teórica, se explican los principales conceptos como el Sistema Nacional de Salud (SNS) en nuestro país, sus diferentes niveles de atención, cómo se desarrolla su proceso de informatización, la necesidad de crear el Registro de Áreas de Salud. Se mencionan los sistemas existentes y se realiza un estudio de las tecnologías, lenguajes, metodologías y herramientas utilizadas para realizar el análisis y diseño, las cuales serán utilizadas en la implementación del sistema.

En el Capítulo 2: Diseño del Sistema, se refleja el flujo de trabajo del diseño del sistema que da seguimiento al flujo de trabajo del análisis. Se presenta el diagrama de clases de algunos casos de uso del sistema, haciendo uso de los esteriotipos WEB. También se describen las clases del diseño utilizadas en estos diagramas de clases.

En el Capítulo 3: Implementación, se justifica la integración del Registro de las Áreas de Salud con otros componentes del Sistema de Información para la Salud (SISalud). Se describen algunos servicios Web, métodos o agentes más complejos de la aplicación. Se presentan los diagramas de componentes, el diagrama de despliegue de la aplicación y los estándares de diseño, codificación y tratamiento de excepciones, tanto en la capa de negocio como en la capa de presentación.

Capítulo 1: Fundamentación Teórica

Introducción

Este capítulo tiene como objetivo fundamental abordar distintos aspectos que se utilizan como soporte teórico para el desarrollo de la aplicación. Se hace una exposición de la estructura organizativa del Sistema Nacional de Salud, sus diferentes niveles de atención, así como del proceso de informatización para la salud. Se incluye la situación problémica, el análisis de los sistemas existentes relacionados con el módulo y el estudio de los conceptos y las tecnologías a utilizar. Se describen además los lenguajes de programación, la metodología y las herramientas de trabajo.

1.1 El Sistema Nacional de Salud

A partir del triunfo de la Revolución el Gobierno Revolucionario se encargó de enfrentar y erradicar los problemas sanitarios existentes en el país y con esto elevar sustancialmente los niveles de vida y salud de la población. Para lograrlo dictó las primeras medidas en el campo de la salud, las cuales fueron:

- ✓ Fundación del Sistema Nacional de Salud (SNS).
- ✓ Implementación del Servicio Médico Rural.
- ✓ Creación de las Áreas de Salud y los Policlínicos.

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 conocemos 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».[2]

[2] Marín Díaz, Miguel E. Fundamentos del Sistema de Salud Pública en Cuba para estudiantes de Informática. Ciudad Habana, 2006.

Capítulo 1: Fundamentación Teórica

El Sistema Nacional de Salud de Cuba se clasifica atendiendo a su distribución administrativa y a los niveles de atención a la población.
Según la distribución administrativa (ver Anexo1):

1. **Nivel Nacional**: está representado por el Ministerio de Salud Pública como órgano rector, tiene funciones metodológicas, normativas, de coordinación y control.
2. **Nivel Provincial**: está 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).
3. **Nivel Municipal**: está representado por las Direcciones Municipales de Salud, dependientes administrativa y financieramente de la Asamblea Municipal del Poder Popular.
   En este nivel constituyen un eslabón importante los Consejos Populares los cuales trabajan en estrecha coordinación con el SNS a su nivel.

Según los niveles de atención a la población el SNS se clasifica:

1. **Atención Primaria de Salud (APS)**

   Constituye el primer encuentro del paciente sano o enfermo con el sistema de salud, el cual puede realizarse en cualquier institución médica, aunque generalmente se realiza en el Policlínico o en el Consultorio Médico.

   Este nivel de atención se identifica como aquel en el que se brindan diagnósticos y tratamientos terapéuticos que no requieren técnicas complejas, que aplicadas con calidad, pueden resolver la mayor parte de las dolencias que afectan a la población. Los médicos pueden, en sus comienzos, sospechar enfermedades graves, que deben ser derivadas a niveles de atención superiores; realizar seguimiento a las personas con padecimientos crónicos y otorgar bienestar a pacientes con patologías incurables. En general tiene carácter ambulatorio, consultoríal, y comprende tanto a sanos como a enfermos.

   La Atención Primaria de Salud es un nivel cualitativamente superior de atención, cuya esencia radica en la participación activa de la comunidad; donde las poblaciones, de objetos pasivos en espera de que se le ofrezcan soluciones, pasan a ser sujetos protagónicos activos ante sus propios

problemas de salud. Decir participación comunitaria, es decir liderazgo, comunicación, cambio de hábitos, estilos de vida, auto responsabilidad y acción creadora.

2. Atención Secundaria de Salud

La Atención Médica Secundaria es la que se proporciona en un segundo escalón, al cual el usuario tiene "por lo general" acceso a través de una remisión del médico en el nivel de Atención Primaria.

Puede tener carácter ambulatorio (policlínicos, servicios externos hospitalarios) o de hospitalización. En el mismo se ofrecen servicios técnico-terapéuticos de elevada complejidad, que dan respuesta a los problemas moderados y graves de salud.

3. Atención Terciaria de Salud

La Atención Médica Terciaria es aquella que por su condición muy especializada, sólo se brinda en determinados centros, ejemplo: Servicios de Neurocirugía, Nefrología, Cirugía Cardiovascular, Transplante Renal, Quemados, etc. Incluye además los Centros e Institutos de Investigaciones.

1.2 Informatización del Sistema Nacional de Salud

Tiene como objetivo acercar eficientemente y con calidad la prestación de los servicios de salud a la población, por lo que se pretende implementar un Programa General de Informatización del SNS, que apoye las estrategias y políticas trazadas por la dirección del país y del MINSAP.

Se quiere que las instituciones del país alcancen un elevado nivel de informatización de las actividades que brindan, partiendo del Sistema de Atención Primaria y tomando como eje al policlínico, de manera que se logre un incremento de la calidad, efectividad y eficiencia de los servicios que se presten a la población. Como solución integral significa la articulación de un nuevo paradigma en la prestación de servicios de salud.

Entre las líneas generales del Desarrollo Informático en la Salud se encuentran: la Atención Primaria, Secundaria y Terciaria, el Sistema Integrado de Urgencia Médica, Vigilancia de Salud, Telemedicina, Medicamentos y Fármacos, Epidemiología, Biblioteca y Universidad Virtual, Docencia Médica, entre otros.

El sistema de salud cubano, posee en el nivel de Atención Primaria una plataforma ideal para articular los avances de las nuevas tecnologías de la información en función de hacer más eficiente todo el aparato estratégico y administrativo que rodea al propio sistema.[3]

1.2.1 Informatización de la Atención Primaria de Salud

La informatización en este nivel debe permitir recoger todos los datos necesarios para la gestión médica, la interacción con los consultorios del Médico de la Familia, la obtención de estadísticas y el flujo de información hacia los diferentes niveles de toma de decisiones. Todo esto traerá consigo una mayor y mejor atención a la población.

Este proceso de informatización debe garantizar la capacidad de comunicación e integración de toda la información, independientemente de donde se haya generado; servir para el aprendizaje basado en experiencias compartidas entre los profesionales en el país, así como lograr la integración con los procesos de los otros niveles de atención y otras aplicaciones que gestionen esta información.

1.2.2 El Sistema de Información para la Salud (SISalud)

El Sistema de Información para la Salud (SISalud) es la integración de un conjunto de aplicaciones cuyo propósito fundamental es la informatización del Sistema Nacional de Salud (SNS). Cada uno de los componentes que integran este sistema desarrolla sus procesos y los pone a disposición del resto, permitiendo la interoperabilidad, el intercambio de información entre los mismos y el acceso a la información desde los diferentes niveles de dirección, ya sea nacional, provincial, municipal o de unidad de salud.

SISalud está compuesto por el Registro Informatizado de Salud (RIS), el Sistema Informatizado de Atención Primaria (SIAP) y el Sistema de Gestión Hospitalaria (SIGH). El RIS está formado por los componentes que integran el Registro no Médico de Información de Salud y el Registro Médico Informatizado de Salud. El SIAP contempla los módulos que son propios de APS, tales como el Registro de Actividades Diarias del Equipo Básico de Salud (EBS) y el Registro de Población que permitirán la

[3] Varios Autores. Propuesta de Esquema Sistema Integral de Salud. Ciudad Habana, 2006.

transformación de los servicios que se brindan en este nivel de atención. El SIGH agrupa los módulos que pertenecen al nivel de Atención Secundaria tales como el Registro de Autopsias.

1.2.3 Registro Informatizado de Salud (RIS)

Es la solución informática integral para la Salud Pública, acorde con los objetivos de la informatización de la sociedad cubana. Constituido por un conjunto de aplicaciones independientes (módulos del sistema) que se interconectan según las necesidades del flujo de información. Es además la herramienta que permite a los usuarios autorizados combinar la información de los diferentes módulos que lo componen, para obtener una información integral en tiempo real para la toma de decisiones en los diferentes niveles de dirección, la docencia, investigación y la gestión en salud.[4]

El RIS, es un sistema formado por componentes, desarrollados con un nivel de cohesión y acoplamiento que le permiten ser capaces de interactuar entre ellos y de esta forma reutilizar la información gestionada por cada componente. El Registro de Áreas de Salud es uno de los módulos del RIS.

Situación problémica:

Con el objetivo de mejorar la salud y la calidad de vida de la población, los programas trazados por la Revolución contribuyeron al incremento del número de unidades de salud, la creación de policlínicos de nuevo tipo y los Consultorios del Médico y la Enfermera de la Familia (CMF). Para garantizar llevar hasta los lugares más intricados las acciones de atención médica y prevención de enfermedades se hace necesario mejorar los servicios que se le brindan a la población en las Áreas de Salud.

Las Áreas de Salud en Cuba son áreas geográficas delimitadas en kilómetros cuadrados, donde sus habitantes son beneficiados con los servicios médicos que se brindan en una unidad de salud que contemple el Programa del Médico y la Enfermera de la Familia. Estas unidades de salud pueden ser hospitales rurales o policlínicos y responden a la mayoría de los problemas de salud de la población en el nivel de la Atención Primaria. En las Áreas de Salud operan los Grupos Básicos de Trabajo y los Equipos

[4] Delgado Ramos, Dr. Ariel, Cabrera Hernández, Ing. Mirna, Juncal, Dra. Virginia (2005). Registro Informatizado de Salud (RIS). Consultado en (Enero, 11, 2007) en http://www.sld.cu/galerias/pdf/sitios/dne/ris.pdf

Básicos de Salud, que constituyen estructuras organizativas fundamentales para la prevención y el control de los problemas de salud de la población en el Sistema Nacional de Salud.

El control de los datos que se gestionan en las Áreas de Salud no es efectivo. Existen dificultades con las acciones a desarrollar, relacionadas con la transmisión de la información generada por los Grupos Básicos de Trabajo y los Equipos Básicos de Salud. La información que generan estas estructuras organizativas, se hace mediante el uso de métodos tradicionales como llamadas telefónicas, correos convencionales y en la mayoría de los casos mediante papeles, que provocan demoras en el procesamiento de los datos.

En los hospitales rurales y en los policlínicos no se gestiona de forma eficiente la información relacionada con las localidades que componen cada una de las Áreas de Salud, los servicios médicos que se brindan por cada departamento, los hospitales base que sirven de apoyo para dar solución a los problemas que sólo pueden tener tratamiento en el nivel secundario de salud, las plantillas según la estructura organizativa propuesta por la Atención Primara de Salud que presentan los Grupos Básicos de Trabajo y los Equipos Básicos de Salud, así como las poblaciones que atienden los Equipos Básicos de Salud y los locales de vivienda o consulta que ocupan los profesionales de la salud que participan en el Programa del Médico y la Enfermera de la Familia.

Con la creación de las Áreas de Salud se han podido observar mejoras en los servicios médico-sanitarios, lo que ha repercutido positivamente en la calidad de vida de los habitantes; pero a la vez se han detectado problemas que dificultan sin duda alguna el control de la información que se genera en el proceso de atención a pacientes y familias. Mucha de la información que se recibe a nivel nacional sobre las acciones que se desarrollan en las Áreas de Salud, tiende a duplicarse, se emplea gran cantidad de tiempo en la elaboración de documentos que muestran datos estadísticos, se pierde el criterio favorable de la información ofrecida, por la no actualización de la misma con respecto a su fecha de emisión. En muchas ocasiones es necesario imprimir nuevamente los datos que llegan desde las provincias para poder captarlos.

Parte de la información que se procesa en estas unidades de salud se realiza de forma local y manual, esto provoca pérdida en la precisión cualitativa de los datos, mediante el flujo a través de los diferentes niveles de dirección. Solamente algunos de los datos, de carácter cuantitativo, se recogen en algunos modelos estadísticos que pertenecen al Sistema de Información Estadístico Complementaria (SIE-C) del

Ministerio de Salud Pública. La descentralización de la información provoca dificultades con el transporte y la periodicidad de los datos. Por estas razones se requiere de un sistema que permita la captación y el procesamiento de los datos a todos los niveles de dirección.

El Sistema de Información para la Salud (SiSalud) constituye una de las estrategias a desarrollar para la informatización del Sistema Nacional de Salud. Entre las partes que componen este sistema se encuentra el Registro Informatizado de Salud. Uno de sus componentes es el Registro de Unidades de Salud, el cual gestiona la información de las Unidades de Salud y define cuándo una unidad es un Área de Salud, pero no gestiona el resto de la información de las unidades como Áreas de Salud. Por tanto no se dispone de un registro nacional de las Áreas de Salud que permita controlar y gestionar de forma centralizada la información de las mismas.

1.2.4 Sistemas Automatizados Existentes Relacionados con las Áreas de Salud

Para la argumentación del tema se realizó una investigación nacional e internacional sobre la existencia de sistemas informáticos, portales Web, entre otros, que tuvieran algún tipo de relación con la funcionalidades del Registro de Áreas de Salud.

Actualmente no existe ningún sistema conocido que cumpla con las funcionalidades o requisitos que se pretenden implementar en este registro, sólo se han encontrado, tanto a nivel nacional como internacional, algunos portales Web que brindan alguna información de las Áreas de Salud; los mismos son sitios estáticos que no gestionan toda la información de las Áreas de Salud, ejemplo de ello son los siguientes:

Nacionalmente existen los sitios del Policlínico Docente Dr. Mario Muñoz Monroy de la Dirección Municipal de Salud de la Habana del Este (Guanabo), Ciudad de la Habana y del Policlínico Universitario Vedado. Dentro de la información que brindan se puede encontrar: los Servicios Asistenciales del Policlínico, los horarios de consultas de las diferentes especialidades y los nombres de los profesionales de salud que las efectuarán.

A nivel internacional Murcia Salud, un portal sanitario de la región de Murcia, España, brinda un listado de los Centros de Salud de Atención Primaria de Murcia, que permite acceder a los datos generales de cada centro de salud como: Zona de Salud, Responsable de Enfermería, Responsable del Personal de Apoyo, Domicilio, Teléfono de cita previa, Teléfono de urgencias, Fax, Superficie en m2, entre otros. También se muestran los servicios que se ofrecen en el propio centro y los servicios externos, además de los nombres

de profesionales del centro y algunos datos de estos. Otro ejemplo es el Sistema Aragonés de Salud, un portal que permite realizar una búsqueda de los centros de salud de la región por diferentes parámetros, como por ejemplo: Nombre del centro, Tipo de centro, Profesional sanitario y Servicio que brinda.

Debido a la importancia que tiene la información de las Áreas de Salud en el proceso de informatización de la Atención Primaria, y la no existencia de sistemas que gestionen esta información, se decide desarrollar el Registro de las Áreas de Salud, el cual sirve de apoyo a otros módulos que forman parte del Sistema de Información para la Salud. Las poblaciones, los Grupos Básicos de Trabajo y los Equipos Básicos de Salud son definidos en este componente y utilizados por diferentes módulos como el de Población, para la gestión de la Historia de Salud Familiar.

1.3 Tecnologías, Herramientas y Metodología Utilizadas

1.3.1 Internet

Internet es una red mundial de millones de computadoras conectadas por un conjunto de protocolos, el más usado es TCP/IP. Aparece por primera vez cuando ARPANET establece su primera conexión entre tres universidades en California y una en Utah. También se usa el término Internet como sustantivo común y por tanto en minúsculas, para designar a cualquier red de redes que use las mismas tecnologías que Internet, independientemente de su extensión o de que sea pública o privada, cada computadora conectada a la red de manera independiente. Para que todos estas computadoras puedan coexistir y comunicarse efectivamente entre sí, debe existir un camino físico que las una (líneas telefónicas, conmutadas, redes digitales, enlaces satelitales, microondas, fibra óptica, cable coaxial, etc.), además deben ponerse de acuerdo con la comunicación, es decir, usar el mismo protocolo de comunicación.

1.3.2 Aplicaciones Web

Una aplicación Web es una aplicación informática que permite a los usuarios acceder a un servidor a través de la red. La popularidad de las aplicaciones Web se debe principalmente a los navegadores como clientes ligeros y que para actualizar y mantener las aplicaciones no es necesario distribuir e instalar software en miles de potenciales clientes. En términos más simples, una Aplicación Web es un Sistema Web que permite a los usuarios ejecutar lógica de negocio a través de un Navegador o lo que es lo mismo, modificar el estado del negocio. Las aplicaciones Web hacen uso de las tecnologías que existen

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

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

1.3.3 El Servidor Web

El servidor Web es un sistema que realiza su función dentro del servidor que escucha las peticiones HTTP que recibe y que satisface, dependiendo del tipo de petición. El servidor Web construye una página Web que ejecuta un programa en el servidor, siempre elabora una respuesta en formato HTML para mostrar al cliente o al navegador que hace la petición. Constituye uno de los componentes fundamentales de las aplicaciones que corren del lado del servidor.

1.3.4 El servidor Web Apache

El servidor Web Apache es un software de código abierto que funciona sobre cualquier plataforma y se distribuye como software libre. Es uno de los servidores Web más potentes y de los más utilizados a nivel mundial. Se encuentra muy por encima de sus más fuertes competidores gratuitos y comerciales. Ofrece una fuerte estabilidad y sencillez, soporta varios protocolos de comunicación y su distribución se encuentra en todas las versiones de Linux. Apache está capacitado tanto para prestar servicios de páginas estáticas como de páginas dinámicas, su configuración se hace mediante la administración de un único archivo, permite el trabajo con las bases de datos, con los ficheros y posee Virtual Host.

1.3.5 El Navegador Web o Browser

Los navegadores constituyen la interfaz de usuario más usada a nivel mundial. Permiten presentar el contenido de una página Web; validar los errores de los datos de entrada a través de scripts, que se

ejecutan del lado del cliente y que pueden acceder de forma dinámica al modelo de objetos del documento de las páginas Web. Los lenguajes de programación que corren del lado del cliente son VBScript, JScript, y Java Script. Este último como uno de los más usados internacionalmente para solucionar procesos, o generalmente, validaciones que se ejecutan en el cliente, las cuales se hacen necesarias antes de enviar los datos al servidor.

Los navegadores o browser permiten trabajar con el documento HTML de forma dinámica (Dinamic HTML, DHTML). Para hacerlo involucran el código HTML de las páginas Web, las hojas de estilo en cascada (Cascade Style Sheets, CSS), el modelo de objetos (Document Object Model) y programas scripts que permiten formatear y posicionar los elementos HTML de las páginas Web, permitiendo un mejor control sobre la visualización de las páginas.

Los navegadores también permiten mostrar el contenido de un documento XML (Extensible Markup Language, XML) en formato HTML, esto es posible gracias a las hojas de estilo extensibles (Extensible Style Sheets, XSL) que permiten visualizar en formato HTML cualquier información o elemento de un documento XML. Los navegadores posibilitan la ejecución de aplicaciones como ActiveX y los Applets de Java dentro de los documentos mostrados. Los Applets de Java son programas pequeños que se descargan del servidor Web y que se ejecutan en la Máquina Virtual de Java del navegador.

1.3.6 HyperText Transfer Protocol, HTTP

HTTP es un protocolo para el desarrollo de sistemas de información multimedia distribuidos, no orientado a estado. Puede ser usado con varios fines, no sólo para transferir ficheros HTML. Se caracteriza por poseer un esquema de direccionamiento comprensible que utiliza Universal Resource Identifier (URI) para localizar sitios (URL) o nombres (URN) sobre los que hay que aplicar un método.

Por defecto, HTTP utiliza el puerto 80, aunque también puede trabajar con otros. Este protocolo no presenta conexión ni estado. Una vez que el servidor da respuesta a la petición del cliente se rompe la conexión entre ambos, no se guarda un historial de la conexión. HTTP acepta cualquier tipo de datos nuevos, utiliza Multipart Internet Mail Extensión para determinar el tipo de datos a transportar. El servidor

HTTP incluye una cabecera que indica al cliente los tipos de datos que presenta el documento, lo que hace posible que los navegadores muestren videos e imágenes entre otros.

1.3.7 Servicio Web

El servicio Web se utiliza para intercambiar datos entre aplicaciones bajo una colección de protocolos y estándares. Hacen posible que distintas aplicaciones, desarrolladas en plataformas diferentes y en cualquier lenguaje de programación, puedan intercambiar datos por la red. La interoperabilidad de estas aplicaciones se logra mediante estándares abiertos que muchas organizaciones se dedican a desarrollar.
Entre las ventajas que tienen los servicios Web podemos mencionar:

- ✓ Permiten la interoperabilidad entre las aplicaciones, sin importar el lenguaje y la plataforma en la que han sido desarrolladas.
- ✓ Formatean a los protocolos y estándares basados en textos, para facilitar el acceso a su contenido y para entender su funcionamiento.
- ✓ Se aprovechan de los sistemas de seguridad firewall apoyándose en el protocolo de comunicación http, sin necesidad de cambiar las reglas de filtrado.
- ✓ Facilitan que diferentes compañías ubicadas en diferentes lugares geográficos puedan proveer servicios integrados.

Como desventajas que presentan los servicios Web se pueden mencionar:

- ✓ Para algunos lenguajes de programación existe muy poca información sobre esta tecnología.
- ✓ Omiten medidas de seguridad basadas en firewall, cuyas reglas de seguridad tratan de bloquear o auditar la comunicación entre ambos lados de la barrera, esto sucede porque usan el protocolo de comunicación HTTP.
- ✓ El rendimiento es bajo comparado con otros modelos de computación ya que no presenta como objetivo la concisión ni la eficiencia de procesamiento.
- ✓ No es tan eficiente como otros estándares abiertos de computación distribuida para realizar transacciones.

Existen razones suficientes para usar servicios Web, la principal es que se basan en la combinación de los protocolos de comunicación HTTP y TCP por el puerto 80. Muchas organizaciones bloquean casi todos los puertos TCP, menos el 80, para proteger sus redes mediante firewalls. El puerto 80 es precisamente el

que usan los navegadores, los servicios Web usan este puerto porque no resulta bloqueado. Antes del surgimiento de SOAP (Simple Object Access Protocol) no existían buenas interfaces de comunicación para intercambiar funcionalidades entre los ordenadores. Los servicios Web aportan gran independencia entre aplicaciones.

1.3.8 Arquitectura en capas

La arquitectura en capas es un estilo de programación muy utilizado en la actualidad. Su principal objetivo es separar la lógica de negocio, de la capa de presentación, al usuario final sólo le interesa la visualización de la información en el sistema, no la forma en que se gestiona. Una de las ventajas principales de la arquitectura es que el desarrollo e implementación se puede realizar por capas y un cambio en una de ellas, debe implicar mínimos cambios en otras capas.

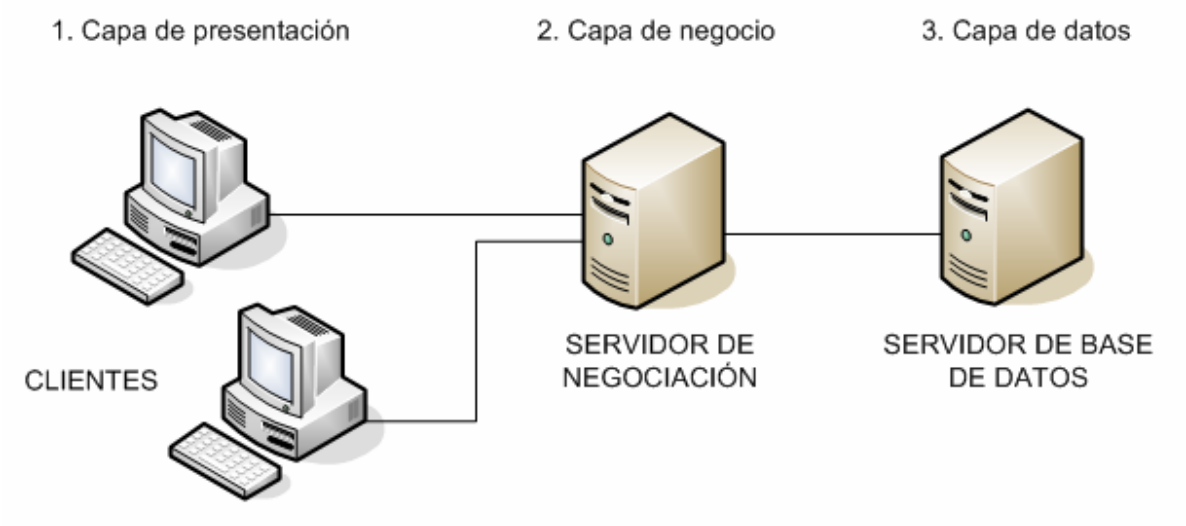


Figura 1.1: Arquitectura en capas

Esta arquitectura permite distribuir la implementación por capas, de forma tal que se puede dividir el grupo de desarrollo en equipos de trabajo por cada una de las capas que tenga el sistema. De esta forma cada equipo de desarrollo se puede abstraer del trabajo de los otros equipos, solamente le interesaría conocer elementos puntuales que existen entre cada una de las capas. El diseño que más se utiliza es el de la arquitectura en tres capas (*presentación, negocio, acceso a datos*), donde a cada capa se le asigna una

misión específica, que permite que el diseño de la arquitectura sea escalable y que pueda ampliarse con facilidad si es necesario.

Capa de presentación: esta capa es la responsable de presentar la información del sistema al usuario, es la única capa del sistema que el usuario puede ver. Las funcionalidades de la capa de presentación consisten en comunicarle al usuario la información, capturar y validar los datos, comprobar que no hay errores de formato y enviar los datos a la capa de negocio, que es la única con la que se puede comunicar. Reúne todos los aspectos necesarios para interactuar y construir las interfaces de usuarios, generalmente involucran el trabajo con ventanas, reportes, menús, gráficos y elementos multimedia.

Capa de negocio: esta capa también se conoce como capa de dominio o capa de lógica de negocio. Establece comunicación con la capa de presentación y con la capa de datos, es la encargada de recibir y responder cada petición de los usuarios. Los programas que la conforman reciben las solicitudes de los clientes, se comunican con las capa de datos o repositorio para almacenar, actualizar o recuperar información y emiten una respuesta. Constituye la parte del sistema donde se establecen todas las reglas de negocio que deben cumplirse. Agrupa todo lo necesario para automatizar y apoyar los procesos del sistema, generalmente engloba todas las tareas que forman parte de los procesos, las reglas y las restricciones que deben cumplirse.

Capa de datos: esta capa también se conoce como capa de repositorio y es la que recibe las solicitudes de la capa de negocio. Está conformada por uno o más gestores de bases de datos, que realizan el procesamiento de los datos, controlan el almacenamiento, modificación, recuperación y eliminación de los datos.

1.3.9 Importancia de la Arquitectura

Todos los elementos agrupados en una misma capa pueden establecer comunicación entre ellos. La comunicación con elementos de otras capas puede variar considerablemente, pues no todos los elementos de una misma capa pueden interactuar con elementos de otras capas.

Todas las capas de una arquitectura pueden estar ubicadas en un mismo servidor, esta no es la mejor solución; la más práctica, es que la capa de presentación esté ubicada en un servidor y la capa de negocio y la capa de datos estén ubicadas en otro. Si las necesidades lo aconsejan y el tamaño de las

bases de datos aumenta, estas se pueden separar en varios ordenadores, que recibirán las peticiones de la capa de negocio. Si la complejidad fuera de la capa de negocio, esta pudiera ubicarse en varios servidores que realizarían sus consultas a un mismo servidor de bases de datos. En sistemas de gran complejidad, la capa de datos y la base de datos corren sobre una serie de servidores diferentes.

Los términos niveles y capas no tienen el mismo significado. Una capa no es más que la forma en que una solución es definida y segmentada desde el punto de vista lógico, sin embargo el nivel se encarga de distribuir las capas físicamente como entidades dentro de la arquitectura. Un ejemplo de arquitectura de tres capas y un solo nivel, es el caso, en el que dentro de un mismo servidor, residen las capas de presentación, la capa de negocio y la capa de datos. Una solución de tres capas que corren sobre dos servidores, se dice que es una arquitectura de tres capas y de dos niveles. La arquitectura de tres capas y tres niveles es aquella donde cada una de las capas corre sobre un servidor diferente.

El patrón de arquitectura en capas tiene sus ventajas y desventajas como cualquier otro patrón. Entre sus ventajas, permite la reutilización de las capas y del código, la solución de un proceso puede aplicarse a otros similares. Permite estandarizar el desarrollo en cada una de las capas y hacer la gestión de cambios por capas.

Aplicaciones de tres capas

Una aplicación puede estar dividida en tres partes: la interfaz de usuario, encargada de interactuar con el usuario, recolectar y validar los datos de entrada; las reglas del negocio que procesan la información y la de acceso a datos para recuperar o modificar información del sistema.

Las aplicaciones de una capa son aquellas en las que la presentación, la lógica del negocio y el acceso a los datos están segmentadas en una sola entidad y físicamente ubicadas dentro de un mismo servidor, donde además, radica la base de datos, y se administran por una misma herramienta.

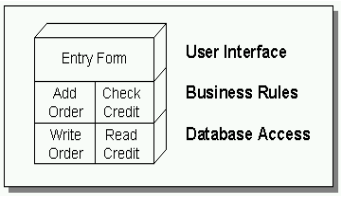


Figura 1.2. Arquitectura Típica de una aplicación de una sola capa.

Las aplicaciones de dos capas son las aplicaciones más conocidas como aplicaciones Cliente/Servidor porque dividen la aplicación entre el cliente y el servidor. Estas aplicaciones se dividen en dos entidades fundamentales, una contiene la capa de presentación y la otra la capa de negocio unida a la capa de datos o de repositorios. La ventaja de tener una aplicación distribuida por partes es que cualquier cambio en un componente no influye directamente en el resto de los componentes de la aplicación.

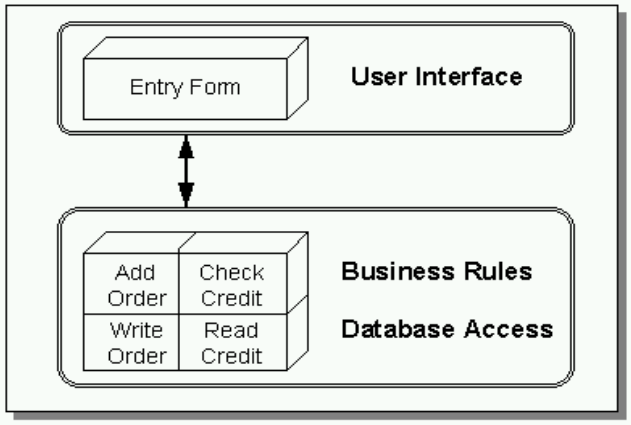


Figura 1.3. Arquitectura de dos capas con el acceso a la Base de Datos y las reglas de negocio encapsuladas.

Se puede tener en el mismo lado la capa de presentación y la capa de negocio

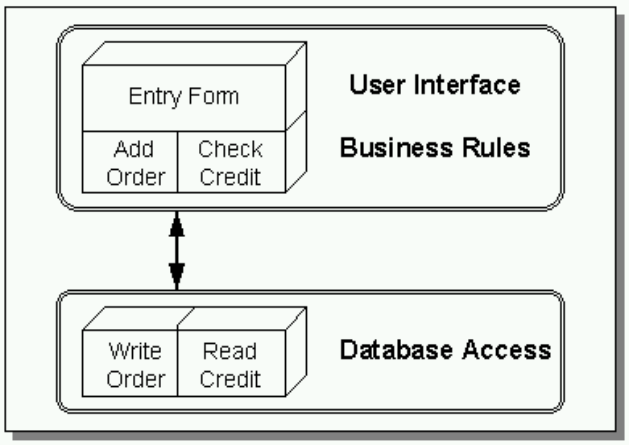


Figura 1.4. Arquitectura de dos capas con la interfaz y las reglas de negocio encapsuladas juntas.

Tener la capa de presentación ligada a la capa de negocio, no es una buena solución, ya que un cambio en la aplicación, provoca hacer el cambio para cada usuario. Agrupar las reglas del negocio con los datos, tiene como ventaja, que se puede cambiar la capa de negocio sin tener que cambiar la capa de presentación, el problema está en que generalmente los servidores de datos no son muy moldeables y es bastante complicado implementar reglas de negocio en los servidores. No resulta una práctica ligar los procesos de negocio con los datos, lo más frecuente es que el servidor de datos procese las consultas y todas las actividades de las bases de datos. Cada cliente debe iniciar y dejar abierta una conexión al servidor para procesar las respuestas.

En las arquitecturas de tres capas cada una de las capas se ve como una entidad. Esto facilita, que la implementación de cada componente sea más flexible. Esta arquitectura es la más compleja. Todas las solicitudes de los clientes se procesan en la capa de negocio.

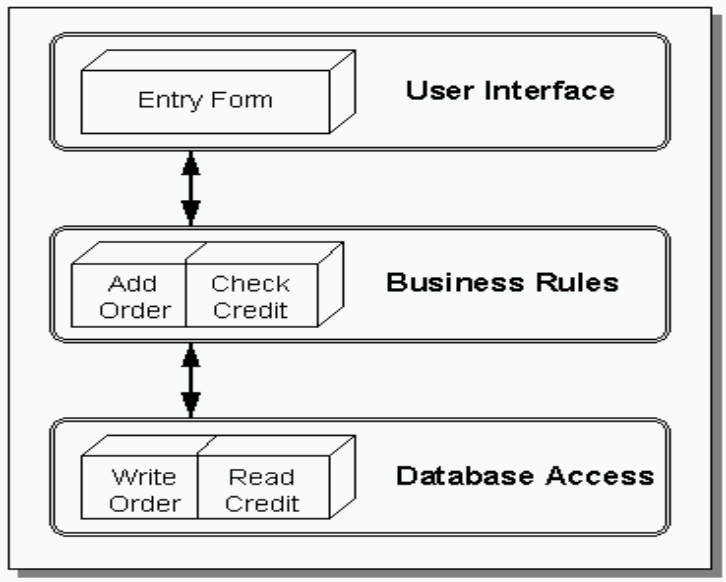


Figura 1.5. Arquitectura tres capas.

El cliente por su parte, para encuestar la capa de negocio no necesita tener instalados los driver ODBC, debe hablar el mismo lenguaje que el gestor de las reglas de negocio y este último debe hablar el mismo idioma que el servidor de bases de datos.

1.3.10 El modelo cliente – servidor

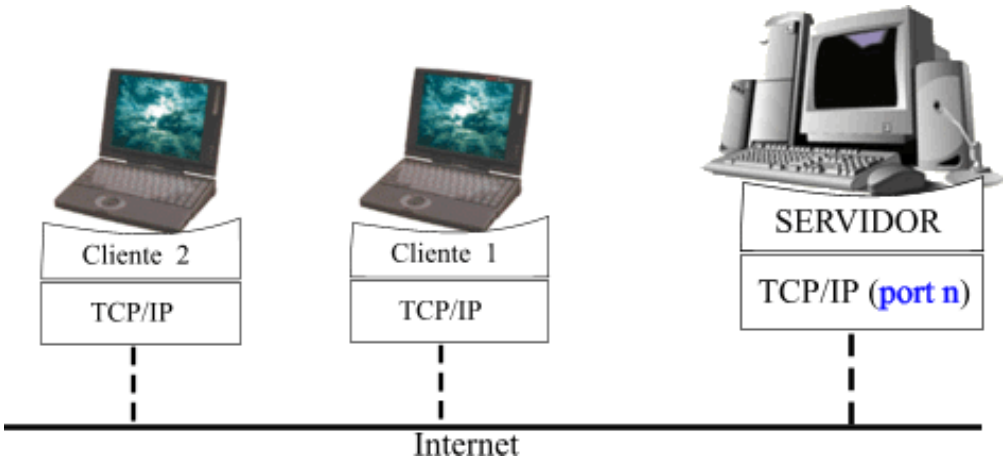


Figura 1.6: El modelo de aplicación cliente/servidor

El modelo Cliente/Servidor es utilizado por todas las aplicaciones de Internet, este modelo presenta una parte cliente y una parte servidor que se pueden ejecutar en el mismo o en otro sistema. Un cliente funciona en un ordenador y pide el servicio a un servidor, este es una aplicación que ofrece servicios a los usuarios de Internet. Los usuarios invocan la parte cliente de la aplicación, que construye una solicitud para ese servicio y se la envía al servidor de la aplicación. El cliente generalmente se comunica con el servidor remoto, mediante el protocolo de comunicación TCP/IP como transporte y pide la información, el servidor se encarga de emitir una respuesta.

Un servidor brinda servicios a millones de usuarios ahorrándole a cada uno de ellos el trabajo de tener instalada y almacenada información de forma local. En la mayoría de los casos el servidor puede tratar múltiples peticiones al mismo tiempo. Los clientes emplean un puerto arbitrario para comunicarse, algunos servidores esperan las peticiones por puertos bien conocidos por los clientes. Los que quieran conectarse a un servidor que no usa un puerto conocido desarrollan un mecanismo para conocer a qué puerto dirigirse.

El modelo Cliente/Servidor puede variar. En dependencia de las aplicaciones que el servidor pone a disposición de los clientes, se pueden mencionar, los servidores de impresión, mediante los cuales los usuarios comparten la impresora; los servidores Web le añaden aspectos nuevos al modelo Cliente/Servidor; los servidores de Lotus Notes, que permiten el trabajo simultaneo de distintos clientes con los mismos datos, documentos o modelos; los servidores de bases de datos donde existe una única base de datos y los servidores de archivos donde los clientes comparten el disco duro.

En la actualidad el hardware, los sistemas operativos y las redes brindan la posibilidad de desarrollar e implementar aplicaciones que en otros tiempos no se podían desarrollar. La selección de una arquitectura de dos o de tres capas no depende del hardware que se tenga. Desde los primeros tiempos de la programación, los desarrolladores han estado implementando aplicaciones Cliente-Servidor, excediendo varias veces las posibilidades del hardware, partiendo las aplicaciones en trozos que dialoguen entre sí. Cualquier aplicación desarrollada bajo el modelo de arquitectura de tres capas, no cumple su objetivo si no es escalable, y se puede desarrollar lo mismo en una PC que en una NetPc o en entornos tan diversos como Mac, Microsoft Windows NT o servidores Unix y brindar servicios a aplicaciones de datos en redes de Microsoft Windows o Mac.

Los problemas del desarrollo de las aplicaciones Cliente/Servidor se pueden dividir en tres partes, los problemas de implementación, los de depuración y los de la calidad del resultado. Esto siempre es bueno conocerlo ya que este tipo de aplicaciones en la mayoría de los casos corren o se desarrollan sobre diferentes servidores, en los que los sistemas operativos pueden variar.

1.3.11 Arquitectura Basada en Componentes (CBA).

La Arquitectura Basada en Componentes consiste en desarrollar una aplicación utilizando uno o varios componentes, que son diseñados significativamente para la aplicación y que hacen uso de otros componentes prefabricados que se ensamblan entre si para proporcionar servicios que el sistema necesita. El objetivo fundamental del la Arquitectura Basada en Componentes es construir aplicaciones complejas, ensamblando módulos o componentes que han sido anteriormente diseñados por un grupo de personas, con el fin de que puedan ser rehusados en múltiples aplicaciones. La interfaz constituye el elemento básico de interconectividad para esta tecnología, cada componente debe describir la interfaz que ofrece, así como las que necesita.

La modularidad, la reusabilidad y componibilidad son características muy relevantes de la Arquitectura Basada en Componentes, estas características coinciden con la tecnología de la programación orientada a objetos, de la que se puede considerar una evolución. Sin embargo esta arquitectura debe ser robusta ya que los componentes deben funcionar en entornos heterogéneos y diversos. La Arquitectura Basada en Componentes puede mejorar la calidad, disminuir los tiempos de desarrollo y gestionar la creciente complejidad de los sistemas.

1.3.12 Arquitectura Orientada a Servicios (Service-Oriented Architecture SOA)

La Arquitectura Orientada a Servicios define cómo interactúan dos entidades de cómputo de tal manera que permita a una entidad efectuar una unidad de trabajo a nombre de la otra. La unidad de trabajo se conoce como servicio, y la interacción entre servicios se define utilizando un lenguaje descriptivo. Cada interacción se contiene a si misma y están sueltas de tal manera que cada interacción sea independiente de cualquier otra interacción. Es un concepto de sistema que define la utilización de servicios para dar soporte a los requerimientos de software de los usuarios.

En un ambiente SOA, los nodos de la red hacen disponibles sus recursos a otros participantes en la red, como servicios independientes a los que tienen acceso de un modo estandarizado. La mayoría de las definiciones de SOA utilizan servicios Web, empleando los protocolos SOAP y WSDL (Web Services Description Language) en su implementación. Pero también es posible implementar una solución de este género utilizando cualquier tecnología basada en servicios.

Lo opuesto a este tipo de arquitecturas son aquellas orientadas a objetos. Para comunicarse entre sí en una SOA, los servicios se fundamentan en una definición formal e independiente de la plataforma subyacente y del lenguaje de programación. De esta forma se trata que los componentes de software desarrollados sean muy reusables, ya que la interfaz se define siguiendo un estándar.

Para que un proyecto SOA tenga éxito, los desarrolladores de software deben orientarse a la mentalidad de crear servicios comunes, que son generados por clientes o middleware para implementar los procesos

de negocio. A su vez, el desarrollo basado en esta arquitectura requiere un compromiso con este modelo en términos de planificación, herramientas e infraestructura.

1.3.13 Sistema operativo GNU/Linux

GNU/Linux es un Sistema Operativo basado en la implementación de libre distribución UNIX para computadoras personales, servidores, y estaciones de trabajo. Como sistema operativo, GNU/Linux es muy eficiente y tiene un excelente diseño. Es multitarea, multiusuario, multiplataforma y multiprocesador. En las plataformas Intel corre en modo protegido; protege la memoria para que un programa no pueda hacer caer al resto del sistema; carga sólo las partes de un programa que se usan. Comparte la memoria entre programas aumentando la velocidad y disminuyendo el uso de memoria; usa un sistema de memoria virtual por páginas; utiliza toda la memoria libre para cache. Permite usar bibliotecas enlazadas, tanto estática como dinámicamente; se distribuye con código fuente. Usa hasta 64 consolas virtuales, tiene un sistema de archivos avanzado pero puede usar los de los otros sistemas. Soporta redes tanto en TCP/IP como en otros protocolos.

1.3.14 Plataforma de Servicios PlaSer

PlaSer Plataforma de Servicios sobre la cual se pueden desarrollar aplicaciones Web basadas en aplicaciones XML-Web Servicies. Permite desarrollar la Arquitectura Basada en Componentes y Orientada a Servicios, utilizando SOAP como protocolo de comunicación para transportar los XML, todo el código ha sido implementado en PHP. Desde el punto de vista de estructura, permite trabajar con cualquier tipo de bases de datos que cumpla con el estándar SQL-92; pero desde el punto de vista de implementación sólo trabaja con las bases de datos soportadas por el componente DBX, ya que encapsula dicho componente y lo utiliza para el acceso a las bases de datos. Los programadores que usan esta plataforma no tienen que preocuparse por la seguridad. Implementa y maneja la seguridad, además de facilitar la programación y homogeneidad de los componentes.

Está integrada por una colección de clases y una “estructura de directorios” (layout) en la capa de presentación. También constituye una capa abstracta de transporte para la interoperabilidad entre los componentes PlaSer. En lo fundamental, está integrada por varias clases desarrolladas en PHP, una librería, que puede o no ser usada para que un componente se integre, pero que de no ser utilizada la

seguridad corre por parte del programador. En esta versión de PlaSer sólo soporta como llamada a procedimientos remotos (*Remote Procedure Call* RPC) el protocolo SOAP, en futuras versiones soportará otros protocolos de transportes e incluso el acceso local a código a nivel de File System.

PlaSer, en su configuración ideal, se distribuye en tres servidores, los cuales se encuentran conectados en cascada y sólo el primero esta conectado a Internet con una IP real. Este contiene la capa de presentación (layout). El segundo contiene a ProxPla y los demás componentes, incluido el componente de seguridad SAAA (Single Autorización Autenticación and Autenticidad) que permite la autenticación y la autorización para entrar al sistema, así como el registro de los métodos del negocio. El tercer servidor contiene las bases de datos desarrolladas en MySQL. Para aumentar la seguridad se pudiera colocar a ProxPla solo en un cuarto servidor separado de los demás componentes. PlaSer también puede ser desplegado totalmente en un solo servidor, aunque esto no es recomendable por motivos de seguridad.

1.4 Lenguajes de Programación Web.

La programación Web incluye tres conceptos fundamentales que se deben conocer antes de hacer una página con cualquier lenguaje de programación Web. Un concepto es el URL (Uniform Resource Locators), sistema con el cual se localizan recursos dentro de la red, aunque también puede ser el nombre que identifique a una computadora o un archivo que indique el camino de un recurso solicitado. Otro de los conceptos fundamentales es el protocolo de comunicación HTTP (Hypertext Transfer Protocol), elemento encargado de llevar la información de una página Web por toda la red. El ultimo de los conceptos es el lenguaje de programación Web HTML (Hypertext Markup Language), cuya funcionalidad es representar los datos que se encuentren almacenados en una página Web, preparar y codificar documentos de hipertextos, además de ser el lenguaje común para las páginas Web.

1.4.1 PHP

En la actualidad, uno de los lenguajes de programación del lado del servidor más conocido y más utilizado es el PHP. Proporciona principalmente características dinámicas a las páginas Web. Permite las conexiones con bases de datos MySQL, aunque no sólo con estas. Permite mezclar código HTML. Se interpreta y se ejecuta directamente del lado del servidor, que es donde se encuentra la página Web. Con

PHP se puede procesar información proveniente de formularios, generar páginas Web de contenidos dinámicos, enviar y recibir cookies.

Los scripts de PHP se usan en tres campos fundamentales, el más tradicional y el principal es el de trabajo es el de los scripts del lado del servidor. Para que los scripts del lado del servidor funcionen, se necesitan tres cosas fundamentales, el intérprete PHP, el servidor Web y el navegador. A través del navegador se puede ver el resultado del programa en PHP conectándose al servidor Web.

Otro campo es el de los scripts en la línea de comandos, donde se crean scripts PHP y se corren, sin tener instalado un servidor Web o un navegador. Para esto sólo se necesita el interprete PHP. Este tipo de scripts regularmente se ejecutan desde el cron de Linux o el planificador de tareas de Windows, se pueden usar también para el procesamiento de textos. El tercer campo donde se usan los scripts de PHP es el de las aplicaciones de interfaz gráficas. Posiblemente PHP no sea el lenguaje de programación más indicado para implementar aplicaciones gráficas, pero si se quiere utilizar alguna característica avanzada en programas clientes, se puede utilizar la biblioteca para crear interfaces gráficas de usuario (PHP-GTK) para escribir estos programas. También se pueden escribir aplicaciones independientes de una plataforma. PHP-GTK es una extensión de PHP.

PHP es multiplataforma, esto quiere decir que puede ser utilizado en cualquiera de los sistemas operativos más importantes del mercado. Se incluyen muchas versiones de Linux, Unix, Microsoft Windows, Mac OS X, y probablemente muchos más. El Apache, Microsoft Internet Information Server, Personal Web Server, Netscape e iPlanet, Oreilly Website Pro Server, Caudium, Xitami, OmniHTTPd, forman parte de la lista de servidores Web que soportan PHP. PHP tiene módulos disponibles para la mayoría de los servidores. Existe libertad con PHP de elegir el Sistema Operativo y el servidor de su gusto.

PHP permite la programación procedimental y la programación orientada a objetos, aunque en cada una de las nuevas versiones se le han ido incorporando características estándares al lenguaje PHP de la programación orientada a objetos. El depósito oficial de código de PHP (PEAR) y otras bibliotecas están escritos íntegramente usando la programación orientada a objetos. HTML no es el único documento que se puede generar a partir de PHP, también se pueden crear imágenes, archivos PDF y películas Flash,

documentos XHTML y archivos XML. Estos archivos, PHP puede guardarlos en el sistema de archivos y presentarlos en la pantalla

.

Una de las características más potentes y destacable que tiene PHP, quizás la que le ha dado mayor popularidad y aceptación a nivel mundial, es el soporte que tiene para una gran cantidad de bases de datos. Resulta tarea sencilla implementar una Interfaz de la Web para una base de datos con PHP. Incluye DBX como Extensión de Abstracción de Bases de Datos, que puede usar cualquier base de datos soportada por esta extensión. El término ODBC (estándar abierto de conexión a bases de datos) también es soportado por PHP, cualquier conexión que permita este estándar puede ser usada desde PHP.

PHP soporta protocolos de comunicación como LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (en Windows) y muchos otros para comunicarse con otros servicios. Puede usar objetos java de forma transparente como si fueran de PHP y la extensión CORBA para acceder a los objetos remotos.

1.4.2 Java Script

Java Script es un lenguaje de programación que se interpreta y se ejecuta del lado del cliente o navegador. Permite agregarle una serie de efectos dinámicos a la Web. Las páginas Web incluyen los scripts, que son las instrucciones que el browser ejecutará. Lo más usual es que el código script esté embebido dentro del código HTML de la página Web. Pero resulta más práctico vincular el código a la página mientras está dentro de un fichero Java Script de extensión .js, de esta forma se evita tener que cambiar página por página el código script cuando se quiera hacer cualquier modificación de una función en Java Script.

Muchas de las funciones que tienen las aplicaciones Web se basan en Java Script. Todos los navegadores actuales para la versión 3 o superior lo admiten sin problemas. Las funciones que pueden hacerse con este lenguaje hacen más rica la navegación entre las páginas Web. En Java Script se puede trabajar con los eventos onclick, onmouseover, entre otros, que permiten interactuar con el menú, cambiar imágenes dentro de la propia página. Se pueden crear menús desplegables, herramientas de selección de color, alarmas, generar mensajes, calendarios, relojes, juegos sencillos para jugar online, efectos de imágenes y audios y muchos más.

1.4.3 XML

Extensible Markup Language (XML) es un metalenguaje extensible de etiquetas. Permite definir la gramática de lenguajes específicos para diferentes necesidades. Es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil, además sirve para estructurar, almacenar e intercambiar información.

En los documentos XML se diferencia su contenido informativo y su estructura de la presentación en papel o pantalla. Se hace explícita la estructura y los contenidos informativos de los documentos mediante la utilización de etiquetas. XML permite crear documentos portables, que puedan intercambiarse y procesarse con facilidad en sistemas informáticos heterogéneos.

1.4.4 XSLT

XSLT es parte del XSL (Lenguaje de hojas de estilo para XML). Es un estándar de la organización W3C. Su propósito es organizar la estructura de un documento XML y darle un formato de salida. Formatear objetos juega un papel fundamental en XSL a la hora de mostrar la información, pero trasformar los datos se hace cada vez más necesaria, y es donde se utiliza XSLT. Para realizar las transformaciones de un documento, XSLT utiliza una o varias reglas de plantilla unidas a un documento fuente a transformar. Estas plantillas alimentan a un procesador de XSLT, el cual realiza las transformaciones deseadas, colocando el resultado en un archivo de salida o, como en el caso de una página Web, directamente en un dispositivo de presentación, como el monitor de un usuario. Actualmente, XSLT es muy usado en la edición Web, generando páginas HTML o XHTML. La unión de XML y XSLT permite separar contenido y presentación, aumentando así la productividad.

1.4.5 Sistemas de Gestión de Bases de Datos (SGBD)

Un sistema de Gestión de Bases de Datos es una colección de datos interrelacionados y un conjunto de programas para acceder a estos datos. Su objetivo principal es proporcionar un entorno de desarrollo

conveniente y eficiente para almacenar y extraer información en la base de datos. Los SGBD permiten a los usuarios definir y mantener la base de datos, así como acceder a la misma de forma controlada

Un Sistema de Gestión de Bases de Datos debe proporcionar a los usuarios la capacidad de almacenar datos en la base de datos, acceder a ellos y actualizarlos; ocultar la organización de los ficheros y las estructuras de almacenamiento. También presentan un catálogo en el que se almacenan las descripciones de los datos y que es accesible por los usuarios. Este catálogo es lo que se denomina diccionario de datos y contiene información que describe los datos de la base de datos.

Un SGBD debe garantizar que todas las actualizaciones correspondientes a una determinada transacción se realicen, o que no se realice ninguna. Asegura que la base de datos se actualice correctamente cuando varios usuarios la están actualizando concurrentemente. Permite recuperar la base de datos en caso de que ocurra algún suceso que la dañe. Garantiza que sólo los usuarios autorizados puedan acceder a la base de datos, así como la protección contra accesos no autorizados, tanto intencionados como accidentales. Proporciona los medios necesarios para garantizar que los datos de la base de datos y los cambios que se realizan sobre ellos, sigan ciertas reglas. Permite que se mantenga la independencia entre los programas y la estructura de la base de datos. Brinda una serie de herramientas que permiten administrar la base de datos de modo efectivo. Un SGBD debe integrarse con algún software de comunicación.

1.4.6 MySQL

MySQL es un sistema de gestión de bases de datos relacionales que se encuentra entre los más populares. Aunque no es el único, se considera que es el perfecto compañero de PHP para generar sitios dinámicos con grandes volúmenes de información y acceso a bases de datos. El servidor MySQL fue desarrollado con el objetivo de manipular grandes bases de datos, mucho más rápido que las soluciones existentes. Se ha usado exitosamente en ambientes de desarrollo sumamente exigentes, por varios años, aunque se encuentra en desarrollo constante. Ofrece un conjunto rico de funciones. Su conectividad, velocidad, y seguridad hacen de MySQL un servidor apropiado para bases de datos en Internet.

Una de las razones para el rápido crecimiento de la popularidad de MySQL, es que se trata de un producto Open Source. Esto quiere decir que se puede descargar de Internet y usarlo sin pagar nada. Se

puede estudiar su código fuente y cambiarlo de acuerdo a sus necesidades. MySQL usa GPL (la Licencia Pública General GNU), que define que es lo que se puede y lo que no se puede hacer con el software para diferentes situaciones. Si uno no está de acuerdo con una la licencia GPL o tiene la necesidad de modificar el código de MySQL en una aplicación, es posible comprar una versión de MySQL con la licencia comercial.

1.5 Metodologías de desarrollo

1.5.1 UML

El lenguaje unificado de modelado (*Unified Modeling Language* UML) prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos y describe la semántica esencial de lo que estos diagramas y símbolos significan. Prescribe una notación estándar y semánticas esenciales para el modelado de un sistema orientado a objetos. Es un lenguaje para especificar y no un método o un proceso, se utiliza para definir un sistema de software, sistemas de hardware, organizaciones del mundo real, para detallar los artefactos en el sistema y para documentar. Se puede aplicar en una gran variedad de formas para soportar una metodología de desarrollo de software, como el Proceso Unificado de Rational, pero no especifica en sí mismo qué metodología o proceso usar. UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas.

1.5.2 RUP

El Rational Unified Process (RUP) es un proceso de desarrollo de software, un marco de trabajo genérico, un conjunto de metodologías adaptables al contexto y necesidades de cada organización. Puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes niveles de aptitud y diferentes tamaños de proyectos. Utiliza el UML para preparar todos los esquemas de un sistema de software. No obstante los verdaderos aspectos definitorios del proceso unificado se resumen en tres fases claves: dirigido por casos de uso, centrado en la arquitectura e iterativo e incremental.

Contiene muchas de las mejores prácticas en el desarrollo de software. Constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Le proporciona a cada miembro del equipo las pautas, plantillas y herramientas, ayudándolos a

que produzcan, dentro de un horario predecible, con un presupuesto razonable y con alta calidad para satisfacer las necesidades de los usuarios.

1.6 Herramientas a utilizar

1.6.1 Macromedia Dreamweaver 8

Es el editor de páginas Web que más desarrollo ha alcanzado, debido a esto, es el programa que más se utiliza en el sector del diseño y la programación Web. Presenta grandes funcionalidades como soporte, tanto para la edición de imágenes Web como para animación a través de su integración con otras herramientas. Este programa soporta inserción de archivos de multimedia, Java Script, para crear efectos e interactividades, hojas de estilos, capas, permitiendo que las páginas Web se realicen con mejor funcionalidad y mayor calidad.
También, incluye soporte para la creación de páginas dinámicas con servidor ASP (Active Server Page) y PHP con acceso a bases de datos para la creación de aplicaciones y diseños Web complejos. Cualquier programador, aunque sea experto en la programación de HTML siempre encontrará en este programa razones para utilizarlo, sobretodo en lo que a productividad se refiere.

1.6.2 NuSphere PHPEd

Es un potente editor de PHP tanto para personas experimentadas como para principiantes, con soporte para múltiples formatos. Resalta los distintos tags con diferentes colores para hacer mucho más fácil la programación. Esta herramienta incluye un cliente de FTP (File Transfer Protocol) y un servidor Web integrados, que se configuran según las necesidades del trabajo que se vaya a realizar.

1.6.3 Zend Estudio

Es el único Ambiente de Desarrollo Integrado (IDE) disponible para diseñadores profesionales, que abarca todos los componentes de desarrollo necesarios para el PHP, como herramientas de bases de datos. Permite acelerar el ciclo de vida de proyectos complejos. Este programa tiene varias propiedades que lo hace una herramienta potente, destacándose, soporte para Servicios Web, integración con el Internet Explorer y código PHP anidado.

1.6.4 MySQL Manager Professional

Dispone de un conjunto de herramientas para manipular y administrar bases de datos especialmente de MySQL, permitiendo otorgar y administrar privilegios de usuarios, ejecutar scripts SQL, extraer o imprimir meta data, importar, exportar datos y realizar peticiones a través de una interfaz gráfica altamente confortable.

1.6.5 MySQL-Front

Es una sencilla pero útil aplicación diseñada especialmente para desarrolladores que trabajan con MySQL. Desde el primer momento en el que se empieza a usar esta herramienta se descubre su facilidad para obtener información sobre las bases de datos, tanto de sus tablas como de su estructura y contenido. Con este programa se pueden hacer acciones básicas como añadir, borrar o modificar tablas, campos y registros, ver variables del servidor, ejecutar o interrumpir procesos, ejecutar scripts SQL, exportar tablas a SQL o a otras bases de datos y ver las propiedades avanzadas de las tablas.

1.6.6 Stylus Studio

Es un completo entorno de desarrollo integrado, realizado especialmente para mejorar la productividad en el desarrollo de sitios Web. Incluye un potente editor de XML, permite crear y probar aplicaciones Web basadas en código XSLT, XML Schema/DTD, SQL/XML y XHTML. Es reconocida como la herramienta de desarrollo XSLT más avanzada, al permitir la edición de XML en modo visual y sincronizado. Contiene un completo set de herramientas para desarrollo en XSLT, entre las que se incluyen un “debugger” XSLT y una utilidad de diseño de hojas de estilo de HTML a XSLT.

1.6.7 Rational Rose Enterprise Edition

Esta herramienta soporta de forma completa la especificación del UML. Propone la utilización de cuatro tipos de modelos para realizar un diseño del sistema; utilizando una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas, creando de esta forma un modelo completo que representa el dominio del problema y del software. Esta herramienta permite generar código en varios lenguajes de programación a partir de un diseño en UML. Además proporciona mecanismos para la realización de ingeniería inversa partiendo del código de un programa, obteniendo información sobre su diseño.

Conclusiones.

En este capítulo se profundizó en el conocimiento del Sistema Nacional de Salud y las Áreas de Salud, definiéndose la estructura organizativa y los distintos niveles de atención. Se realizó un análisis de las tecnologías a utilizar en el desarrollo del sistema propuesto y se fundamentó la elección de los lenguajes de programación, el sistema gestor de bases de datos, y la metodología de desarrollo a utilizar, teniendo en cuenta las políticas del MINSAP determinadas por el uso de herramientas no propietarias en la elaboración de sus aplicaciones.

CAPÍTULO 2: DISEÑO DEL SISTEMA

Introducción

En este capítulo se describe el flujo de trabajo de diseño del sistema, como seguimiento al flujo de trabajo del análisis. Para ello se construyen los artefactos asociados con este flujo de trabajo. Se hace una referencia del trabajo realizado en el análisis en cuanto a los requisitos funcionales y no funcionales. Se expondrá el diagrama de clases del diseño de este se hará la descripción de todas sus clases. Para la realización de los diagramas de clases del diseño se usaron los estereotipos Web tratando como clase a cada uno de los componentes del sistema.

2.1 Modelo de Diseño

El modelo de diseño está muy cercano al de implementación, lo que es natural para guardar y mantener el modelo de diseño a través del ciclo de vida completo del software.En el diseño se modela el sistema y se encuentra su forma (incluida la arquitectura) para que soporte todos los requisitos, incluyendo los no funcionales y las restricciones que se le suponen. Una entrada esencial en el diseño es el resultado del análisis, o sea el modelo de análisis, que proporciona una comprensión detallada de los requisitos. Además impone una estructura del sistema que debemos esforzarnos por conservar lo más fielmente posible cuando demos forma al sistema.

Concretamente se definen como propósitos del diseño:

- ✓ Adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, componentes reutilizables, sistemas operativos, tecnologías de distribución, concurrencia y de interfaz de usuario.
- ✓ Crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales y clases.
- ✓ Descomponer los trabajos de implementación en partes más manejables, que puedan ser llevadas a cabo por diferentes equipos de desarrollo.
- ✓ Reutilizar las interfaces entre los subsistemas, lo cual es muy útil cuando utilizamos interfaces como elementos de sincronización entre diferentes equipos de desarrollo.[5]

[5] Flujo de Trabajo Análisis y Diseño. UCI. Ciudad Habana, Cuba: s.n., 2006.

2.2 Justificación del uso de Patrones.

Los patrones de diseño se utilizan en las aplicaciones de hoy para facilitar la elección de soluciones a problemas concretos del diseño. Estos patrones fueron propuestos originalmente por el arquitecto Christopher Alexander en el contexto del diseño y construcción urbanística.

Partiendo de la definición original de Alexander, un patrón de diseño es una solución a un problema, que se usa repetidamente en contextos similares con algunas variantes en la implementación. También en el campo de la informática se puede definir un patrón de diseño, como una solución estándar para un problema común de programación, una técnica para flexibilizar el código haciéndolo satisfacer ciertos criterios, un proyecto o estructura de implementación que logra una finalidad determinada, un lenguaje de programación de alto nivel, una manera más práctica de describir ciertos aspectos de la organización de un programa, conexiones entre componentes de programas y la forma de un diagrama de objeto o de un modelo de objeto, además de ser la forma de obtener una solución a partir de la abstracción de ejemplos específicos de diseño.

Para que una solución pueda ser considerada un patrón de diseño debe ser eficaz, debe demostrar que resuelve satisfactoriamente el problema, ser reutilizable y que pueda ser aplicada en diferentes casos.
Los Patrones de Diseño (*Design Patterns*) son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces.

Según la escala o nivel de abstracción se pueden clasificar en:

- ✓ Patrones arquitecturales: Aquellos que expresan un esquema organizativo estructural fundamental para sistemas software.
- ✓ Patrones de diseño: Aquellos que expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas software.

En la aplicación se utilizan los siguientes patrones de diseño:

2.2.1 Patrón de Diseño Proxy

Se puede definir un proxy, como un objeto de fuerza, donde todas las llamadas al objeto que se deseen invocar pasen previamente a través de él. Esto permite realizar acciones que se podrían llamar transversales a la funcionalidad realizada por el objeto final invocado, como por ejemplo, trazabilidad, seguridad, transaccionalidad y otros. El patrón Proxy se utiliza como intermediario para acceder a un objeto, permitiendo controlar el acceso a él.

Cuando se quiere crear objetos que consumen muchos recursos, pero no se quieren instanciar a no ser que el cliente lo solicite o se cumplan otras condiciones determinadas; es cuando se puede utilizar este patrón de diseño. Proporciona un sustituto o representante de otro objeto para controlar el acceso a éste. Actúa como intermediario entre el objeto cliente y el objeto deseado. En ocasiones se desea retrasar la instanciación de un objeto hasta que sea necesario utilizarlo. Así el objeto proxy lo sustituye ofreciendo la misma interfaz, sólo cuando es necesario le solicita al objeto real la información que necesita. Otro motivo para su uso es sustituir a un sistema remoto, de forma que se accede al mismo a través de una clase proxy, que permite controlar el acceso.

En el Registro de las Áreas de Salud todas las peticiones llegan a un único punto, esto sirve para hacer chequeos que garantizan la seguridad (tiene que ver con el SAAA) y en un futuro para el tratamiento de la integridad referencial.

2.2.2 Fachada

El patrón de diseño Fachada sirve para proveer de una interfaz unificada sencilla que haga de intermediaria entre un cliente y una interfaz o grupo de interfaces más complejas.
Con Fachada se puede:

Hacer una biblioteca de software más fácil de usar y entender, ya que Fachada implementa métodos convenientes para tareas comunes; hacer el código que usa la librería más legible, por la misma razón; reducir la dependencia de código externo en los trabajos internos de una librería, ya que la mayoría del

código lo usa Fachada, permitiendo así más flexibilidad en el desarrollo de sistemas; envuelve una colección mal diseñada de APIs con un solo API bien diseñado.
Fachada debe utilizarse sólo para crear clases sencillas, no clases que "sirvan para todo" o "lo hagan todo".

Fachada en la aplicación: se abstrae de la capa de presentación de interactuar con las clases de PlaSer, este patrón también puede ser el que denominan algunos como “no converses con extraños” puede ser útil para cuando se decida utilizar otra forma de interactuar con Servicios Web que no sea a través de la librería PlaSer, como por ejemplo en la migración a PHP5.

2.2.3 Alta cohesión y bajo acoplamiento

2.2.3.1 Cohesión

Cuando se dice que una clase tiene una alta cohesión, se quiere decir que el objeto tiene bien delimitadas sus responsabilidades. En la programación orientada a objetos, cada objeto tiene una responsabilidad (o varias, aunque esto sería en la mayoría de las ocasiones un mal diseño) que ha de cumplir dentro de un programa.

La cohesión de un objeto significa cuán relacionadas y enfocadas están las acciones del objeto, de manera que sus responsabilidades sean pocas y limitadas a su función. Las responsabilidades de un objeto se traducen después en métodos que realizan acciones.

2.2.3.2 Acoplamiento

El acoplamiento hace referencia a las relaciones que tienen los objetos entre sí dentro de un sistema. Teóricamente, cuando una serie de objetos tienen una relación con varios objetos, cuando estos últimos son cambiados, los objetos relacionados han de verse afectados necesariamente. Por ello la situación ideal, es que cada objeto tenga las mínimas dependencias posibles con el resto del sistema. De esta forma, se pueden realizar modificaciones en partes del programa sin necesidad de cambiar la mitad del sistema. Al utilizar un bajo acoplamiento se beneficia el programador, ya que es más sencillo de reutilizar un objeto con bajo acoplamiento que otro con alto acoplamiento. El alto acoplamiento se da normalmente, cuando un objeto ha de saber demasiados detalles internos de otro para su funcionamiento, es decir, se

rompe el encapsulamiento de otro objeto. Por ello cuanto menos acoplamiento y mayor cohesión, mejor diseñado estará el sistema.
Este patrón Alta Cohesión y Bajo Acoplamiento se utiliza en el diseño de los componentes del sistema.

2.3 Definición de Elementos de Diseño

2.3.1. Subsistemas y clases del diseño

Los subsistemas son un mecanismo de organización de los artefactos del modelo de diseño en piezas más manejables. Pueden contar de clases del diseño, realización de casos de uso, interfaces y otros subsistemas. Deben ser:

- ✓ Cohesivos: sus contenidos deben estar fuertemente relacionados.
- ✓ Débilmente acoplados: minimizar las dependencias entre subsistemas.

En la figura 2.1 se describe cómo esta estructurada la aplicación en subsistemas, siguiendo la arquitectura de tres capas.

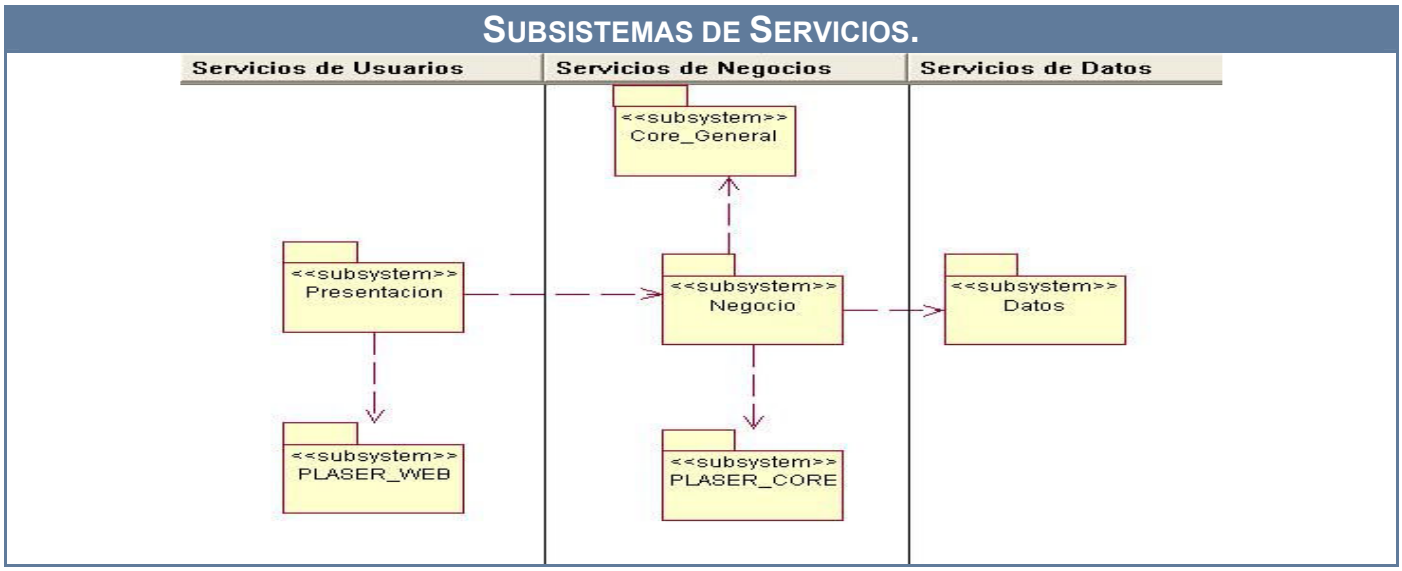


Figura 2.1 Diagrama de Subsistemas del diseño

- ✓ El subsistema ***Presentación*** agrupa todos los componentes y clases de la capa de presentación de cada una de las funcionalidades de la aplicación.
- ✓ El subsistema ***PLASER_WEB*** agrupa un conjunto de clases de la plataforma de servicios PlaSer, que intervienen en el diseño de la capa de presentación, una de ellas es la clase *Fachada.*
- ✓ El subsistema ***Negocio*** agrupa todas las clases o métodos del negocio de cada una de las funcionalidades de la aplicación.
- ✓ El subsistema ***PLASER_CORE*** agrupa un conjunto de clases de la plataforma de servicios PlaSer, que intervienen en el diseño de la capa de negocio, una de ellas es la clase *dbz_class*.
- ✓ El subsistema ***Datos*** agrupa los datos de la aplicación.

Las clases de diseño deben ser la encapsulación completa de todos los atributos y métodos que se pueden esperar, en forma razonable, que existan para la clase, es decir, que deben contener los métodos que sean suficientes para lograr el objetivo de la clase.

Las clases de diseño se definen como el proceso de refinamiento de las clases anteriores, donde se identifican los atributos y servicios necesarios para implementar algún elemento del dominio de negocios. En ellas se implementan abstracciones del negocio en un nivel más bajo, las cuales se requieren para el manejo de las clases del dominio de negocio. Las clases persistentes representan almacenamientos de datos que persistirán más allá de la ejecución el software.

2.3.2. Estereotipos Web

La notación UML tiene como una de sus características más relevantes la capacidad de absorber nuevas semánticas sin romper su lógica interna. La necesidad de implementar Servicios Web a través de complejas arquitecturas con múltiples capas de componentes y una gran dispersión geográfica de nodos, ha supuesto todo un reto al abordar su modelado y especificación. Por lo que Jim Conallen ha desarrollado desde 1998 una extensión de la notación UML denominada WAE "Web Application Extensión", que permite rentabilizar toda la gramática interna de UML para modelar aplicaciones con elementos específicos de la arquitectura de un entorno Web.

Un fichero *script* interpretable con extensión php contiene código que se ejecuta en el servidor y código que se ejecuta en el contexto de la máquina cliente. Jim Conallen estableció como parte de su extensión,

que cada fichero script interpretable podría modelarse a partir de varias clases de UML, una clase representaría el código puramente servidor, una clase representaría el código puramente cliente, y una clase representaría los formularios presentes en el código cliente. Así, su extensión presenta como elementos más significativos 3 clases de UML estereotipadas con los siguientes estereotipos “Server Page”, “Client Page”, “Form” empleados para el código servidor, código cliente y formularios respectivamente.

Las relaciones posibles a establecerse entre los tres elementos claves son:

Hasta Desde	**Client Page**	**Form**	**Server Page**
Client Page	<<Link>> , <<redirect>>	Contiene	<<Link>> , <<redirect>>
Form	Agregado por.	---	<<Submit>>
Server Page	<<Build>>, <<redirect>>		<<Redirect>>

Tabla 2.1 Relaciones entre las clases principales que conforman la extensión de UML para Web.

Las páginas servidoras se encargan de construir o generar el resultado HTML que conforma el código cliente (*build*). Los formularios envían sus datos a las páginas servidoras para procesar los pedidos (*submit*), y además forman parte del código cliente o resultado HTML; 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*). Una página cliente podría contener varios formularios e incluso estos pueden enviar sus datos a distintas páginas servidoras encargadas de procesarlos.

Una página cliente es construida por una sola página servidora, aunque podría ser un código HTML estático, en este caso, ningún código servidor la construye. Una página servidora puede, para completar su funcionamiento, incluir código existente en otra página servidora, a partir de la sentencia incluye, ejemplo *“include(../../FachadaRAS.php)”* .En este caso, podría representarse dicha relación en el diagrama de clases, y aunque no es propia de la extensión de UML propuesta por Conallen, las herramientas para el modelado y generación de código se podrían considerar, en aras de representar todas las relaciones existentes en el modelo.

En caso de la existencia de un fichero con código servidor, que contenga funciones para procesamiento, que no generan resultado HTML directo, o sea la página no construye resultado HTML; entonces se modelaría una sola clase de UML, la cual es la representación del fichero *script* interpretable final.

Las funciones JavaScript son contempladas como métodos en una clase UML, empleada para representar el código cliente. Se pueden mencionar las funciones script para realizar validaciones de los datos, entre otras.[6]

2.4 Diagramas de Clases del Diseño

Un **Diagrama de Clases** es un diagrama que muestra un conjunto de interfaces, colaboraciones y sus relaciones. Gráficamente, un diagrama de clases es una colección de nodos y arcos. Los diagramas de clases son los más utilizados en el modelado de sistemas orientados a objetos. Los diagramas de clases se utilizan para modelar la vista de diseño estática de un sistema Principalmente, esto incluye modelar el vocabulario del sistema, modelar las colaboraciones o modelar esquemas. Los diagramas de clases son importantes no sólo para visualizar, especificar y documentar modelos estructurales, sino también para construir sistemas ejecutables, aplicando ingeniería directa e inversa.

A continuación se presentan los diagramas de clases del diseño de algunos casos de uso, modelados con la extensión de UML para aplicaciones Web.

[6] Ing. Jose Angel Franco Navarro. (2004). UML en acción. Modelando Aplicaciones Web.. Ciudad de la Habana. Cuba: .

DIAGRAMA DE CLASES.

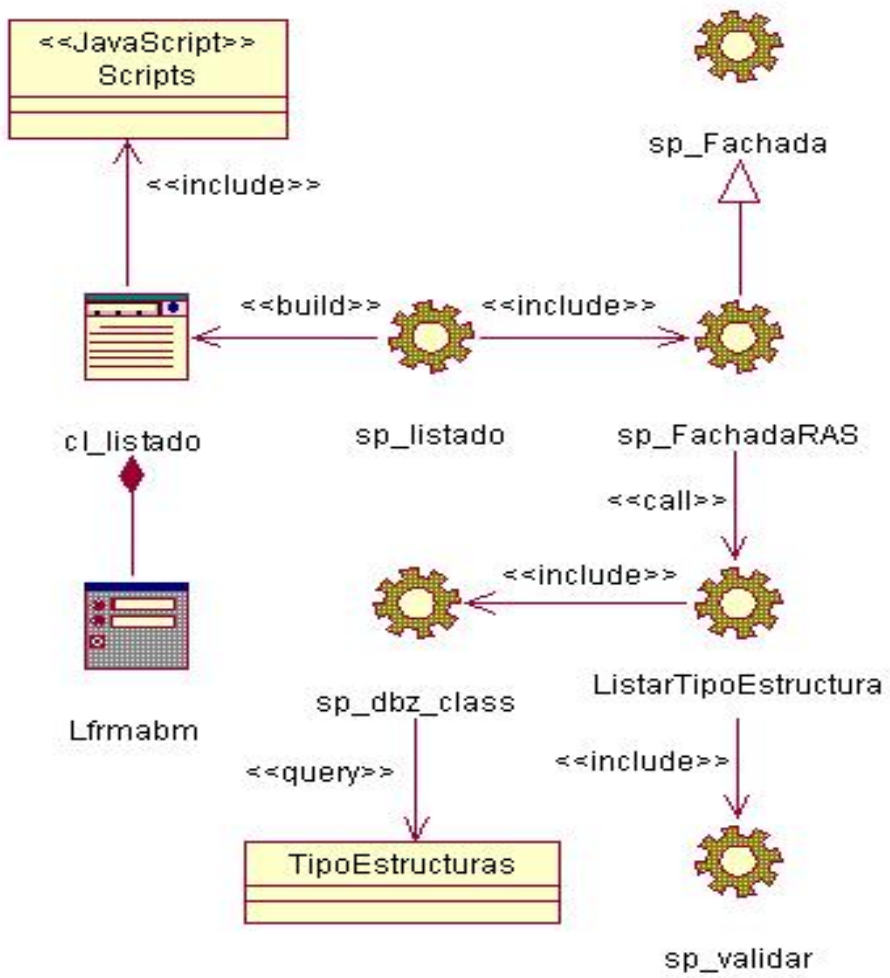


Figura 2.2 Diagrama de Clases del Diseño Caso de uso Listar Tipo de Estructuras.

DIAGRAMA DE CLASES.

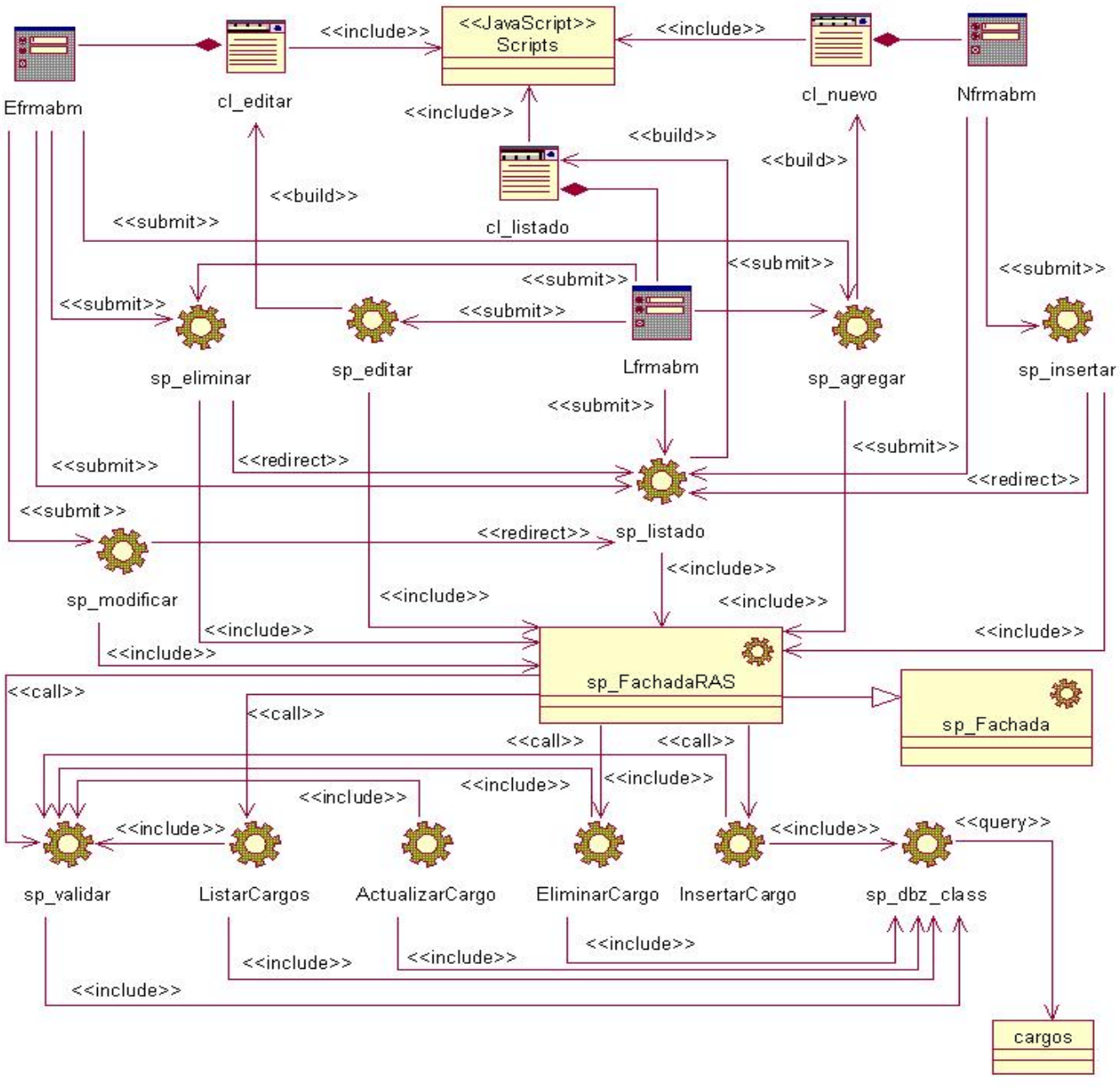


Figura 2.3 Diagrama de Clases del Diseño del Caso de uso Configurar Cargos

DIAGRAMA DE CLASES.

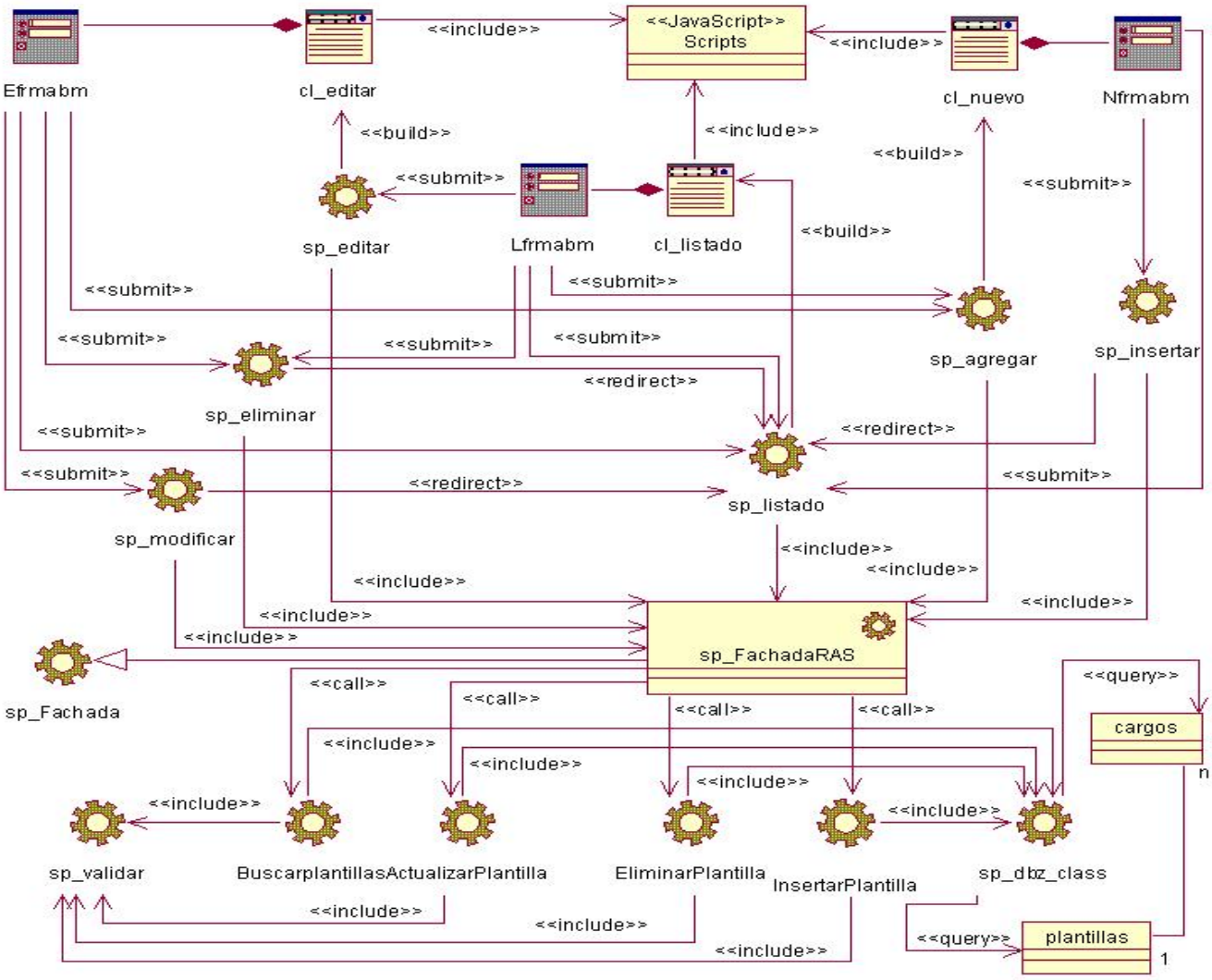


Figura 2.4 Diagrama de Clases del Diseño del Caso de uso Configurar Plantillas

DIAGRAMA DE CLASES.

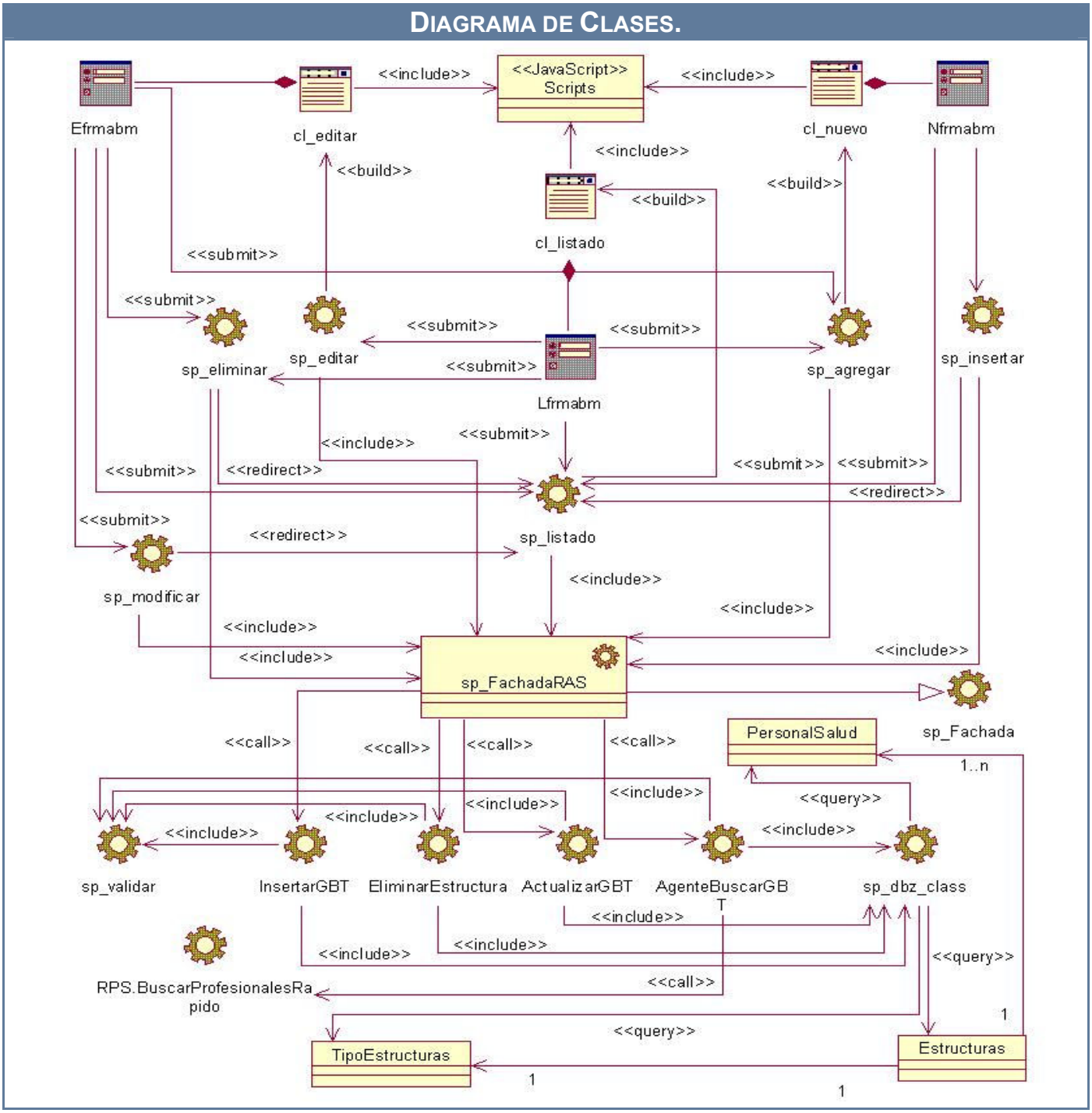


Figura 2.5 Diagrama de Clases del Diseño del Caso de uso Gestionar GBT

DIAGRAMA DE CLASES.

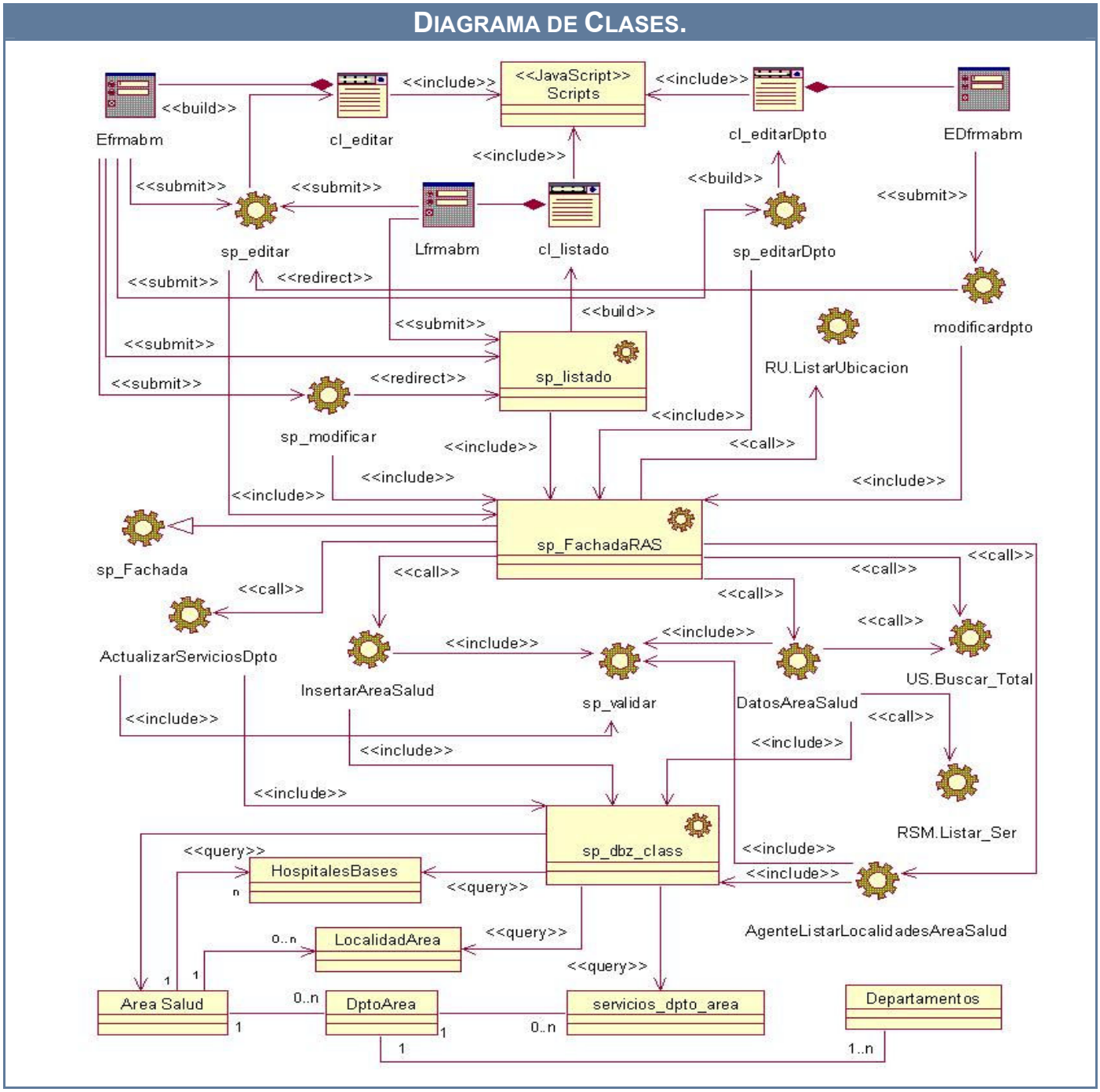


Figura 2.6 Diagrama de Clases del Diseño del Caso de uso Configurar Áreas de Salud

2.5 Descripción de las Clases del Diseño

A continuación se describen las clases utilizadas en los diagramas de clases presentados anteriormente.

2.5.1 Páginas Clientes (*Client Page*)

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 o *Browser*. Permite al usuario interactuar con la aplicación, realizar búsquedas de información e imprimir documentos en formato Adobe Acrobat (PDF) y Microsoft Excel (XSL). Si el usuario es un editor, esta página le permitirá eliminar, modificar y agregar información, en caso contrario sólo le permite realizar búsquedas y mostrar la información. Visualiza los datos en forma de páginas como si fuera un libro, de tal manera que el usuario puede moverse de una página a otra de una forma sencilla.
En cada Caso de Uso:
Muestra un listado con la información de los datos obtenidos de la base de datos. Se utiliza en los siguientes casos de usos: ✓ Listar Tipo de Estructuras. ✓ Configurar Cargos. Muestra un listado con el resultado de la búsqueda de los datos obtenidos de la base de datos. Se utiliza en los siguientes casos de uso: ✓ Configurar Plantillas. ✓ Configurar Áreas de Salud. ✓ Gestionar GBT.

Tabla 2.2 Descripción de la clase del diseño *cl_listado*.

Nombre: *cl_nuevo*
Tipo de Clase :*Client Page*
Descripción General
La clase *cl_nuevo* es una página Web que se ejecuta del lado del cliente sobre un navegador Web o *Browser*. Permite introducir y seleccionar datos para insertar en el sistema. Está disponible solamente para usuarios

editores con permisos para introducir información en cada una de las opciones de la aplicación. Una de las características de esta página es que sigue un conjunto de estándares que garantizan que el usuario que la utilice se familiarice con ella de forma rápida. Tiene un vínculo con una página de ayuda donde se le explica al usuario cómo debe introducir los datos.
En cada Caso de Uso:
Muestra una página de agregar, donde el usuario con permiso para editar puede introducir la información. Se utiliza en los siguientes casos de usos: ✓ Configurar Cargos. ✓ Configurar Plantillas. ✓ Gestionar GBT.

Tabla 2.3 Descripción de la clase del diseño *cl_nuevo*.

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 se ha seleccionado para editar desde la página cliente cl_listado. Permite modificar la información que se visualiza en la misma. Esta página muestra los datos del elemento seleccionado que no se visualizan en la página cl_listado. Si el usuario es visualizador, los controles de la página aparecen deshabilitados de manera tal que sólo pueda ver la información, en caso contrario aparecerán habilitados.
En cada Caso de Uso:
Muestra la página para editar, donde el usuario con permisos para editar puede modificar la información, si es visualizador sólo podrá ver la información. Se utiliza en los siguientes casos de usos: ✓ Configurar Cargos. ✓ Configurar Plantillas. ✓ Configurar Áreas de Salud. ✓ Gestionar GBT.

Tabla 2.4 Descripción de la clase del diseño *cl_editar*.

2.5.2 Páginas Servidoras de la Capa de Presentación

Nombre: *sp_listado*
Tipo de Clase :*Server Page*
Descripción General
La clase *sp_listado* es una página Web que se ejecuta del lado del servidor. 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.
En cada Caso de Uso:
Construye una página cliente que le muestra un listado de la información en los siguientes casos de uso: ✓ Listar Tipo de Estructuras. ✓ Configurar Cargos. ✓ Configurar Plantillas. ✓ Configurar Áreas de Salud. ✓ Gestionar GBT.

Tabla 2.5 Descripción de la clase del diseño *sp_listado*.

Nombre: *sp_agregar*
Tipo de Clase :*Server Page*
Descripción General
La clase *sp_agregar* es una página Web que se ejecuta del lado del servidor. Su actividad fundamental es construir la página cliente cl_nuevo. Aplica un documento XSLT (nuevo.xsl) a otro documento XML para transformarlo y mostrarlo al usuario en formato XHTML.
En cada Caso de Uso:
Crea una página cliente de agregar, donde el usuario con permisos para editar podrá introducir información. Se utiliza en los siguientes casos de usos: ✓ Configurar Cargos. ✓ Configurar Plantillas. ✓ Gestionar GBT.

Tabla 2.6 Descripción de la clase del diseño *sp_agregar*.

Nombre: *sp_editar*
Tipo de Clase :*Server Page*
Descripción General
La clase *sp_editar* es una página Web que se ejecuta del lado del servidor. Su actividad fundamental es construir la página cliente *cl_editar*. Aplica un documento XSLT (editar.xsl) a otro documento XML para transformarlo y mostrarlo al usuario en formato XHTML.
En cada Caso de Uso:
Crea una página cliente de editar, donde el usuario con permisos para editar podrá modificar la información. Si el usuario no es editor se le construye una página en la que sólo podrá ver la información. Se utiliza en los siguientes casos de uso: ✓ Configurar Cargos. ✓ Configurar Plantillas. ✓ Configurar Áreas de Salud. ✓ Gestionar GBT.

Tabla 2.7 Descripción de la clase del diseño *sp_editar*.

Nombre: *sp_insertar*
Tipo de Clase :*Server Page*
Descripción General
La clase *sp_insertar* es una página Web que se ejecuta del lado del servidor. Recibe y valida los datos que se envían desde la página cliente *cl_nuevo*. Codifica las descripciones y los números los convierte en enteros. Se comunica con el método del negocio a través de la clase sp_FachadaRAS y redirecciona su ejecución a la página *sp_listado*.
En cada Caso de Uso:
Permite agregar información que se envía desde el cliente. Se utiliza en los siguientes casos de usos: ✓ Configurar Cargos. ✓ Configurar Plantillas. ✓ Gestionar GBT.

Tabla 2.8 Descripción de la clase del diseño *sp_insertar.*

Nombre: *sp_modificar*
Tipo de Clase :*Server Page*
Descripción General
La clase *sp_modificar* es una página Web que se ejecuta del lado del servidor. Recibe y valida los datos que se envían desde la página cliente *cl_editar*. Codifica las descripciones y los números los convierte en enteros. Se comunica con el método del negocio a través de la *sp_FachadaRAS* y redirecciona su ejecución a la página *sp_listado* para mostrar el dato actualizado.
En cada Caso de Uso:
Permite modificar información que se envía desde el cliente. Se utiliza en los siguientes casos de usos: ✓ Configurar Cargos. ✓ Configurar Plantillas. ✓ Configurar Áreas de Salud. ✓ Gestionar GBT.

Tabla 2.9 Descripción de la clase del diseño *sp_modificar.*

Nombre: *sp_eliminar*
Tipo de clase :*Server Page*
Descripción general
La clase *sp_eliminar* es una página Web que se ejecuta del lado del servidor. Recibe el id del elemento que se desea eliminar desde la página cliente *cl_listado*, lo convierte en entero e invoca al método del negocio a través de la *sp_FachadaRAS* y redirecciona su ejecución a la página sp_listado.
En cada Caso de Uso:
Permite eliminar información que se envía desde el cliente. Se utiliza en los siguientes casos de usos: ✓ Configurar Cargos. ✓ Configurar Plantillas. ✓ Gestionar GBT.

Tabla 2.10 Descripción de la clase del diseño *sp_eliminar.*

La clase ***script*** representa un fichero Java Script que se ejecuta del lado del cliente, su código es interpretado por cualquier navegador Web. Las funciones implementadas dentro de este fichero se utilizan en la validación de los parámetros de entrada, para validar que se introduzcan los datos obligatorios y evitar la entrada de caracteres especiales. Se emplean en combinación del código HTML de la página, para brindarle al usuario una interacción dinámica con la página Web (DHTML). A través de estas funciones también se puede acceder a las páginas del servidor.

2.5.3 Páginas servidoras de la Capa de Negocio

Nombre	*ListarTipoEstructura*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Tipo Estructuras*
Parámetros de Entrada	
Nombre	**Tipo**
Offset	int
Cantidad	int
Tipo_salida	int
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Recibe un arreglo de datos con los parámetros de entrada. Esta página establece una conexión a través de la clase de acceso a datos *dbz_class* con el servidor de bases de datos y ejecuta una consulta SQL. La responsabilidad fundamental de esta página es retornar un listado con la información de los Tipos de Estructuras. En dependencia del valor del parámetro de entrada *tipo_salida* será el formato de la respuesta de este método. Su implementación permite retornar documentos en formatos XML, Acrobat Reader (PDF) y Microsoft Excel (XSL). Una vez terminadas las consultas SQL, se cierra la conexión con la base de datos.	
Parámetros de Salida	
Nombre	**Tipo**
Resultado	array
○ Id	○ int
○ descripción	○ string
Offset	int

Total	int
Cantidad	int

Tabla 2.11 Descripción de la clase del diseño *ListarTipoEstructura.*

Nombre	*InsertarCargo*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Cargos*
Parámetros de Entrada	
Nombre	**Tipo**
Descripcion	string
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida cada uno de los parámetros de entrada que le llegan desde el cliente que esté consumiendo este servicio Web. Si los valores de los parámetros de entrada son correctos, establece una conexión con la base de datos a través de la clase de acceso a datos *dbz_class.* Este método tiene como responsabilidad ejecutar las consultas necesarias para insertar los datos de un Cargo. Una vez terminada la ejecución de las consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si este método detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se lograron insertar correctamente los datos de un Cargo.	

Tabla 2.12 Descripción de la clase del diseño *InsertarCargo*.

Nombre	*EliminarCargo*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Cargos*
Parámetros de Entrada	
Nombre	**Tipo**
id_cargo	Int
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida el parámetro de entrada que le llega desde el cliente que solicite este servicio Web. Si el valor que recibe es un entero positivo establece una conexión directa con el servidor de bases de datos a	

través de la clase de acceso a datos *dbz_class*. Este método tiene la responsabilidad de ejecutar las consultas necesarias para eliminar los datos del cargo recibido como parámetro, una vez verificada su existencia en la base de datos. Terminada la ejecución de las consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si en el proceso de ejecución del método se detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se eliminaron correctamente los datos de un cargo.

Tabla 2.13 Descripción de la clase del diseño *EliminarCargo*.

Nombre	*ListarCargo*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Cargos*
Parámetros de Entrada	
Nombre	**Tipo**
Offset	int
Cantidad	int
Tipo_salida	int
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Recibe un arreglo de datos con los parámetros de entrada, los cuales son validados. Esta página establece una conexión a través de la clase de acceso a datos *dbz_class* con el servidor de bases de datos. Tiene como responsabilidad ejecutar una consulta SQL para retornar un listado con la información de los cargos existentes. En dependencia del valor del parámetro de entrada *tipo_salida* será el formato de la respuesta de este método. Su implementación permite retornar documentos en formatos XML, Acrobat Reader (PDF) y Microsoft Excel (XSL). Este servicio Web puede ser consumido desde cualquier componente, tanto desde la capa de presentación como desde la capa de negocio. Una vez terminadas las consultas SQL, se cierra la conexión con la base de datos.	
Parámetros de Salida	
Nombre	**Tipo**

Resultado	array
○ id	○ int
○ descripción	○ string
Offset	int
Total	int
Cantidad	int

Tabla 2.14 Descripción de la clase del diseño *listarCargos*.

Nombre	*ActualizarCargo*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Cargos*
Parámetros de Entrada	
Nombre	**Tipo**
id_cargo	int
Descripcion	string
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida cada uno de los parámetros de entrada que le llegan desde el cliente que esté consumiendo este servicio Web. Si los valores de los parámetros de entrada son correctos, establece una conexión con la base de datos a través de la clase de acceso a datos *dbz_class* y ejecuta las consultas que actualizan los datos de un cargo. Si en el proceso de actualizar los datos ocurre algún error generalmente causado por las reglas de la integridad referencial de la base de datos, se detiene el proceso y todo lo que se había hecho hasta el momento retorna a su estado inicial. Una vez terminada la ejecución de consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si este método detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se modificaron correctamente los datos de un cargo.	

Tabla 2.15 Descripción de la clase del diseño *ActualizarCargo*.

Nombre	*BuscarPlantillas*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Plantillas*
Parámetros de Entrada	
Nombre	**Tipo**
Descripción	string
id _ cargo	int
Tipo_salida	int
Offset	int
Cantidad	int
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Recibe un arreglo de datos con los parámetros de entrada, los valida y según los criterios de búsqueda que se envían desde el cliente forma una consulta SQL dinámica. Esta página establece una conexión a través de la clase de acceso a datos *dbz_class* con el servidor de bases de datos y ejecuta las consultas SQL. En dependencia del valor que se le envíe por el parámetro de entrada *tipo_salida* así será el formato de la respuesta de este método. Su implementación permite retornar documentos en formatos XML, Acrobat Reader (PDF) y Microsoft Excel (XSL). Este servicio Web puede ser consumido desde cualquier componente, tanto desde la capa de presentación como desde la capa de negocio. Una vez terminada las consultas SQL, cierra la conexión con la base de datos. La responsabilidad fundamental de esta página es hacer búsquedas de información relacionada con las plantillas y sus cargos.	
Parámetros de Salida	
Nombre	**Tipo**
Resultado	array
○ id	○ int
○ descripción	○ string
○ cargos	○ array
○ id_cargo	○ int
○ descripción	○ string
○ cantidad_cargo	○ int
Offset	Int

Total	int
Cantidad	int

Tabla 2.16 Descripción de la clase del diseño *BuscarPlantillas*.

Nombre	*InsertarrPlantilla*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Plantillas*
Parámetros de Entrada	
Nombre	**Tipo**
Descripcion	string
id_cargos	array
Cantidad	array
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida cada uno de los parámetros de entrada que le llegan desde el cliente que esté consumiendo este servicio Web. Si los valores de los parámetros de entrada son correctos, establece una conexión con la base de datos a través de la clase de acceso a datos *dbz_class* y ejecuta las consultas necesarias para insertar los datos de una plantilla. Si en el proceso de insertar los datos ocurre algún error, generalmente causado por las reglas de la integridad referencial de la base de datos, se detiene el proceso y se deshacen todas las operaciones realizadas hasta el momento de ocurrir el error; esto es posible porque este servicio Web tiene presente el uso de las transacciones, que evitan la inconsistencia de los datos. Una vez terminada la ejecución de consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si este método detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se insertaron correctamente los datos de una plantilla con sus cargos asociados.	

Tabla 2.17 Descripción de la clase del diseño *InsertarPlantillas*.

Nombre	*EliminarPlantilla*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Plantillas*
Parámetros de Entrada	

Nombre	**Tipo**
id_plantilla	Int
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida el parámetro de entrada que le llega desde el cliente que solicite este servicio Web. Si el valor que recibe es un entero positivo establece una conexión directa con el servidor de bases de datos a través de la clase de acceso a datos *dbz_class*, valida la existencia de este valor. De existir una plantilla con ese valor, ejecuta las consultas necesarias para eliminar los datos de la plantilla seleccionada. Una vez terminada la ejecución de consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si en el proceso de ejecución del método se detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se eliminaron correctamente los datos de una plantilla y de sus cargos.	

Tabla 2.18 Descripción de la clase del diseño *EliminarPlantillas*.

Nombre	*ActualizarPlantilla*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Plantillas*
Parámetros de Entrada	
Nombre	**Tipo**
id_plantilla	int
Id_cargos	array
Cantidad	array
Descripcion	string
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida cada uno de los parámetros de entrada que le llegan desde el cliente que esté consumiendo este servicio Web. Si los valores de los parámetros de entrada son correctos, establece una conexión con la base de datos a través de la clase de acceso a datos *dbz_class* y ejecuta las consultas que actualizan los datos de una plantilla. Si en el proceso de actualizar los datos ocurre algún error, generalmente causado por las reglas de la integridad referencial de la base de datos, se detiene el proceso y se deshacen todas las operaciones realizadas hasta el momento de ocurrir el error; esto es	

posible porque este servicio Web tiene presente el uso de las transacciones, que evitan la inconsistencia de los datos. Una vez terminada la ejecución de consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si este método detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se modificaron correctamente los datos de una plantilla con sus cargos asociados.

Tabla 2.19 Descripción de la clase del diseño *ActualizarPlantillas*.

Nombre	*InsertarGBT*
Tipo de Clase	*Server Page*
Caso de Uso	*Gestionar Grupo Básico de Trabajo*
Parámetros de Entrada	
Nombre	**Tipo**
id_plantilla	int
Id_cargos	array
Descripcion	string
id_unidad	int
Integrantes	array
Cargos	array
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida cada uno de los parámetros de entrada que le llegan desde el cliente que esté consumiendo este servicio Web. Si los valores de los parámetros de entrada son correctos, establece una conexión con la base de datos a través de la clase de acceso a datos *dbz_class* y ejecuta las consultas necesarias para insertar los datos de un Grupo Básico de Trabajo. Si en el proceso de insertar los datos ocurre algún error, generalmente causado por las reglas de la integridad referencial de la base de datos, se detiene el proceso y se deshacen todas las operaciones realizadas hasta el momento de ocurrir el error; esto es posible porque este servicio Web tiene presente el uso de las transacciones, que evitan la inconsistencia de los datos. Una vez terminada la ejecución de consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si este método detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se lograron insertar correctamente los datos de	

un Grupo Básico de Trabajo.

Tabla 2.20 Descripción de la clase del diseño *InsertarGBT*.

Nombre	*ActualizarGBT*
Tipo de Clase	*Server Page*
Caso de Uso	*Gestionar Grupo Básico de Trabajo*
Parámetros de Entrada	
Nombre	**Tipo**
id_plantilla	int
Cargos	array
id_estructura	Int
Descripcion	string
Integrantes	array
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida cada uno de los parámetros de entrada que le llegan desde el cliente que esté consumiendo este servicio Web. Si los valores de los parámetros de entrada son correctos, establece una conexión con la base de datos a través de la clase de acceso a datos *dbz_class* y ejecuta las consultas que actualizan los datos de una plantilla. Si en el proceso de actualizar los datos ocurre algún error, generalmente causado por las reglas de la integridad referencial de la base de datos, se detiene el proceso y se deshacen todas las operaciones realizadas hasta el momento de ocurrir el error; esto es posible porque este servicio Web tiene presente el uso de las transacciones, que evitan la inconsistencia de los datos. Una vez terminada la ejecución de consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si este método detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se modificaron correctamente los datos del Grupo Básico de Trabajo.	

Tabla 2.21 Descripción de la clase del diseño *ActualizarGBT*.

Nombre	*AgenteBuscarGBT*
Tipo de Clase	*Server Page*
Caso de Uso	*Gestionar Grupo Básico de Trabajo*

Parámetros de Entrada	
Nombre	**Tipo**
id_unidad	int
Ape1	array
Ape2	string
Nombre	string
Registro	int
id_estructura_gbt	int
id_plantilla	int
Descripcion	string
id_cargo	int
Orden	int
Ordenarpor	int
Cantidad	int
Offset	int
id_estructura_ebs	int
Tipo_salida	int
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Tiene como condición fundamental que se le deben pasar todos los parámetros. Si los valores que recibe son correctos, establece una conexión directa con el servidor de bases de datos a través de la clase de acceso a datos *dbz_class.* Este método es un agente que tiene como responsabilidad fundamental realizar búsquedas de información en la capa de negocio que estén relacionadas con los Grupos Básicos de Trabajo. Para realizar la búsqueda de estos datos establece comunicación con el Registro Personal de Salud. En dependencia del valor que se le envíe por el parámetro de entrada *tipo_salida* así será el formato de la respuesta de este método. Su implementación permite retornar documentos en formatos XML, Acrobat Reader (PDF) y Microsoft Excel (XSL). Este servicio Web puede ser consumido desde cualquier componente, tanto desde la capa de presentación como desde la capa de negocio. Una vez terminada las consultas SQL, cierra la conexión con la base de datos.	
Parámetros de Salida	
Estructura	array
○ id_estructura	○ Int
○ estructura	○ string

○ id_plantilla	○ Int
○ Plantilla	○ String
○ Id_unidad	○ Int
○ profesional	○ array
○ id_profesional	○ int
○ registromedico	○ int
○ nombre	○ string
○ ape1	○ string
○ ape2	○ string
○ id_tipoprofesional	○ int
○ tipoprofesional	○ string
○ id_cargo	○ int
Offset	
Cantidad	
Total	

Tabla 2.22 Descripción de la clase del diseño *AgenteBuscarGBT*.

Nombre	*InsertarAreasSalud*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Áreas de Salud*
Parámetros de Entrada	
Nombre	**Tipo**
id_unidad	int
código_area_salud	int
Km	Int
id_hospitales	array
id_localidad	array
id_dptos	array
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida los parámetros de entrada que le llegan desde el cliente que solicite este servicio Web. Si los valores que recibe son correctos, establece una conexión directa con el servidor de bases de datos a través de la clase de acceso a datos *dbz_class.* Este método tiene como responsabilidad la inserción de	

los datos que caracterizan a un Área de Salud como las localidades, departamentos, hospitales bases así como el código del área y su extensión geográfica dada en km cuadrados. En su proceso de ejecución comprueba primero la existencia de los datos que se desean insertar; si no existen en la base de datos, los inserta o de lo contrario realiza una actualización de los mismos. Una vez terminada las consultas SQL, cierra la conexión con la base de datos. Si en el proceso de ejecución del método se detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se modificaron correctamente los datos del Área de Salud.

Tabla 2.23 Descripción de la clase del diseño *InsertarAreaSalud*.

Nombre	*DatosAreaSalud*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Áreas de Salud*
Parámetros de Entrada	
Nombre	**Tipo**
id_unidad	int
Id_provincia	int
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida los parámetros de entrada que le llegan desde el cliente que solicite este servicio Web. Si los valores que recibe son correctos, establece una conexión directa con el servidor de bases de datos a través de la clase de acceso a datos *dbz_class.* Este método tiene como responsabilidad la búsqueda de los datos de un Área de Salud, relacionados con sus hospitales base, localidades, departamentos y servicios médicos. Para realizar la búsqueda de estos datos establece comunicación con los componentes: Registro de Unidades de Salud, Registro de Ubicación y Registro de Servicios Médicos. Una vez terminada la ejecución de las consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor.	
Parámetros de Salida	
Departamentos	array
○ id_dpto	○ int
○ descripcion	○ string

DepartamentosArea	array
○ id_dpto	○ string
○ descripcion	○ int
Areasalud	array
○ id_unidad	○ int
○ codigo_area_salud	○ int
○ km	○ int
ResultHospitalesBasesArea	array
○ id_unidad	○ int
○ nombre_unidad	○ string
ResultHospitalesBases	array
○ id_unida	○ int
○ nombre_unidad	○ string
ResultLocalidades	array
○ id_localidad	○ string
○ descripción	○ string

Tabla 2.24 Descripción de la clase del diseño *DatosAreaSalud*.

Nombre	*ActualizarServiciosDpto*
Tipo de Clase	*Server Page*
Caso de Uso	*Configurar Áreas de Salud*
Parámetros de Entrada	
Nombre	**Tipo**
id_servicios	Array
Id_dpto	
Descripción General	
Esta clase es una página Web que se ejecuta del lado del servidor. Forma parte de la capa de negocio del sistema. Valida cada uno de los parámetros de entrada que le llegan desde el cliente que esté consumiendo este servicio Web. Si los valores de los parámetros de entrada son correctos, establece una conexión con la base de datos a través de la clase de acceso a datos *dbz_class.* Este método tiene como responsabilidad la ejecución de las consultas que actualizan los datos de un departamento con sus servicios médicos. Si en el proceso de actualizar los datos ocurre algún error, generalmente causado por	

las reglas de la integridad referencial de la base de datos, se detiene el proceso y se deshacen todas las operaciones realizadas hasta el momento de ocurrir el error; esto es posible porque este servicio Web tiene presente el uso de las transacciones, que evitan la inconsistencia de los datos. Una vez terminada la ejecución de las consultas SQL, se cierra la conexión con la base de datos para no sobrecargar el servidor. Si este método detecta algún error relacionado con los parámetros de entrada o las consultas a la base de datos, retorna una excepción que indica el error cometido, de lo contrario retorna el valor 1 que indica que se modificaron correctamente los datos de un departamento del Área de Salud y sus Servicios Médicos.

Tabla 2.25 Descripción de la clase del diseño *ActualizarServiciosDpto*.

2.5.4 Otras clases utilizadas

Clase	Propósito
dbz_class	Clase utilizada para establecer la conexión con bases de datos MySql, usa el módulo dbx de PHP para su funcionalidad. Crea un objeto conexión que permite hacer consultas, y recuperar los resultados; insertar, eliminar y actualizar datos. Esta clase está en la capa de negocio y forma parte de la plataforma de servicios PLASER.
Fachada	Clase general que se encuentra en la plataforma de servicios PLASER, permite aplicar un encapsulamiento a esta librería de clases, creándose de esta forma un nivel de abstracción, logrando una mejor arquitectura del sistema.
FachadaRAS	Clase que hereda de Fachada, se aplica en la capa de presentación, disminuyendo así la carga de negocio en gran medida. Esta clase implementa el patrón Fachada, de esta forma la aplicación sólo le hará las peticiones a la clase Fachada. Es la clase principal de la capa de presentación, contiene las funcionalidades del sistema que permiten acceder a los métodos del negocio. Cualquier solución general del módulo debe implementarse en ella.

plantilla	Almacena las plantillas que se pueden asignar a un GBT o un EBS en el área de salud.
cargos	Amacena los cargos que se asignarán al personal de los GBT o EBS en el área de salud
personal_salud	Almacena los datos del personal de la salud que integra un GBT ó un EBS en el área de salud.
Tipo_estructura	Almacena los tipos de estructuras que existen en las áreas de salud.
estructuras	Almacena los datos de las estructuras del área de salud (datos comunes a los GBT y EBS).
hospitales_base	Almacena los hospitales base que tiene cada área de salud.
localidad_area	Almacena las localidades que tiene cada área de salud.
departamentos	Almacena los departamentos de las áreas de salud.
area_salud	Amacena los datos de las áreas de salud.
servicios_dpto_area	Amacena los servicios médicos que se brindan en los departamentos de cada área de Salud.
departamentos_area	Almacena los departamentos que tiene cada área de salud.

Tabla 2.26 Descripción de otras clases.

Conclusiones

En el presente capítulo se modela todo el flujo de trabajo de diseño, como continuación del análisis propuesto para la solución del sistema. Se hace una fundamentación de la utilización de patrones y elementos del diseño. Se presentan algunos artefactos, de los obtenidos en este flujo: los diagramas de clases del diseño de algunos casos de uso y la descripción de las clases utilizadas en estos. Además, se argumenta el uso de los estereotipos Web, mediante un acercamiento a la descripción y a la funcionalidad del sistema, desde el punto de vista del programador.

.

CAPÍTULO 3: IMPLEMENTACIÓN

Introducción

En este capítulo, se desarrolla una secuencia del flujo de análisis y diseño. Se definieron mediante el uso de los estereotipos Web, un conjunto de clases o ficheros que serán tratados como componentes de la aplicación. Se fundamenta la integración del Registro de las Áreas de Salud con otros componentes del Sistema de Información para la Salud (SISalud). Haciendo una breve descripción de los servicios Web y los métodos o agentes más complejos de la aplicación. Además, se presentan los diagramas de componentes, el diagrama de despliegue de la aplicación y un estudio de los estándares de diseño, codificación y tratamiento de excepciones.

3.1 Dependencias y Relaciones con otros Sistemas

La utilización de una arquitectura basada en componentes y orientada a servicios, donde cada módulo es un componente con funcionalidad propia y que utiliza los servicios brindados por otros componentes hacen que el funcionamiento del Registro de Áreas de Salud (RAS) dependa de otros componentes de SISalud, los cuales se detallan a continuación. La posibilidad de utilizar los servicios Web brindados por otros componentes, garantiza la reutilización de los datos y evita la duplicidad de la información.

3.1.1 Componente de Seguridad (SAAA)

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

La autenticación, autorización y seguridad en el RAS se implementó utilizando el SAAA, cada petición del usuario, autorizada o no, queda registrada en el sistema.
El certificado digital se utiliza para configurar las opciones del RAS, a las que tiene acceso el usuario en dependencia de su nivel y sus permisos. Si el usuario es un editor de un Área de Salud tendrá acceso a las opciones que gestionan la información de su área. Si el usuario es un editor nacional podrá acceder a las opciones de Configurar Sistema. Si el usuario es un visualizador, sólo podrá ver la información en las diferentes opciones, dependiendo del nivel en que se encuentre.

3.1.2 Registro de Unidades de Salud (RUS)

Este Registro tiene almacenada la información correspondiente a las unidades de salud de todo el país. Brinda un conjunto de servicios Web que facilitan la búsqueda de información relacionada con los procesos de negocio que se gestionan en este codificador. Una de las funcionalidades del RUS es la de relacionar un tipo de unidad de salud con sus atributos (cualitativos o de capacidad), esta relación incluye también la combinación de estos atributos con las unidades de salud que los posean. Los Hospitales Rurales y los Policlínicos son tipos de unidades de salud que contemplan el Programa del Médico y la Enfermera de la Familia, a las mismas se les puede agregar un atributo cualitativo que indique si son o no Áreas de Salud.

Uno de los servicios Web disponibles del RUS y que se utiliza en el RAS, es el que permite la búsqueda de las unidades de salud. La información que ofrece este servicio se realiza teniendo en cuenta un conjunto de parámetros de entrada como el nombre, la ubicación geográfica (Provincia o Municipio) o los atributos cualitativos de las unidades de salud. El RAS se comunica con el RUS para buscar los datos de las Áreas de Salud y de sus Hospitales Bases. Estas búsquedas se realizan teniendo en cuenta el nivel en que se haya autenticado un usuario en el sistema, ejemplo, si el usuario es del nivel de unidad de salud y la unidad a la que pertenece es un área de salud, se buscarán los datos de dicha unidad en el RUS.

3.1.3 Registro de Ubicación (RU)

El RU es uno de los componentes no médicos del Registro Informatizado de Salud. El mismo gestiona la información de las Provincias, Municipios, Localidades, Calles y Manzanas del país. Los servicios que ofrece este componente se utilizan en el RAS para solicitar los criterios de búsqueda en las diferentes

opciones, relacionados con la información que se gestiona en RU. También se emplean para introducir las direcciones de las casas y centros laborales que existen en una población, con estos datos los integrantes de los Equipos Básico de Salud podrán conocer la ubicación que tienen sus poblaciones. Además se utilizan para introducir la dirección de los locales (de consultas y viviendas).

3.1.4 Registro de Localidades (RL)

El Registro de Localidades, de forma centralizada gestiona la información de los Consejos Populares, Circunscripciones, Zonas CDR y CDR de cada municipio del país. Los servicios que ofrece este componente se utilizan en el RAS para solicitar los criterios de búsqueda en las diferentes opciones, relacionados con la información que se gestiona en RL. En el RAS para definir las poblaciones en un área de salud, es necesario conocer los Consejos Populares que estas abarcan, así como los CDR a los que pertenecen las casas que conforman las poblaciones del área.

3.1.5 Registro de Personal de la Salud (RPS)

Este componente tiene la funcionalidad de gestionar la información de todos los profesionales de la salud del país, ya sean médicos o no médicos. En las Áreas de Salud se encuentran los Grupos Básicos de Trabajo y Equipos Básicos de Salud. Los mismos están compuestos por especialistas y profesionales de la salud, que ejercen sus funciones para prevenir y resolver los problemas de la salud de la población. El RPS tiene implementado un conjunto de servicios Web que brindan la información que se procesa en este componente. Estos servicios Web son utilizados por el RAS para gestionar la información de las estructuras organizativas y funcionales (GBT y EBS) que operan en las Áreas de Salud.

3.1.6 Registro de Estudiantes (RE)

Este módulo es el encargado de gestionar la información de los estudiantes de medicina que pueden ser ubicados en las Áreas de Salud como integrantes de los Equipos Básicos de Salud, y que bajo la dirección de los Grupos Básicos de Trabajo y la supervisión de profesionales de la salud, ejercen las funciones del médico como si fueran graduados de Medicina General Integral. Estos estudiantes pertenecen a una Facultad de Medicina, donde ejercen sus actividades docentes y están asignados a una unidad de salud donde vinculan la docencia con la práctica. El RAS utiliza los servicios Web del RE para ubicarlos en los EBS en las diferentes Áreas de Salud.

3.1.7 Registro de Servicios Médicos (RSM)

El RSM tiene la responsabilidad de gestionar los Servicios Médicos que se brindan a la población en las diferentes Especialidades, los cuales sirven para dar solución a los problemas de salud. Este codificador brinda servicios Web que permiten la obtención de las diferentes Especialidades médicas y de los Servicios que ofrece cada especialidad. El RAS en su configuración, define los Servicios Médicos que se brindan por cada departamento en las Áreas de Salud.

3.2 Modelo de Implementación

En el Modelo de Implementación en [RUP 2003] se define y realizan los siguientes objetivos:

- ✓ Definir la organización del sistema en términos de Subsistemas de Implementación organizados en capas.
- ✓ Implementar los elementos de diseño en términos de "Elementos de Implementación" (ficheros Fuentes, binarios, ejecutables y otros).
- ✓ Probar los componentes desarrollados independientemente como unidades.
- ✓ Integrar los resultados producidos por desarrolladores independientes o equipos en un sistema ejecutable.

3.2.1 Diagrama de Componentes

Un componente es una parte modular de un sistema, desplegable y reemplazable que encapsula implementación y un conjunto de interfaces, proporciona la realización de los mismos. Un componente típicamente contiene clases y puede ser implementado por uno o más artefactos (ficheros ejecutables, binarios, etc).

Un diagrama de componentes muestra un conjunto de elementos tales como componentes, subsistemas de implementación y sus relaciones. Permite modelar la gestión de la configuración de los archivos de código fuente y las versiones, así como las bibliotecas.

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. Un

subsistema de implementación es una colección de componentes y otros subsistemas de implementación usados para estructurar el modelo de implementación y dividirlos en pequeñas partes que pueden ser integradas y probadas de forma separada

.

El uso más importante de los diagramas de componentes es mostrar la estructura de alto nivel del modelo de implementación, especificando:

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

3.2.2 Distribución general de los subsistemas de implementación

DIAGRAMA DE IMPLEMENTACIÓN RAS.

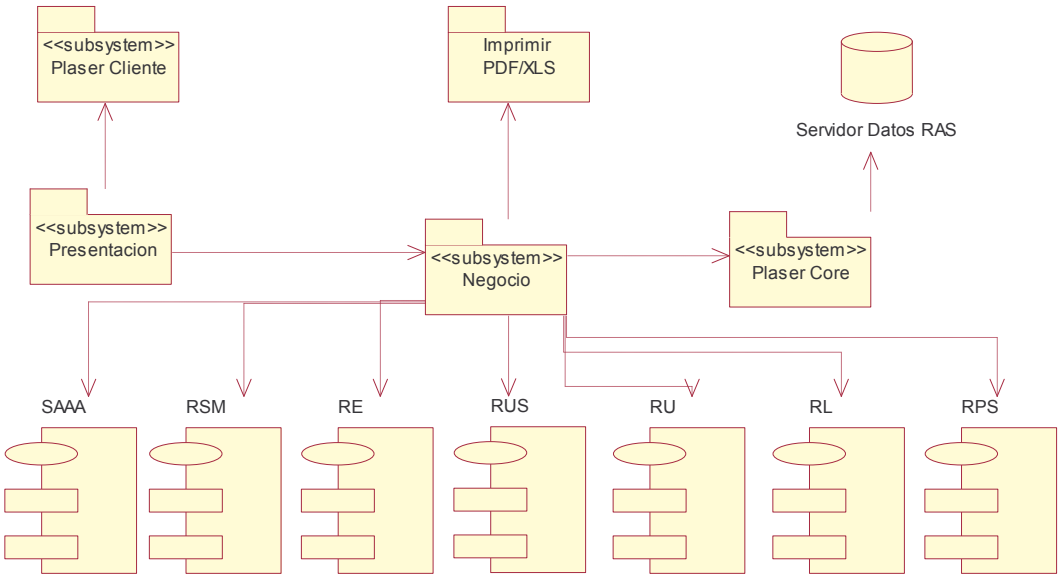


Figura 3.1 Diagrama General de los subsistemas de implementación del RAS.

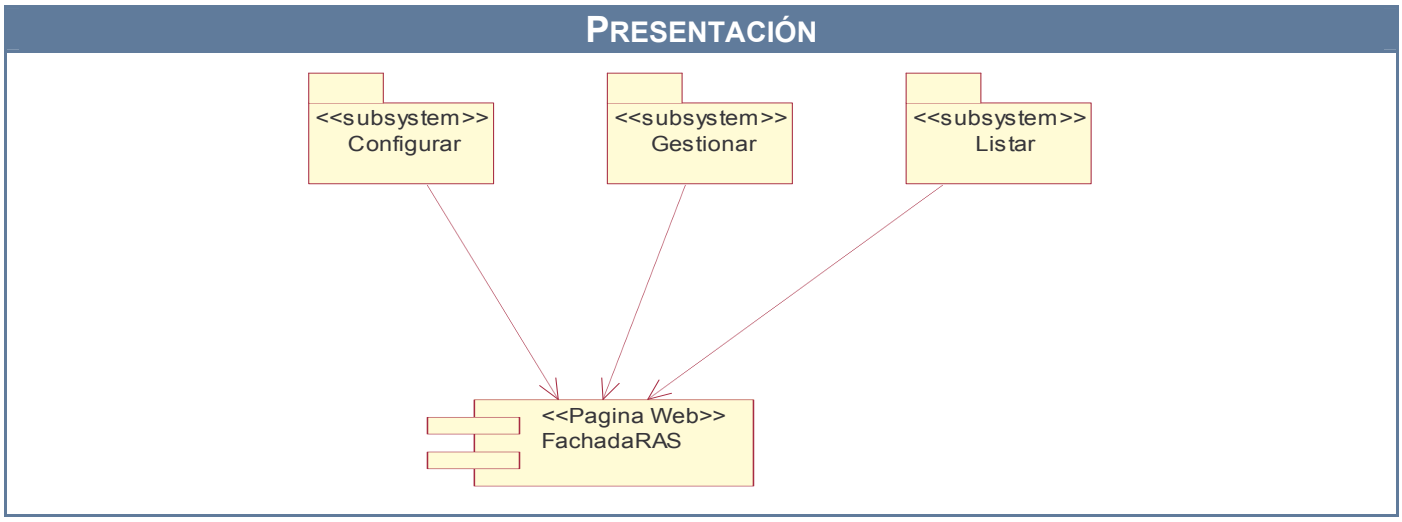


Figura 3.2 Diagrama de componentes del subsistema Presentación.

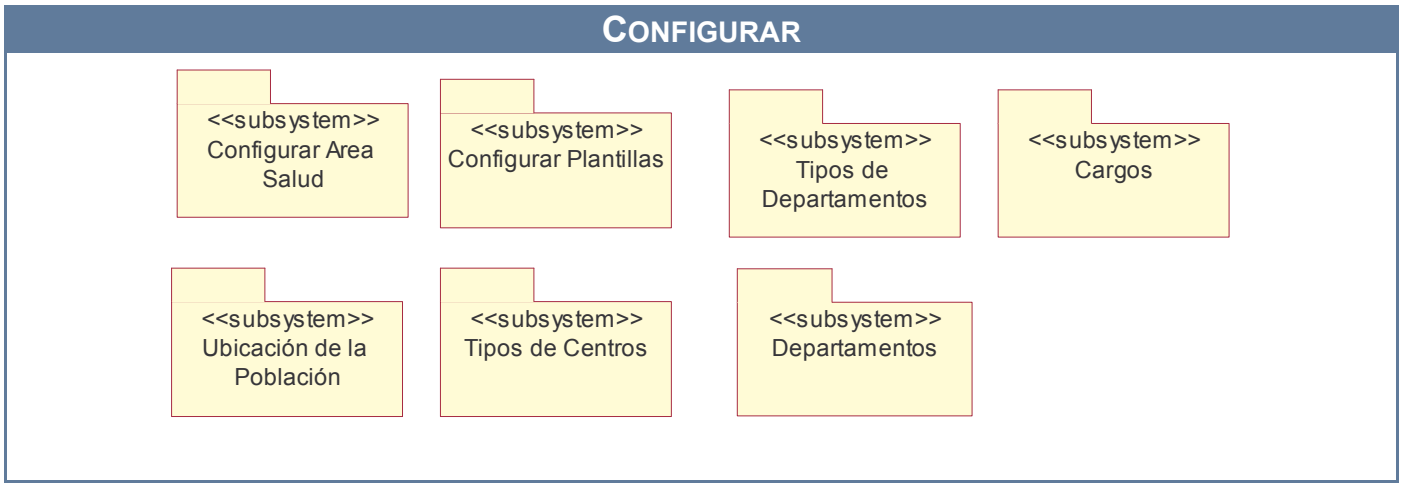


Figura 3.3 Diagrama de componentes del subsistema Configurar.

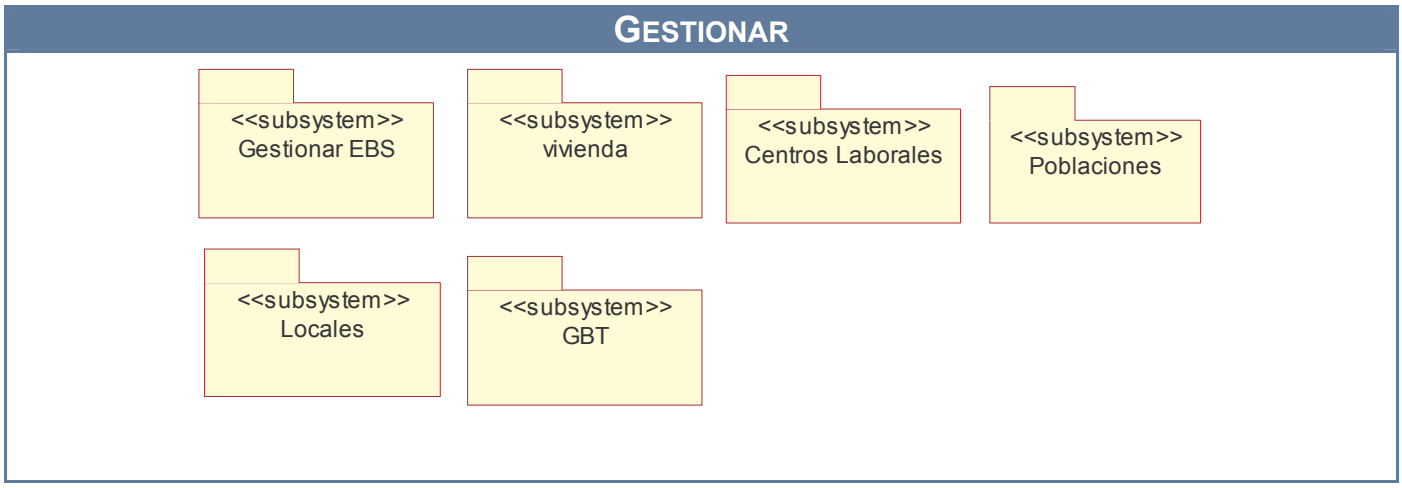


Figura 3.4 Diagrama de componentes del subsistema Gestionar.

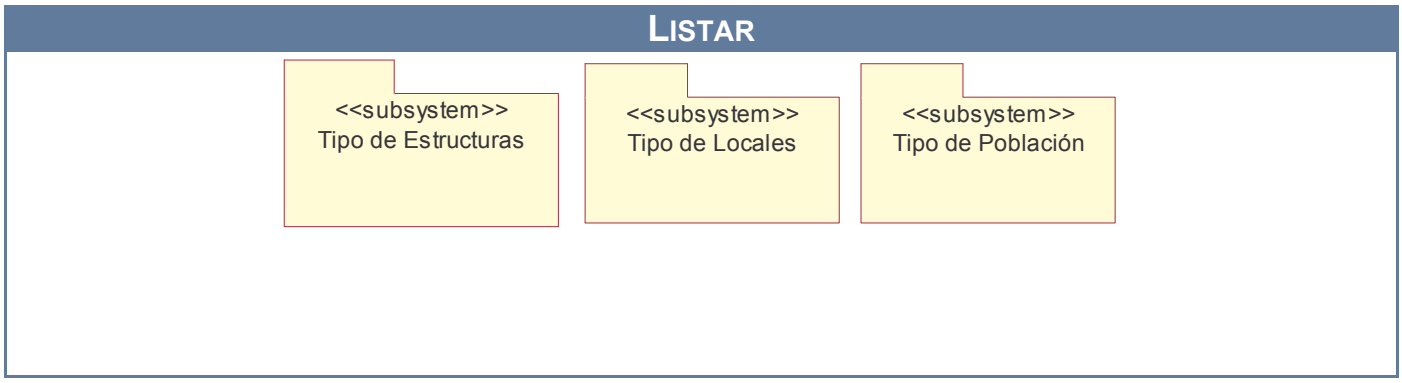


Figura 3.5 Diagrama de componentes del subsistema Gestionar.

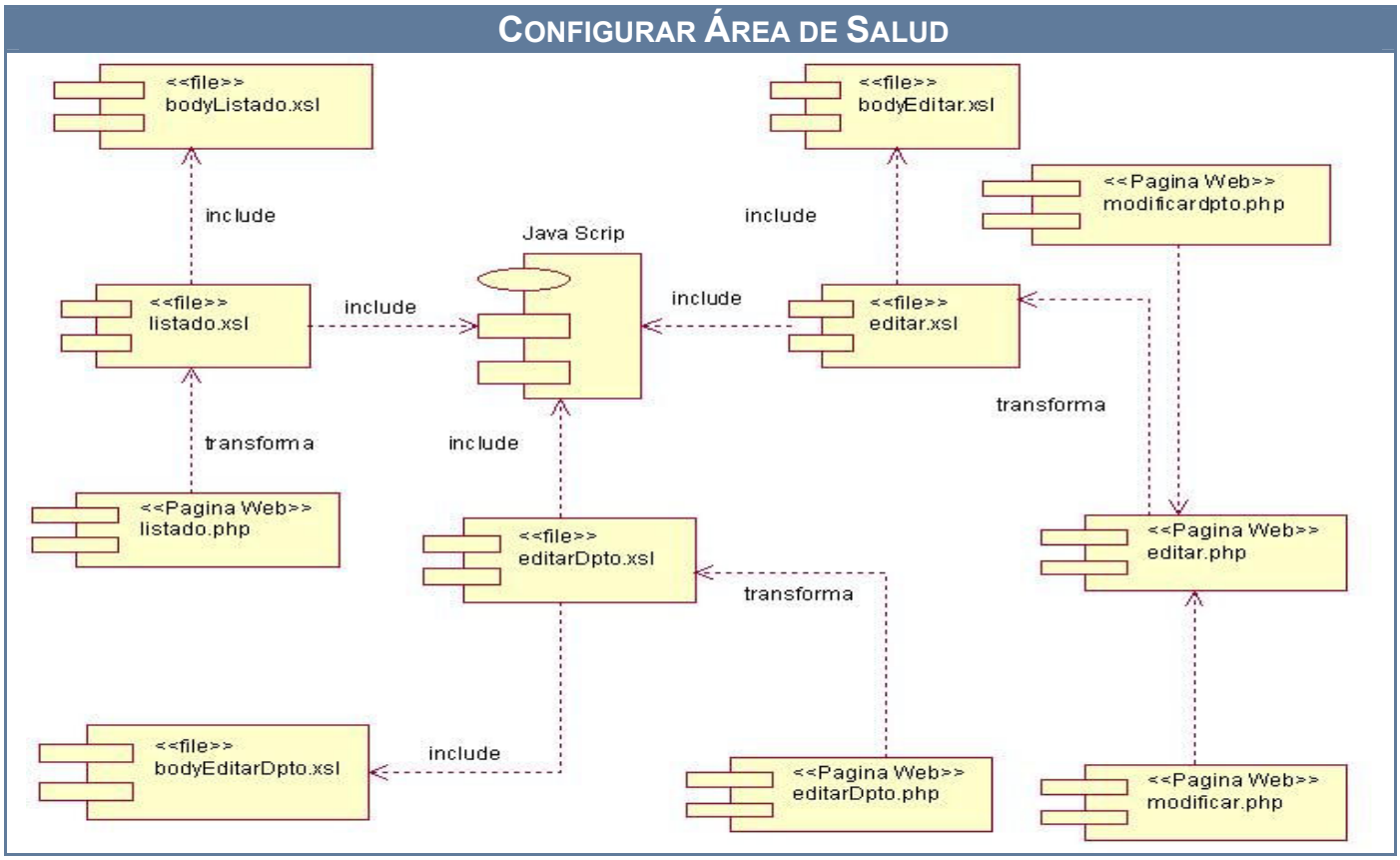


Figura 3.6 Diagrama de componentes de la capa de presentación del subsistema Configurar Área de Salud.

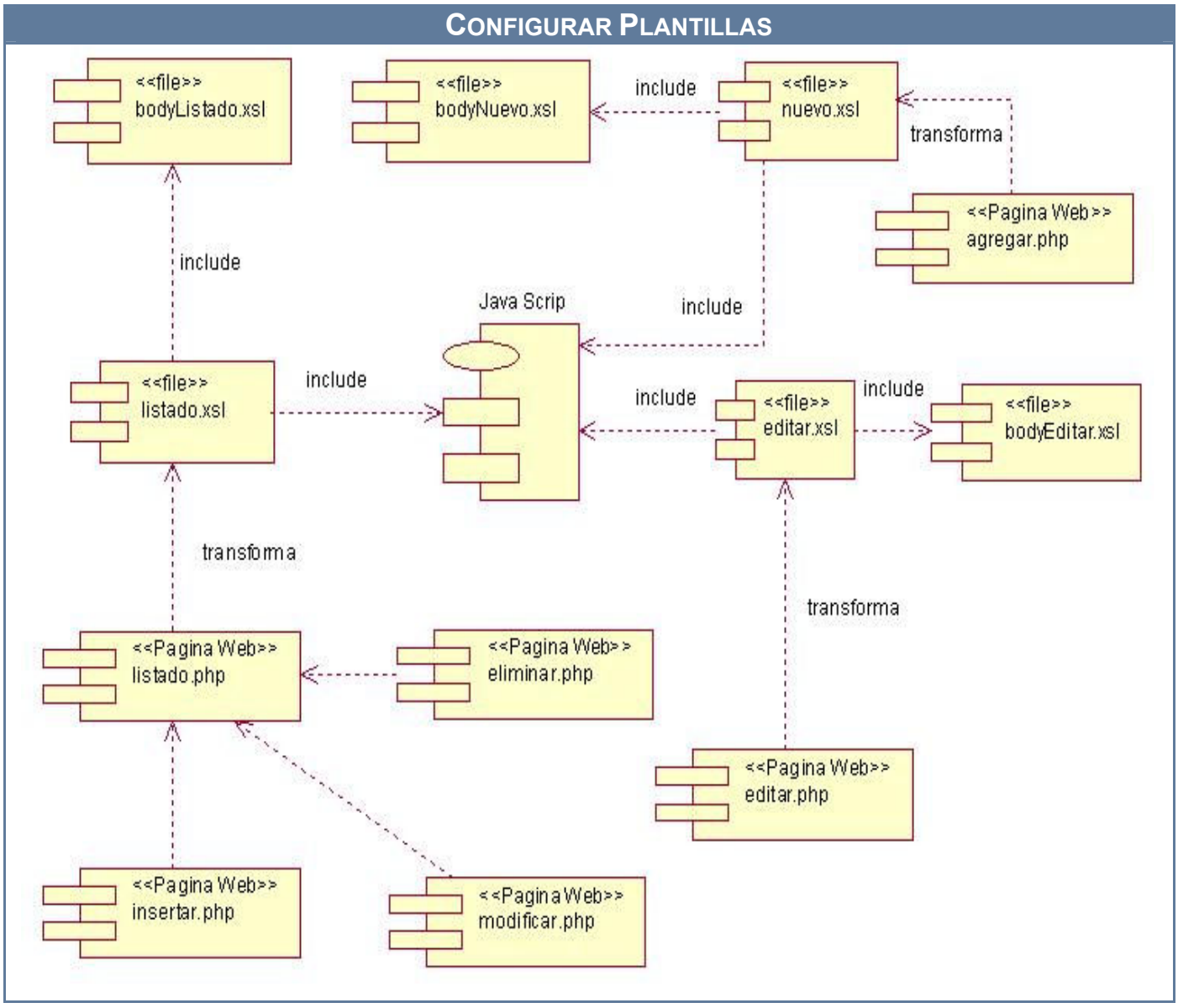


Figura 3.7 Diagramas de componentes de la capa de presentación del subsistema Configurar Plantillas.

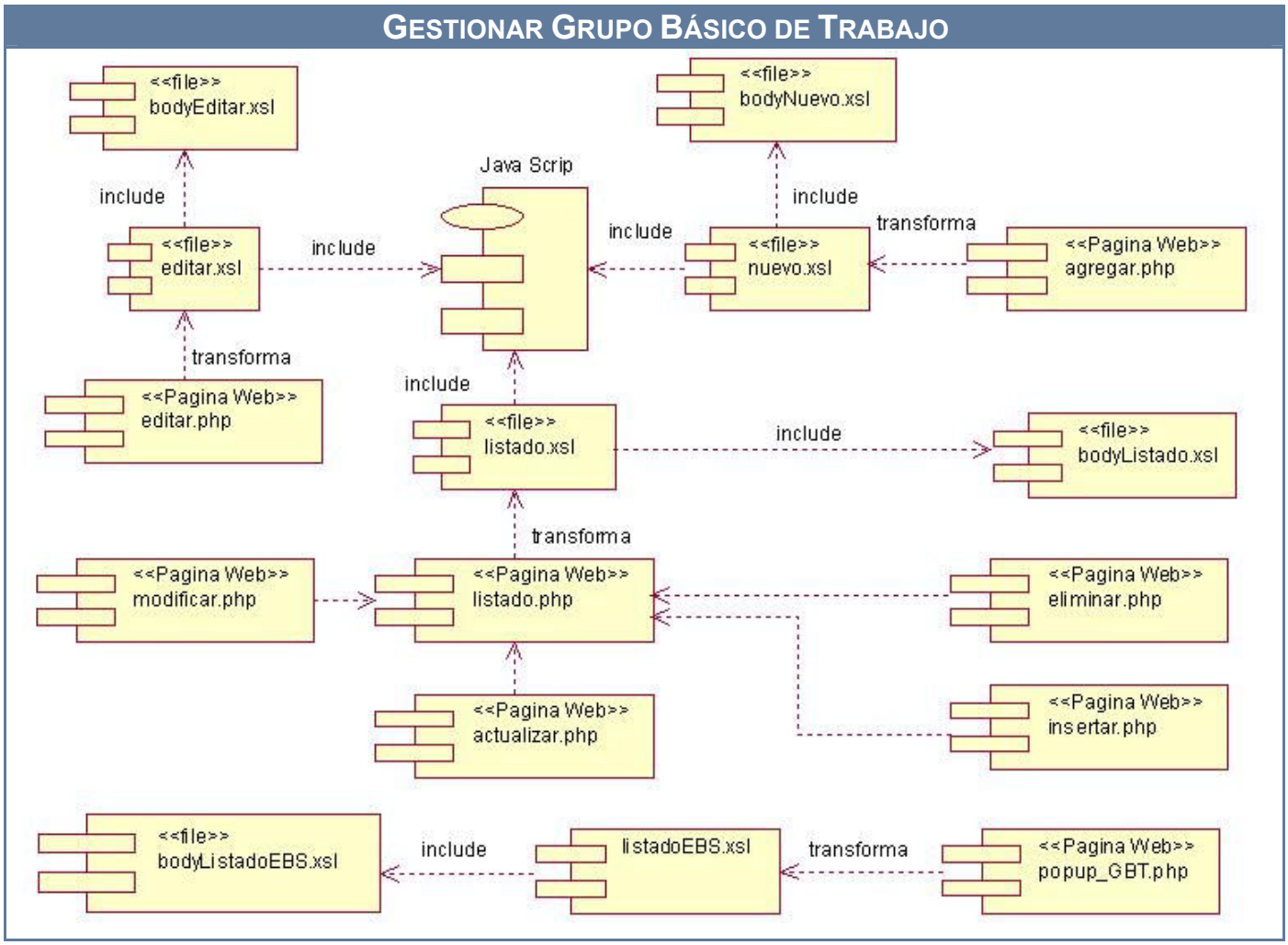


Figura 3.8 Diagramas de componentes de la capa de presentación del subsistema Gestionar Grupo Básico de trabajo.

3.3 Diagrama de Despliegue

Un diagrama de despliegue es un grafo de nodos unidos por conexiones de comunicación que muestra las relaciones físicas entre los componentes *hardware* que forman la topología sobre la que se ejecuta el sistema y la distribución de las partes de este en ellos.

La división entre cliente y servidor en un sistema es complicada ya que implica tomar algunas decisiones sobre dónde colocar físicamente sus componentes software, qué cantidad de software debe residir en el cliente, etc. Por lo que para modelarlo hay que identificar los nodos que representan los procesadores cliente y servidor del sistema de acuerdo a las capas que se van a implementar, destacar los dispositivos relacionados con el comportamiento del sistema, proporcionar señales visuales para esos procesadores y dispositivos a través de estereotipos y modelar la tipología de esos nodos mediante un diagrama de despliegue. [7]

Aunque UML no es un lenguaje de especificación de hardware de propósito general, se ha diseñado para modelar muchos de los aspectos hardware de un sistema a un nivel suficiente para que un ingeniero software pueda especificar la plataforma sobre la que se ejecuta el software del sistema y para que un ingeniero de sistemas pueda manejar la frontera entre el hardware y el software.
Para lograr una mejor escalabilidad, se disponen para el proyecto de tres servidores, uno para la capa de presentación, otro para la capa de reglas del negocio y otro para la base de datos.

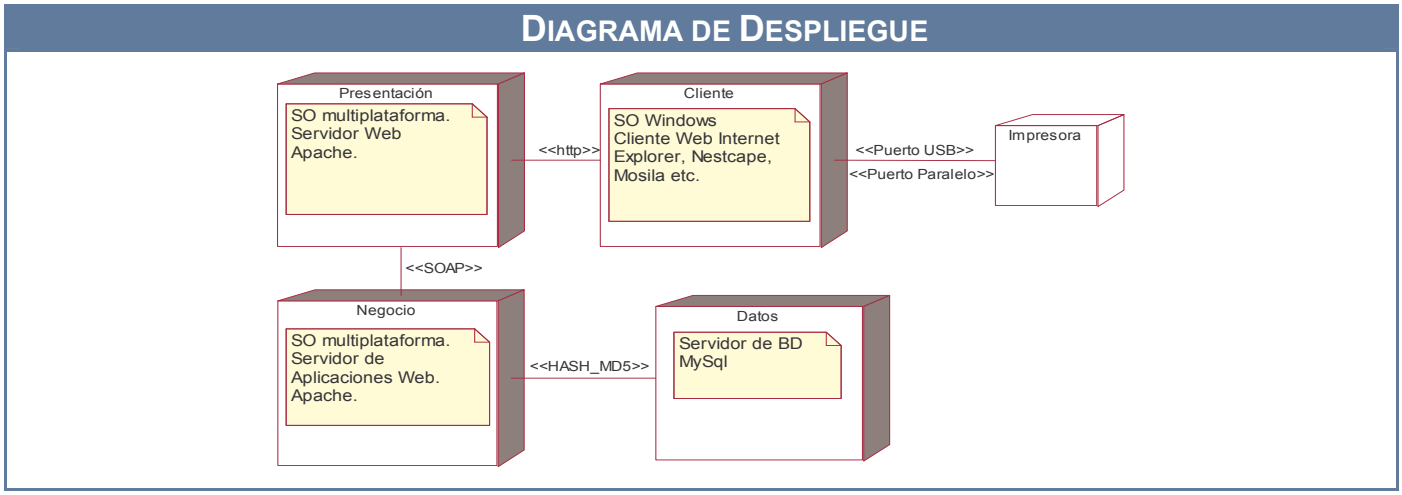


Figura 3.9 Diagrama de Despliegue.

[7] Piker, M, José. *DNS*. http://www.dcc.uchile.cl/~jpiquer/Internet/DNS/node1.html. (20/5/2005)

3.4 Descripción de los Métodos del Negocio más complejos.

A continuación se explican algunos de los métodos que se utilizan en los Casos de Uso, cuyos diagramas de clases se presentaron en el capítulo anterior.

AgenteBuscarGBT

Este método lista toda la información de los GBT, que se corresponden con los parámetros de entrada. (Ver tabla 2.17).
Lo primero que hace este agente es comunicarse con el componente Registro Personal de la Salud invocando al servicio Web *BuscarProfesionalesRapido,* para buscar los datos de los profesionales del Área de Salud. Seguido de esto, comprueba que exista alguna coincidencia entre los parámetros de búsqueda y los datos obtenidos de la búsqueda del Registro de Personal de la Salud. De ser así, realiza una búsqueda de los Grupos Básicos de Trabajo que contengan como integrantes a los profesionales que fueron encontrados en la selección anterior. En este proceso se tienen en cuenta los criterios de búsquedas propios de los Grupos Básicos de Trabajo. Este método debe devolver de forma paginada los datos de los GBT con un listado de los profesionales de la salud que lo integran.

DatosAreaSalud

El método se encarga de realizar una búsqueda de los datos de un área de salud. Los parámetros de entrada y salida del método están descritos en el Capítulo 2. (Ver tabla 2.24).
Lo primero que realiza el método es validar los parámetros de entrada.

Se hace una consulta a la base de datos del RAS para obtener los servicios del área. Se comunica con el componente Registro de Servicios Médicos invocando al servicio Web *Listar_Ser*, pasándole como parámetro un array con los identificadores de los servicios que tiene el área (el resultado de la consulta a la base de datos), para buscar sus datos. Hace una consulta a la base de datos del RAS para obtener los departamentos del área, a cada departamento obtenido le asigna sus servicios con la correspondiente descripción que devuelve el servicio Web *Listar_ser*.

Hace consultas a la base de datos del RAS para obtener los datos del área y de sus hospitales base. Se comunica con el componente Registro de Unidades de Salud invocando al servicio Web *Buscar_Total,* para obtener los datos del área como unidad de salud y de sus hospitales base.
Hace consulta a la base de datos del RAS para obtener las localidades del área y se comunica con el componente Registro de Ubicación invocando al servicio Web *ListarLocalidad* de donde se obtendrá la información de las localidades que se encuentran ubicadas en el Área de Salud.

InsertarAreaSalud

Este método es el encargado de insertar y actualizar la información de las Áreas de Salud, sus hospitales base, localidades y departamentos. Los parámetros de entrada y salida del método están descriptos en el capítulo 2 (ver Tabla 2.23).

Si los datos del Área de Salud no existen en la base de datos, al invocar a este método, los datos que llegan desde la capa de presentación son validados e insertados; si ya existen serán validados y actualizados en la base de datos. Este servicio tiene en cuenta que el usuario puede o no introducir todos los parámetros de entrada del método, de esta forma ejecuta la acción dependiendo de lo que se pase en los parámetros. Si el usuario desea eliminar una localidad o un departamento del Área, cuya información no pueda ser eliminada por estar asociada a otros datos, el método retorna una excepción indicando el error cometido. Para retornar un error indicando que una localidad no se puede eliminar, se comunica con el Registro de Ubicación invocando al servicio Web *listarLocalidad* para obtener la descripción de la localidad, ya que en la base de datos del Registro de Áreas de Salud sólo se tienen los identificadores de las localidades del Área.

AgenteListarLocalidadesAreaSalud
Es el encargado de buscar toda la información referente a las Localidades, Manzanas, Calles, Consejos Populares, Circunscripciones, Zonas CDR y CDR que se encuentran ubicados en el Área de Salud.

Para mostrar esta información, dado el identificador de un Área de Salud que se pasa como parámetro, lo primero que hace el método es obtener de la base de datos de áreas de salud, los identificadores de las localidades del Área de Salud. Con el resultado de esta consulta, se comunica con el componente

Registro de Ubicación invocando al servicio *ListarLocalidad* para obtener los datos de las Localidades, Manzanas y Calles. Después se comunica con el componente Registro Localidad invocando al servicio *ListarGerarquia, para obtener* los datos de los Consejos Populares, Circunscripciones, Zonas CDR y CDR, correspondientes a los identificadores de las localidades resultado de la consulta a la base de datos

Este agente es de gran importancia en la capa de presentación porque la información que retorna es utilizada en múltiples ocasiones y por muchas opciones del sistema, ahorrando la comunicación concurrente del módulo con los Registros de Ubicación y de Localidades respectivamente.

3.5 Estándares de Diseño, Codificación y Tratamiento de Errores.

3.5.1 Estándares de Codificación

Con el propósito de distribuir los esfuerzos y mejorar los rendimientos de la aplicación, se utilizó la programación orientada a servicios, creando clases genéricas que permiten la definición y distribución de las llamadas a los procedimientos de los módulos distribuidos, la definición de los métodos propios de cada módulo y el manejo de las bases de datos.

Actualmente se encuentran estándares de codificación para la mayoría de los lenguajes existentes. El uso de los mismos, partiendo de las convenciones definidas, permite una mejor comunicación entre los programadores, creando las condiciones para la reusabilidad y el mantenimiento de los sistemas. Para definir el estilo de codificación a seguir en la aplicación se utilizó la notación estándar establecida para aplicaciones desarrolladas en PHP (PHP CODIG Standard), que mayormente está basada en el estándar de código para aplicaciones en C++ (C++ CODIG Standard).

Las etiquetas de apertura y cierre del lenguaje serán de la forma <?php ?>, ya que siempre están disponibles en cualquier configuración.
Se hará uso de los arreglos predefinidos para el manejo de los valores enviados por el usuario $_GET, $_POST, $_FILES evitando el uso de $_REQUEST.

Para nombrar las variables se seguirá la regla de escribir los identificadores con letras minúsculas y en lenguaje español, utilizando como separador para las palabras el carácter "_", tratando de usar nombres sugerentes a la acción de la variable.
Todos los campos identificadores van a comenzar con el identificador (id) seguido del nombre del campo. Ejemplo: id _calle.
Los arreglos empezarán con el identificador array y las palabras no se separarán con el carácter "_". Ejemplo: Arrayidpoblacion.

Las estructuras se identificarán poniendo al final del nombre, struct. Ejemplo: paginadostruct.
Para identificar las clases se pondrá delante la letra C. Ejemplo: CFachadaRAS. En los métodos no se usarán abreviaturas y las palabras continuas deben comenzar con mayúsculas. Ejemplo: LiscarCargos.
Para comentar el código se utilizará, en el caso de una línea, al final de la misma el carácter "//" y seguido el comentario y en el caso de un bloque se utilizará los caracteres "/* */".

Se usará una indentación en el código de cuatro espacios para facilitar la lectura de éste. Las llaves se usarán poniendo la llave inicial en una línea para ella sóla, y en su respectiva columna la llave final también en una línea.

El idioma de las clases auxiliares como sesión y error, será el inglés para garantizar la homogeneidad con las programadas en este ámbito en el mundo, en el caso de los Servicios Web y la interfase de administración se usará el español para esclarecer los objetivos de cada método o script a utilizar.

Para lograr que las comparaciones sean seguras, se colocarán siempre los valores constantes a la izquierda de la comparación "if (6 == $variable)", con esto garantizará la generación de un error cuando por error escriba '=' y no '=='. Se utilizará el operador "?" para sentencias cortas, preferiblemente que ocupen una sola línea. La sentencia switch siempre tendrá la opción default y se evitará el uso de continue y break, ya que podrían perder la vista lógica del código fuente.

El almacenamiento de la información será en scripts SQL para construir la base de datos e interactuar con ella desde las aplicaciones.

Las palabras correspondientes a las sentencias SQL y sus parámetros deben ir en mayúsculas Ej: SELECT * FROM tb_cargos

Los nombres de las tablas deben ir en minúsculas y cada palabra separada por línea abajo "_". (Ejemplo: tb_area_salud)

En el caso de los XSL será con el mismo nombre que el fichero de la capa de presentación. (Ejemplo: editar.xsl)

Los controles seguirán el siguiente tratamiento:

Control	Prefijo	Ejemplo
Botón	Btn	btnAceptar
Etiqueta	Lbl	lblNombre
Lista/Menú	Mn	mnPrincipal
Campo de Texto	Txt	txtFecha
Botón de Opción	Opt	optSexo
Casilla de Verificación	Chx	chxBorrar
Grid o rejilla	Grid	grUsuario

Las páginas HTML se harán sin incluir código y todas las funciones JavaScript que se usarán se escribirán dentro de ficheros ".js".

Para la capa de datos tienen que nombrar la base de datos poniendo el identificador del proyecto "APS" seguido del carácter "_" y del nombre del módulo. Ejemplo: bd_area_salud

Las tablas se identificarán con el acrónimo tb_Nombre, ejemplo: tb_plantillas.

Los campos de la base de datos se nombrarán igual que las variables. (Ejemplo: id_unidad)

3.5.2 Tratamiento de Excepciones

Se utilizará JavaScript para depurar los errores. Por medio de este lenguaje serán informados la mayoría de los errores de la página, como apoyo a las validaciones de entrada de datos, garantizando que los datos introducidos por los usuarios sean válidos, o les sea posible corregirlos en caso contrario.

Otros errores en la capa de negocio serán tratados devolviendo un Fault cuyos elementos FaultCode, FaultString, FaultAutor describiremos a continuación:

FaultCode:

Código de texto utilizado para indicar la clase de error, será codificado de la siguiente manera.

Código del proyecto-código del módulo (:) número del método (.) número del error. Ejemplo: APS-RF: 1.5 que indica error 5 en el método 1 del módulo Registro de Fallecidos perteneciente al Proyecto APS.

FaultString:

Una explicación del error asequible al humano (explicativo). Debe tenerse en cuenta que este texto puede ser mostrado al operador final del sistema. Ejemplo: Formato de entrada no válido para la fecha de cierre estadístico.

FaultActor:

Un texto que indica quien provocó el error, siempre será el nombre del método que eleva la excepción. Ejemplo: RAS.InsertarCargo.

Detail:

Este elemento se usa para llevar mensajes de error específicos de aplicaciones, se empleará únicamente en errores cuya resolución depende del Centro de Control, en cualquier otro caso este elemento debe estar vacío.

También se utilizaron codificadores para evitar posibles errores por parte del usuario al registrar información de poca variabilidad, se pueden citar los casos de los grupos de edades, escolaridad, tipo de embarazo, certificación, sitio de la defunción, entre otros datos registrados en los certificados médicos de defunción.

3.5.3 Tratamiento de excepciones en la Capa de Presentación.

Antes de eliminar o modificar cualquier información se deben hacer las siguientes preguntas:

- ✓ ¿Está seguro que desea Eliminar los elementos seleccionados?

En el caso de agregar algo nuevo y falten campos obligatorios por llenar, se hace de forma particular para cada caso pero siempre siguiendo un mismo estándar:

- ✓ Por favor, se requiere “nombre del campo”.

- ✓ Por favor, especifique “nombre del campo”.
  Ejemplo
  Por favor, teclee el Nombre de la Plantilla.
  Por favor, teclee el Nombre del Cargo.

Al no existir resultados al realizar una Búsqueda dichos mensajes no se emiten en nuevas ventanas, sino en el mismo lugar donde se mostrarían los resultados en caso de existir.
No existen “Elementos” registrados hasta el momento para estos criterios de búsqueda.
Dichos elementos se tratan de forma personalizada.

Ejemplo:

- ✓ No existen “Plantillas” registradas hasta el momento para estos criterios de búsqueda.
- ✓ No existen “Cargos” registrados hasta el momento para estos criterios de búsqueda.
- ✓ No existen “GBT” registrados hasta el momento para estos criterios de búsqueda.

Los mensajes de errores que llegan a la capa de presentación desde la capa de negocio se muestran en la página de error del módulo, estos mensajes quedan bien definidos para el programador pero al usuario no le aporta nada, porque este no entiende de nombre de método ni de componte y menos del número del error, es por eso que se hace necesario llevar a cabo un proceso de actualización con respecto a cómo mostrar lo mensajes de errores en la capa de presentación, de forma tal que el error se muestre en la página donde esté trabajando el usuario y no en otra página lo cual obliga al usuario a presionar un botón más.

3.5.4 Estándares en la Interfaz de la aplicación

Los módulos que se desarrollan como parte del Sistema de Información para la Salud mantienen una profunda interrelación, permitiendo un acoplamiento entre ellos para ofrecer las respuestas que demanda el negocio propuesto por el Sistema Nacional de Salud. Este hecho tiene una repercusión determinante en la definición de la interfaz gráfica que se propone.

3.5.5 Diseño de Interfaz Gráfica del Proyecto APS

Para lograr una mayor eficiencia en el proceso de trabajo, y sobre todo para lograr una coherencia formal entre todos los módulos del sistema, y que sean identificados así como parte de un todo, se han pautado una serie de elementos comunes que facilitarán su reconocimiento y el uso que se haga de ellos.

Se diseñará una Pantalla Inicial global del Sistema Integral de Salud, desde la cual se accederá a los diferentes módulos del RIS, del SIAP, del SIGH y del SIAE. Esta pantalla contará con accesos a los diferentes módulos, informaciones generales, guías de ayuda, sistema de avisos que genera cada registro y enlaces definidos.

Así mismo será diseñada una Pantalla Inicial para cada una de las aplicaciones, que contará con accesos a todas las utilidades, avisos, ayuda y un enlace para regresar a la Pantalla Inicial del Sistema de Información para la Salud.

La estructura base de las aplicaciones es la misma para todos los módulos: las pantallas más usadas, los modelos establecidos, las rutas de navegación, las utilidades básicas, la organización de los elementos en pantalla y el diseño de identificadores serán comunes para todos.

Para particularizar el diseño de cada módulo se ha definido una pauta de dos colores básicos en este caso, gris y amarillo, con sus degradaciones hacia blanco y negro, así como la diferenciación por logotipo e imagen principal del cabezal, que identificará a cada módulo.

Su diseño está determinado fundamentalmente por el principio de la usabilidad, teniendo en cuenta que no se trata de un sitio Web, sino de una aplicación de trabajo donde el diseño tiene como principal propósito facilitar su uso, comprensión y navegación, por encima de ornamentos inútiles, aunque manteniendo pautas estéticas, orgánicas y agradables.

Formalmente, usabilidad se define como la medida en que un producto puede ser usado por determinados usuarios para conseguir objetivos específicos con efectividad, eficiencia y satisfacción, en un contexto de uso especificado [ISO 9241-11].

La resolución óptima para la cual están diseñadas las aplicaciones es de 800 x 600 px. El fondo siempre será blanco y los elementos de pantalla de los colores definidos para cada módulo.
Se ha definido un cabezal pequeño de 65 px de altura, más pequeño que el utilizado en las páginas Web, que recomiendan cabezales de hasta 80 px de altura.

El menú principal siempre estará situado en una barra superior horizontal de solo 15 px de altura. No existirá barra vertical de menú situada la izquierda de la página (como usualmente se hace) para ampliar el espacio de trabajo, pues estará reservado lo más amplio posible para la inserción de grandes tablas y formularios que constituyen la base fundamental de estas aplicaciones.

El logo siempre estará ubicado en el extremo superior izquierdo de la página, es una imagen que cuenta con un ancho de 270 px y se corresponde con el nombre de cada módulo. Estará constituido por un juego tipográfico en Frankling Gothic Medium, y en el caso de las aplicaciones propias del Proyecto APS, estando especificado dentro del logo como una especie de genérico.

Bajo el logo existirá una barra de ubicación dentro del sitio, funcionando como hipervínculo, que servirá como referencia para saber donde se encuentra el usuario o para acceder rápidamente a cualquiera de los niveles superiores de navegación dentro de los que se encuentra. Además se encontrará destacado dentro del menú principal (con un destaque en el color secundario) en cuál de los elementos del menú se encuentra el usuario en ese momento.

La tipografía será siempre Tahoma, por su amplia legibilidad y por las facilidades conocidas que brinda para la lectura digital. El menú principal será a 7 ptos y los submenús a 6 ptos. Los demás puntajes se definirían en dependencia de las necesidades puntuales de cada pantalla.

El espacio de trabajo comienza 33 px por debajo del menú. El espacio intermedio que queda es también con fondo blanco y está reservado para el texto de ubicación dentro del sitio (justificado a la izquierda) y para ubicar los botones propios de la pantalla (justificados a la derecha). Estos se organizarán en una o dos filas, de hasta cuatro botones (13 x 72 px) cada una. Los botones se corresponden también con los colores pautados.

Entre los elementos comunes del menú principal se encuentran Inicio para regresar a la página inicial del módulo, Salir para desconectarse del sistema, y Otros Módulos para facilitar los enlaces a otros módulos necesarios. Son también comunes a casi todos los botones del menú principal Configurar para la configuración de codificadores, Cierre para la realización de cierre estadístico y Reportes para generar reportes de actividades u operaciones.
Es común para todos los módulos el diseño de una serie de ventanas, en las que sólo cambiarían los colores, en dependencia de cada uno. Son estas las ventanas de precaución, error, validación de datos, etc.

En cuanto a los elementos de diseño del interior de las pantallas, es decir, de las tablas, formularios, etc., se definen los edit que se utilicen con una altura de 16 px y la separación entre estos y entre ellos y los bordes de tablas será de 8 px. Será de 8 px la separación entre el texto y el edit. Los textos de estos campos serán justificados siempre a la derecha, es decir, justificados a 8 ptos de cada edit.

En el caso de tablas generadas por búsquedas, que ordenan una serie de elementos, y necesiten selección, se harán a través de checkboxs justificados a la izquierda de la tabla. Siempre habrá un checkbox en la fila de título, también a la izquierda, que facilite seleccionar todos. Es necesario destacar que estas tablas pueden tener una cantidad grande de líneas generadas por la búsqueda, por lo que debe quedar pautado que hasta 25 resultados la tabla funcione con scroll, pero más de esta cantidad será entonces por paginado, al estilo de Google, con 25 resultados por página.

Existen detalles que serán definidos particularmente en cada uno de los módulos, ya que satisfacen a necesidades específicas de los mismos. El interés general es mantener el diseño y la estructura del sitio lo más simple posible, la simplicidad es entendimiento del contenido, de la estructura, es facilidad para encontrar lo que se busca, es también velocidad de descarga.

Conclusiones

En este capítulo se hizo una explicación de la integración del Registro de Áreas de Salud con otros componentes de SISalud. Se presentaron los diagramas de componentes de los casos de usos realizados en el flujo de trabajo “diseño” y el diagrama de despliegue, que describe la estructura de distribución de la

aplicación. La descripción de los métodos más complejos de la capa de negocio de algunos casos de uso, refleja el nivel de integración del RAS al resto de los componentes del Sistema de Información para la Salud (SISalud). Además se refleja el uso de los estándares de diseño, codificación y tratamiento de excepciones.

Conclusiones

Al concluir el trabajo se ha dado cumplimiento al objetivo propuesto, pues se dispone de un registro que gestiona la información de las Áreas de Salud, el cual puede ser consultado en los diferentes niveles de dirección del SNS y brinda información a las demás aplicaciones de SISalud. Se dió cumplimiento a las tareas propuestas:

- ✓ Se modelaron, siguiendo RUP, los flujos de trabajo Diseño e Implementación, teniendo en cuenta la arquitectura definida por el MINSAP, la integración con otros componentes y las tecnologías actuales para el desarrollo.
- ✓ Se implementó el registro RAS, utilizando los estándares de diseño, codificación y excepción de errores.

Recomendaciones

Tomando como base la investigación realizada y la experiencia acumulada durante la realización de este trabajo, se proponen las siguientes recomendaciones:

- ✓ Diseñar e Implementar los reportes estadísticos, que permitan obtener la información que se gestiona en las Áreas de Salud en los diferentes niveles de dirección del Sistema Nacional de Salud.
- ✓ Definir un plan de capacitación que permita el adiestramiento del personal en las áreas de salud.
- ✓ Desplegar la aplicación en los policlínicos definidos para la realización de la prueba piloto.
- ✓ Definir un plan de acción, en la Dirección Nacional de APS, para la configuración de los codificadores que se utilizan en la gestión de la información de las Áreas de Salud.

Bibliografía

1-Delgado Ramos, Dr. Ariel, Cabrera Hernández, Ing. Mirna, Juncal, Dra. Virginia (2005). Registro Informatizado de Salud (RIS). Consultado en (Enero, 11, 2007) en http://www.sld.cu/galerias/pdf/sitios/dne/ris.pdf

2-espaweb.com [En línea] // espaweb.com. - 1997. - 3 de Febrero de 2007. - http://www.espaweb.com/respuestas_online/PHP.html.

3-es.tldp [En línea] / aut. Antonio Saorín José J. Grinaldo // es.tldp. - 12 de Marzo de 2003. - 6 de Febrero de 2007. - http://es.tldp.org/Tutoriales/doc-servir-web-escuela/doc-servir-web-escuela-html/apache.html.

4-es.wikipedia.org [En línea] // es.wikipedia.org. - 16 de Enero de 2007. - 10 de Febrero de 2007. - http://es.wikipedia.org/wiki/Servicio_Web.

5-Flujo de Trabajo Análisis y Diseño. UCI. Ciudad Habana, Cuba: s.n., 2006.

6-fpress.com [En línea] / aut. Rodríguez Alberto // fpress.com. - FoxPress, 12 de Septiembre de 1997. - 9 de Febrero de 2007. - http://www.fpress.com/revista/Num9711/Nov97.htm.

7-infor.uva.es [En línea] // infor.uva.es. - 3 de Marzo de 2002. - 6 de Febrero de 2007. - http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node17.html .

8-infor.uva.es [En línea] / aut. Vega Jesús // infor.uva.es. - 21 de Marzo de 2002. - 5 de febrero de 2007. - http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node19.html.

9-infor.uva.es [En línea] / aut. Vega Jesús // infor.uva.es. - 3 de Marzo de 2002. - 6 de Febrero de 2007. - http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node20.html.

10-infor.uva.es [En línea] / aut. Vega Jesús // infor.uva.es. - 3 de marzo de 2002. - 10 de Febrero de 2007. - http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node13.html.

11-Ing. Jose Angel Franco Navarro. (2004). UML en acción. Modelando Aplicaciones Web.. Ciudad de la Habana. Cuba:

12-ldc.usb.ve [En línea] / aut. Teruel Prof. Alejandro // ldc.usb.ve. - 15 de Septiembre de 2000. - 6 de Febrero de 2007. - http://www.ldc.usb.ve/~teruel/ci3715/clases/arqCapas.html.

13-lenguajes-de-programacion.com [En línea] // lenguajes-de-programacion.com. - 2006. - 2 de Febrero de 2007. - http://lenguajes-de-programacion.com/programacion-web.shtml.

14-linalco.com [En línea] / aut. Libre Especilistas en Linux y Software // linalco.com. - 3 de Febrero de 2007. - http://www.linalco.com/apache.html.

15-linuxfocus.org [En línea] / aut. Willighagen Egon // linuxfocus.org. - 3 de Marzo de 2001. - 5 de Febrero de 2007. - http://www.linuxfocus.org/Castellano/July2000/article156.shtml.

16-Marín Díaz, Miguel E. Fundamentos del Sistema de Salud Pública en Cuba para estudiantes de Informática. Ciudad Habana, 2006.

17-monografia.com [En línea] / aut. Rodríguez MsC. Caridad Salazar Alea y MsC. Antonio Cortina // monografia.com. - 23 de enero de 2001. - 5 de Febrero de 2007. - http://www.monografias.com/trabajos14/aplicacion-distrib/aplicacion-distrib.shtml.

18-mysql-hispano.org [En línea] / aut. Eduardo // mysql-hispano.org . - 26 de Agosto de 2002. - 6 de Febrero de 2007. - http://www.mysql-hispano.org/page.php?id=2.

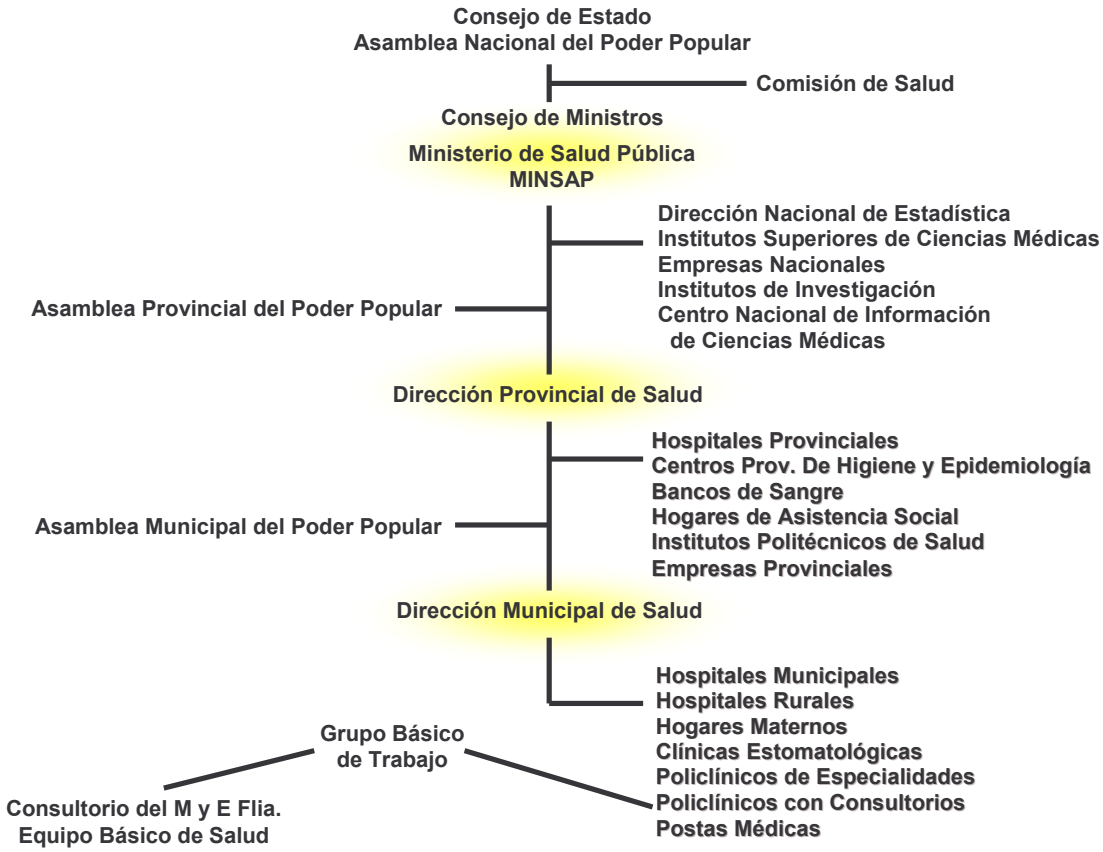
19-mx.php.net [En línea] / aut. Group PHP // mx.php.net. - 13 de Febrero de 2001. - 4 de Febrero de 2007. - http://mx.php.net/manual/es/intro-whatcando.php.

20-neo.lcc.uma.es [En línea] // neo.lcc.uma.es. - 8 de Febrero de 2007. - http://neo.lcc.uma.es/evirtual/cdd/tutorial/aplicacion/cliente-servidor.html.

21-platea.pntic.mec.es [En línea] // platea.pntic.mec.es. - 2000. - 4 de Febrero de 2007. - http://platea.pntic.mec.es/jmas/manual/html/java__javascript.html.

22-Piker, M, José. *DNS*. http://www.dcc.uchile.cl/~jpiquer/Internet/DNS/node1.html. (20/5/2005)
23-Soñora Cruz, William (2005). Registro de Actividades Diarias del Equipo Básico de Salud Para el Sistema Informatizado de Atención Primaria (Tesis de Pregrado, Instituto Superior Politécnico "José Antonio Echeverría", Universidad De Holguín "Oscar Lucero Moya" Universidad de las Ciencias Informáticas).
24-uji.es [En línea] / aut. Andrés Maria Mercedes Marqués // uji.es. - 12 de Febrero de 2001. - 6 de Febrero de 2007. - http://www3.uji.es/~mmarques/f47/apun/node39.html.
25-Varios Autores. Propuesta de Esquema Sistema Integral de Salud. Ciudad Habana,2006.
26-webigac1.igac.gov.co [En línea] // webigac1.igac.gov.co . - 2004. - 8 de febrero de 2007. - http://webigac1.igac.gov.co/temp/paginas/ctr_sistemasdegestiondebasededatos.htm.
27-wikipedia [En línea] // wikipedia. - 2 de febrero de 2007. - 10 de febrero de 2007. - http://es.wikipedia.org/wiki/Internet.
28-wikipedia.org [En línea] // wikipedia.org. - 7 de Enero de 2007. - 10 de Febrero de 2007 - http://es.wikipedia.org/wiki/Aplicaci%C3%B3n_web.

Anexos



Anexo 1 Estructura Organizativa del Sistema Nacional de Salud

Glosario de Términos

A continuación se presentan todos los términos médicos manejados a lo largo de todo el proyecto de desarrollo de un sistema para la gestión del Área de Salud.

Acciones de Salud: Actividades que se realizan a nivel de Consultorio tales como: Consultas médicas, Terrenos, vacunación y Citología Orgánica (pruebas citológicas).

Atención Primaria de Salud (APS): Nivel asistencial que constituye la puerta de entrada fundamental del paciente al Sistema Nacional de Salud, donde debe darse solución alrededor del 90% de los problemas que afectan a la población. En este nivel se realizan acciones de promoción, prevención, curación y de Rehabilitación.

Área de Salud: Área geográfica a la que presta sus servicios una unidad de salud que contemple el Programa de Trabajo del Médico y la Enfermera de la Familia. Tienen el nombre del Policlínico.

Citología Orgánica: Proceder realizado a las mujeres entre 25 y 60 años de edad, consistente en la toma de una muestra por raspado del cuello uterino y su estudio en el laboratorio. Permite detectar afecciones tales como neoplasias de cuello, infecciones, etc.

Consulta Médica: Relación interpersonal médico – paciente que puede ocurrir en el Consultorio o en el hogar del paciente, en la que el médico realiza acciones de prevención, diagnóstico, tratamiento y Rehabilitación para la solución de los problemas identificados.

Consultorio: Local perteneciente al Área de Salud donde se realizan las actividades asistenciales (consultas, curaciones, vacunación, exámenes, etc.) del Equipo Básico de Salud de la Atención Primaria, que puede estar ubicado en la comunidad, centros educacionales o de trabajo, etc.

Equipo Básico de Salud (EBS): Binomio de trabajo conformado por el médico y la enfermera de la familia (en las circunstancias actuales se define como un médico jefe del equipo, una o más enfermeras y un estudiante de medicina), que atiende una población geográficamente determinada, que puede estar ubicado en la comunidad, centros laborales o educacionales.

Consejos Populares: Estructura administrativa de Gobierno que establece la subordinación de los CDR a las circunscripciones y estas a los consejos populares (Provincia – Municipio – Distrito – Consejo Popular – Circunscripción - CDR).

Grupo Básico de Trabajo (GBT): Equipo de trabajo multidisciplinario integrado por un conjunto de especialistas de diferentes especialidades médicas (Medicina Interna, Gineco-obstetricia, Medicina General Integral (MGI), Licenciado en sicología, Profesores, Enfermera Supervisora, Técnico de higiene y

epidemiología, Técnico en trabajo social, etc.) todos en función de atender a un determinado grupo, generalmente entre 15 y 20, de Equipo Básico de Salud. El grupo cumple funciones asistenciales, docentes y gerenciales dirigidas a incrementar la calidad de la atención a la salud de la población. Se puede dar el caso de existir estos de forma incompleta y/o existir como Grupo Básico de Trabajo de nuevo tipo donde cuente con el Especialista de MGI como único profesor del mismo.

Hospital Base: Unidad de Salud (Generalmente Hospitales) que se responsabilizan de manera integra con las unidades de salud de la Atención Primaria (Áreas de Salud) para el enfrentamiento de los problemas de salud de una determinada población.

Local: Este término ha sido usado en este documento con similar significado a Consultorio. Un local es una entidad que puede ser para Consultorio (local de trabajo de los Equipos Básicos de Salud) o para ser vivienda de los especialistas que integran el Equipo Básico de Salud, es por ello que los locales de Consulta pueden o no tener asociados hasta dos locales de viviendas.

Plan Turquino Manatí: Plan socioeconómico estatal para el desarrollo integral y sostenible de las zonas montañosas, conjugando armónicamente los requerimientos productivos con el desarrollo social, la conservación de la naturaleza y el fortalecimiento de la defensa del país.

Profundidad de la consulta: Es un número de 1 a 3 que indica el nivel (Área de Salud, Grupo Básico de Trabajo, Equipo Básico de Salud) que establece la estructura (total, desglose) de la información solicitada por el usuario.

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.

Unidad de Salud: Centro de trabajo del Ministerio de Salud Pública.

Términos Informáticos.

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

Cohesión: Es una 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 los otros componentes.

Usuario autenticado: Es aquel usuario que ha proporcionado información mediante la cual el mecanismo de seguridad garantiza su identificación al intentar acceder a los componentes del sistema, los mecanismos de autenticación pueden ser tan simples como una contraseña o tan complejos como un dispositivo analizador de patrones retínales.