UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 7

Trabajo de Diploma para Optar por el Título de
Ingeniero en Ciencias Informáticas

Módulo Configuración para Escritorio del Sistema Integral para la Atención Primaria de Salud (SIAPS)

Autores:

Indira Massip Mursulí

Fidel González Barreda

Tutores:

Ing. Alexander Fonseca Cardosa

Ing. Yandy Rojas Barrios

Ciudad de La Habana, julio de 2010

"Año 52 de la Revolución"

Datos de Contacto

Ing. Alexander Fonseca Cardosa (Tutor). Instructor recién graduado en Ingeniero en Ciencias Informáticas en el año 2008, en la Universidad de las Ciencias Informáticas. Profesor Facultad # 7. Ha impartido las asignaturas Matemática 3 y 4. Forma parte del proyecto APS. Ha presentado ponencias en UCIENCIA 2008 e Informática 2009. Su dirección de correo electrónico: afonseca@uci.cu.

Ing. Yandy Rojas Barrios (Tutor). Instructor recién graduado en Ingeniero en Ciencias Informáticas en el año 2008, en la Universidad de las Ciencias Informáticas. Profesor Facultad # 7. Ha impartido las asignaturas Física 1 y 2. Forma parte del proyecto APS. Ha presentado ponencias en UCIENCIA 2008 e Informática 2009. Su dirección de correo es yrbarrios@uci.cu.

RESUMEN

Como resultado del auge de la informática y las comunicaciones a nivel mundial, la máxima dirección del Estado cubano, tomó como medida fundamental, llevar a cabo la informatización del Sistema Nacional de Salud (SNS) para una mayor organización y calidad de los servicios.

Precisamente, este trabajo propone como objetivo desarrollar un módulo que configure los parámetros necesarios para el funcionamiento del Sistema Integral para la Atención Primaria de Salud en las unidades de salud con limitaciones tecnológicas. Este módulo ofrece una nueva funcionalidad que permite que la información generada en el Módulo Configuración para Escritorio pueda ser vista en los diferentes niveles del Sistema Nacional de Salud (SNS). Además, se evita la pérdida de datos y les facilita a los usuarios el procesamiento rápido y seguro de toda la información.

Para dar cumplimiento al objetivo se utilizan tecnologías de avanzada a nivel mundial, tales como: Netbeans IDE 6.7.1, RazorSQL 4.5 y como Motor de Bases de Datos HSQLDB 1.8.0, más la herramienta Visual Paradigm 6.4. Para la documentación de dicho trabajo se usa la metodología de RUP, con el lenguaje de modelado UML. Estas se emplearon por todas las facilidades que brindan y porque responden a las políticas trazadas por el Ministerio de Salud Pública de software libre y código abierto.

Palabras clave:

Sistema Nacional de Salud, Sistema Integral para la Atención Primaria de Salud, Módulo Configuración para Escritorio.

TABLA DE CONTENIDOS

ÍNDICE

TABLA DE CONTENIDOS

TABLA DE CONTENIDOS

Introducción

Una de las grandes metas del Estado cubano es que las instituciones del país alcancen un elevado nivel de informatización de las actividades que brinda. El futuro de la sociedad dependerá, en gran medida, del potencial humano, de la gestión de la producción y de los conocimientos que se alcancen. En realidad, la informática en sus diferentes manifestaciones, tiene asegurado un papel protagónico en ese porvenir.

Coherente con este propósito, el Ministerio de Salud Pública (MINSAP) contempla entre sus prioridades la incorporación de los avances de las Nuevas Tecnologías de la Informática y las Comunicaciones (N'TICs) en los procesos relacionados con la salud cubana. Es por ello, que las instituciones de salud han desarrollado sistemas informáticos que permiten ofrecer al pueblo un servicio con la mayor calidad y eficiencia posible, en la misma medida que se coordinan los esfuerzos para perfeccionar el Sistema Nacional de Salud (SNS).

El SNS está estructurado en tres niveles de atención médica:

- Nivel de atención primaria: se brinda a nivel de los consultorios, policlínicos y hospitales rurales por medio del Programa de Medicina Familiar; es la primera interacción que va a tener el paciente con el médico y se estima un aproximado de atención primaria de un 80% de las necesidades médicas de la población, según la Organización Mundial de la Salud.
- Nivel de atención secundaria: son las instituciones subordinadas a la Dirección Provincial de Salud, ejemplo hospitales provinciales, o sea, atienden a toda la población de una provincia determinada y se estima un aproximado de atención secundaria de un 15% de las necesidades médicas de la población, según la Organización Mundial de la Salud.
- Nivel de atención terciaria: por su condición muy especializada, sólo se brinda en determinados centros, ejemplo: Instituto de Neurocirugía, Instituto de Cirugía Cardiovascular, Instituto de Nefrología, Instituto de Gastroenterología, entre otros y se estima un aproximado de atención terciaria de un 5% de las necesidades médicas de la población, según la Organización Mundial de la Salud.

El punto de partida de la atención médica es la Atención Primaria de Salud (APS), la puerta principal de entrada al sistema de salud, con énfasis en la prevención de enfermedades y promoción de la atención médica. La APS constituye, en los momentos actuales, un escenario de profundas transformaciones,

desde el llamado del Comandante en Jefe, en septiembre del 2002, donde planteó la necesidad de llegar al perfeccionamiento del sistema.

Hasta 1980 todos los sistemas informáticos estaban centralizados en centros de cálculo. A partir de entonces, se comienzan a simplificar, descentralizar y popularizar las aplicaciones informáticas y a mediados de los 1980, cuando llegaron las primeras Computadoras Personales a algunos policlínicos comunitarios docentes, posibilitó una de las primeras aplicaciones informáticas en la APS, donde se automatizaron los sistemas de despenalización y estadísticas. Luego, el interés por el control estadístico de la población, sus problemas de salud, así como los servicios de los consultorios y policlínicos para uso gerencial de salud, han aumentado el trabajo informático en la APS cubana.

El principal problema al que se enfrenta la APS en los últimos años es la demanda excesiva, la cual se manifiesta en dos indicadores:

- La presión o sobrecarga asistencial que no es más que el número total de pacientes atendidos por el profesional sanitario al día.
- La frecuentación que es el número de visitas por paciente al año.

La sobrecarga asistencial tiene como consecuencias principales una disminución del tiempo de consulta dedicado al paciente, muy por debajo actualmente de los 10 minutos que reclaman la mayoría de los profesionales, incapacidad de desarrollar programas de prevención y promoción de la salud y una mayor prescripción. Esto a su vez genera un aumento de la frecuentación por parte del usuario, con lo que se entra en una espiral de deterioro progresivo de la calidad de la asistencia que se presta en los Centros de Salud. Las técnicas de citación en visitas programadas parecen haber fracasado y las consultas pueden rondar entre el 20 y 40% diariamente.

La recopilación de información se realiza de forma manual y en papel, lo cual demanda una porción muy significativa de tiempo. Según estudios realizados, se demostró que por cada hora de cuidado directo sobre el paciente se generaba entre 30 y 60 minutos de registro en papel, aumentando la cantidad de documentación acumulada.

A partir de una estrategia trazada por el MINSAP y el Ministerio de la Informática y las Comunicaciones (MIC), con la colaboración de la Universidad de Ciencia Informáticas (UCI) y el Centro Especializado en

Soluciones para la Informática Médica (CESIM), se decide el desarrollo y completamiento de la informatización de la APS.

El Sistema Integral para la Atención Primaria de Salud (SIAPS) tiene como objetivo la solución a múltiples inconvenientes vinculados al sector de la salud y abarca todos los procesos principales del espectro de la atención primaria de salud en la comunidad, dando un paso de avance en la prevención, investigación, recuperación y predicción de los problemas de salud del paciente. Este incluye módulos como: Clínico Quirúrgico, Medicina Familiar, Medios de Diagnóstico, Enfermería y otros. En total se consideran 22 módulos y con la posibilidad de vincular otros.

En dicho sistema actualmente no se gestiona los nomencladores médicos tales como: Clasificación Internacional de Enfermedades (CIE), Clasificación Internacional para la Atención Primaria de la Salud (CIAP) y otros, no se garantiza la seguridad del sistema y no existe una interoperabilidad, que facilite el intercambio de datos en forma precisa, efectiva y consistente; y que además, haga uso de la información intercambiada, centradas en un registro médico electrónico de los pacientes con uso de un repositorio de datos clínico común. Así se logra un sistema íntegro capaz de inter-operar o comunicarse entre diferentes tecnologías y aplicaciones de software que admite reunir todos los datos de los pacientes, sin importar el lugar donde recibió algún tipo de cuidado, así como consultorios médicos, policlínicos, consultorios odontológicos, laboratorios, departamentos de imágenes, entre otros.

La mayoría de estas entidades en la actualidad, presentan deficiencias tecnológicas, las mismas poseen computadoras con pocas prestaciones, cuentan con 256 Mb de RAM y micro 2.0 GHz, estas entidades no cuentan con conexión, por lo que se trabaja en modo desconectado, en caso de que se pueda establecer la conexión se haría en un horario determinado del día a través del módem, que en su mayoría tienen un ancho de banda de 56 Kbps. A estas limitaciones se le suman las llamadas “Zonas de Silencio”, que no solo están dadas por la falta del fluido eléctrico o comunicación telefónica, sino que en este caso particular no presentan el cableado de fibra óptica necesaria para establecer una conexión, pues el cableado solo llega hasta las cabeceras provinciales y municipales.

Luego de haber analizado la situación existente, teniendo en cuenta las dificultades expuestas y la necesidad de buscar una solución a ellas, todos los esfuerzos estarán encaminados a solucionar el siguiente **problema científico:** ¿cómo configurar los parámetros necesarios para el funcionamiento del

Sistema Integral para la Atención Primaria de Salud en las unidades de salud con limitaciones tecnológicas?

El **objeto de estudio** se identifica como el proceso de gestión de la información en la Atención Primaria de Salud y el **campo de acción** está enmarcado en el proceso de configuración de los parámetros necesarios para el funcionamiento del Sistema Integral para la Atención Primaria de Salud en las unidades de salud con limitaciones tecnológicas.

El **objetivo general** de la investigación es desarrollar un módulo que configure los parámetros necesarios para el funcionamiento del Sistema Integral para la Atención Primaria de Salud en las unidades de salud con limitaciones tecnológicas.

Dado el anterior objetivo, se derivan las siguientes **tareas investigativas**:

1. Realizar un análisis valorativo del estado del arte de las soluciones existentes que disponen de un módulo de configuración para la salud.
2. Especificar la metodología y tecnologías informáticas definidas por el CESIM para el desarrollo de aplicaciones de escritorio.
3. Realizar el Modelo de Dominio teniendo en cuenta los conceptos asociados al problema.
4. Realizar la especificación de requisitos y el diseño de los prototipos no funcionales de interfaz de usuario.
5. Realizar el Modelo de Diseño según las pautas y estándares establecidos en el proyecto.
6. Implementar las funcionalidades que permitan el funcionamiento del SIAPS haciendo uso de los estándares de codificación establecidos.

Una vez culminadas las tareas se esperan obtener varios beneficios que garanticen el funcionamiento de los restantes módulos en desarrollo, entre las cuales se pueden enunciar:

1. Facilitar el uso de estándares definidos internacionalmente como CIE, CIAP, EDO y CPASP.
2. Garantizar la interoperabilidad entre los módulos del SIAPS que posibilite un correcto intercambio de datos.
3. Gestionar la información de manera eficiente para todo el nivel Atención Primaria de Salud.

4. Garantizar el soporte y la seguridad para los restantes módulos de SIAPS.

Este documento está compuesto por cuatro capítulos, los cuales contienen todo lo relacionado con el trabajo investigativo realizado.

CAPÍTULO 1. Fundamentación Teórica. En este capítulo se hace una valoración crítica de las soluciones existentes en el ámbito internacional, nacional y en la Universidad. Se profundiza, también, en el estudio de las técnicas, tendencias tecnológicas y metodologías usadas en la actualidad en la que se apoya la solución del problema.

CAPÍTULO 2. Características del Sistema. Se describe la propuesta del sistema, se justifican los patrones de diseño a utilizar en el desarrollo de la aplicación, se crea el Modelo de Negocio que contiene el Modelo de Dominio y la Captura de Requerimientos en el cual existe dos actividades fundamentales: la especificación de requerimientos funcionales y no funcionales, más el modelamiento del sistema.

CAPÍTULO 3. Diseño del Sistema. Se realiza el análisis del sistema a desarrollar, se justifican los patrones de diseño a utilizar durante la aplicación y los diagramas de clases del diseño, el diagrama de clases persistentes, el modelo de datos, aparte de la explicación de sus tablas y sus atributos. Como resultado del flujo de trabajo Diseño se generan los artefactos mencionados anteriormente, que son el punto de partida para la implementación del sistema.

CAPÍTULO 4. Implementación y prueba. En este capítulo se argumenta la justificación de la integración con otros componentes. Se muestra el modelo de implementación con el diagrama de despliegue y el de componentes, aborda la descripción de los estándares de diseño, codificación y el tratamiento de errores en la solución del sistema.

Capítulo 1. Fundamentación Teórica

Contar con un sistema informático que provea información adecuada y oportuna constituye el eje fundamental en la toma de decisiones y en las acciones a realizar para el cumplimiento de los objetivos y metas de una institución médica. El manejo correcto de la información puede potenciar las estrategias diagnósticas y terapéuticas, la administración del personal de la institución y los usuarios, la coherencia tecnológica y las inversiones de un centro médico.

Una de las necesidades de la modernidad en el ámbito informático, es en cómo configurar los servicios o recursos que se brinda a un sistema, de manera que los resultados sean buenos y beneficien a todos y cada uno de los involucrados.

La configuración hoy en día propone mejorar el proceso de toma de decisiones y controlar el sistema en todo momento. Propone, asimismo, tener disponible la información relevante de forma inmediata y hacer llegar la información a los usuarios adecuados en el momento preciso. Es un reto de la actualidad el estar al día en la teoría y aplicarla en la práctica para poder ser eficientes y humanos. Cómo configurar, implica saber qué y por qué se configura.

En este capítulo se caracteriza al Sistema Nacional de Salud en Cuba, se realiza una descripción sobre los conceptos asociados a la configuración de un sistema. Igualmente, se realiza un análisis sobre las soluciones existentes en el ámbito nacional e internacional como soporte teórico del sistema a desarrollar y se describe la situación problémica existente y a partir de este un problema a resolver.

Por último, se profundiza en el estudio detallado de las técnicas, tendencias tecnológicas y metodologías usadas actualmente en la que se apoya la solución del problema.

1.1 Marco conceptual

1.1.1 Sistema Nacional de Salud

Desde el triunfo de la Revolución se adoptaron medidas para transformar la salud pública en Cuba, una de las principales y más novedosa fue la creación del Sistema Nacional de Salud (SNS), designándose al Ministerio de Salud Pública (MINSAP) como su organismo rector, que es el encargado de dirigir, ejecutar y controlar la aplicación de la política del Estado y del Gobierno referente a la salud pública.

A partir de los años 60 esta estructura organizativa comenzó a realizar importantes reformas, como parte fundamental de las transformaciones del período revolucionario y en respuesta al respeto más absoluto de uno de los derechos humanos fundamentales de todo ciudadano. Surge el servicio de hospitales rurales llevando la atención médica a zonas apartadas de la geografía nacional y se dan los primeros pasos para el fortalecimiento de la atención primaria; surgen así los policlínicos integrales, como una unidad asistencial, creada para brindar servicios y resolver los principales problemas existentes en los primeros años de la revolución.

En la década del 80 aparece el Médico y la Enfermera de la Familia, los que sientan precedentes en la salud pública internacional por su carácter novedoso y futurista, especialmente con la implantación y desarrollo del modelo de atención de Medicina Familiar a partir de 1984.

Con el surgimiento del médico y la enfermera de la familia, se ratifica como el eje del actual desarrollo estratégico, orientándose una parte de las estrategias en función del mismo. Este modelo de atención es la mayor fortaleza y potencialidad que tiene el SNS. Por su existencia, filosofía, bases teóricas y lo que ha podido proporcionarle al sistema de salud, se ha logrado mantener los indicadores de salud y satisfacer las necesidades de la población, constituyendo un pilar básico de la Salud Pública Cubana.

El SNS está estructurado en tres niveles de atención médica, el nivel de atención primaria, que se brinda a nivel de los policlínicos y hospitales rurales a través del Programa de Medicina Familiar y abarca a todos los Equipos Básicos de Salud (EBS). Es la primera interacción que va a tener el paciente con el SNS; el nivel de atención secundaria, que se brinda a nivel de las instituciones hospitalarias, por lo general son de carácter provincial, o sea, atienden a toda la población de una provincia determinada y el nivel de atención terciaria, que por su condición muy especializada, sólo se brinda en determinados centros, ejemplo: Instituto de Neurocirugía, Instituto de Cirugía Cardiovascular, Instituto de Nefrología, Instituto de Gastroenterología, entre otros o en centros hospitalarios y de investigación categorizados como centros de referencia nacional y en algunos casos de referencia internacional.[1]

1.1.2 Problemas de Salud

Cualquier queja, observación o hecho que el paciente y/o el médico percibe como una desviación de la normalidad, que ha afectado, afecta, o puede afectar la capacidad funcional del paciente.

1.1.3 Interoperabilidad

Se define interoperabilidad como la capacidad que tienen dos o más sistemas para intercambiar información y utilizarla de inmediato en su lógica de proceso interno. En este contexto, hay que adoptar los estándares y los modelos de referencia que capacitan los sistemas y disminuyen los costos de implementación.

1.1.4 Configuración

Adaptar una aplicación software o un elemento hardware al resto de los elementos del entorno y a las necesidades específicas del usuario.

1.1.5 Equipo Básico de Salud

Binomio conformado por el médico y enfermera de la familia, que atiende una población geográficamente determinada, que pueden estar ubicados en la comunidad, en centros laborales o educacionales.

1.1.6 Prestaciones Médico Asistencial

Es cualquier prestación médica simple o compleja que se realiza a una persona como parte de un tratamiento.

1.1.7 Informatizar

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

1.1.8 Policlínico

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

1.1.9 Codificación

Transformación de la formulación de un mensaje a través de las reglas o normas de un código o lenguaje predeterminado.

1.2 Estándares de Salud

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

Estándar se define como norma o serie de reglas y definiciones que especifican cómo llevar a cabo un proceso o producir un producto. A lo largo de la historia, la aplicación de estándares permitió la reutilización de recursos y evitó la duplicidad de esfuerzos. Un ejemplo es el ancho de las vías del ferrocarril o el sistema ASCII que utilizan las computadoras. Para el desarrollo de cualquier aplicación es necesario la definición y adopción de estándares que permitan la integración y homogeneidad de los sistemas, esclareciendo que un estándar es toda regla, requisito o recomendación basada en principios ya probados y en la práctica.

En Informática para la salud, donde el énfasis está puesto en la manipulación y transmisión de información, los estándares se necesitan. Aunque en la actualidad están evolucionando tan rápido que cualquier descripción se desactualiza. La clave del éxito de estandarizar, está en comenzar definiendo los estándares necesarios y, posteriormente, crear sistemas de información basados en aquellos. De esta forma, se evita realizar inversiones de tiempo y dinero.

La interoperabilidad requiere de estándares por múltiples razones, para intercambiar datos y además para entender y hacer uso de los datos intercambiados. Es necesario adoptar estándares a todos los niveles de agregación (local, regional y nacional), para evitar que se rompa la cadena de interoperabilidad. Caso contrario nunca se logrará un sistema de salud integrado. El consenso sobre qué estándares utilizar y cómo utilizarlos juega un rol fundamental, ya que no existe un axioma sobre cómo llegar a la interoperabilidad.

Hoy en día existen estándares que se ocupan de la interoperabilidad, a modo práctico, se organizan en 6 categorías:

Intercambio de datos y mensajería: Permiten que las transacciones para el intercambio de datos fluyan de manera consistente entre los sistemas u organizaciones. Contienen las especificaciones o instrucciones necesarias para la estructura, formato y elementos del dato a intercambiar. Entre los más comunes se encuentran HL7 para datos como información demográfica del paciente o consultas y DICOM para imágenes.

Terminología: Proveen codificación específica para conceptos clínicos como: patologías, lista de problemas, alergias, diagnósticos y medicamentos que pueden tener o no variantes léxicas. A modo de ejemplo se tiene a SNOMED para términos clínicos, LOINC para resultados de laboratorio y CIE para enfermedades y causas de muerte.

Documentos: Indican el tipo de información que debe ser incluido en un documento y dónde puede encontrarse. Un estándar reconocido en los registros clínicos en papel es el formato SOEP (Subjetivo-Objetivo-Evaluación-Plan). El CCR (Continuity of Care Record) es un estándar para la comunicación de información de las patologías del paciente, medicaciones, antecedentes y plan de cuidado recomendado para ser compartido entre profesionales del equipo de salud.

Aplicaciones: Determinan la forma por la cual las reglas del negocio se implementan y las aplicaciones pueden interactuar. Esto incluye el log-in único para diversas aplicaciones dentro del mismo entorno y estándares para brindar una vista comprensiva de datos a través de múltiples bases de datos no integradas.

Conceptual: Permiten el transporte de datos a través de los sistemas sin que estos pierdan sentido y contexto. El HL7 RIM (Reference Information Model) provee el marco conceptual para la descripción de datos clínicos y el contexto que lo rodea explicando el "quién, qué, cuándo, dónde y cómo".

Arquitectura: definen el proceso involucrado en el almacenamiento y distribución de los datos. La red de información de salud pública del CDC y el sistema de registro electrónico nacional de enfermedades son ejemplos.

La necesidad actual de integrar, consolidar y coordinar la información de los pacientes por medio de la creación de redes que brinden cuidado médico, incorporando técnicas de gerenciamiento como indicadores de calidad y ofreciendo herramientas de avanzada como sistemas de soporte para la toma de decisiones exige la implementación de estándares.

1.2.1 Nomenclador de Prestaciones Médicas y Clasificaciones de Enfermedad

El lenguaje médico, en su estructura y su significación, no se ajusta a criterios lógicos uniformes y universales. Eso es debido a las vicisitudes históricas que han sufrido los términos, y a la dificultad inherente a la formación de los conceptos médicos. La ausencia de un criterio uniforme obstaculiza la comunicación científica y dificulta la recuperación de información almacenada. Para hacer frente a estos inconvenientes han surgido las nomenclaturas. La nomenclatura se define como un conjunto de las palabras técnicas y propias de una ciencia o facultad.

Un nomenclador de prestaciones médicas entonces viene siendo el catálogo de nombres de técnicas médicas destinadas al tratamiento de una enfermedad o patología. Este catálogo no implica la adición de un significado de cada técnica (si no se hablaría de un diccionario), ni a la descripción de

los procedimientos de la misma (si no se hablaría de un manual), sólo el nombre y ciertos aspectos que individualizan y valoran una prestación médica.

Los nomencladores de prestaciones médicas asistenciales son como instrumentos para controlar el cumplimento de las obligaciones médicas asistenciales de las entidades sanitarias, en el mismo, las pautas a tener en cuenta al momento de:

- Listar, clasificar y describir las prestaciones médico-asistenciales.
- Individualizar unívocamente una prestación.
- Valorar una prestación.
- Asociar procedimientos, pautas y entidades relacionadas a la valorización y normas de cobertura e interpretación de prestaciones médico-asistencial.

Por tanto, el objetivo principal de una nomenclatura para un dominio dado del conocimiento es la catalogación de todos los conceptos que alguna vez se hayan expresado en dicho ámbito. Una clasificación en cambio, busca organizar los conceptos expresados en ese mismo ámbito conforme a un esquema preconcebido, buscando simultáneamente discriminar los elementos del conjunto y agruparlos en subconjuntos. Los nomencladores y las clasificaciones médicas están estrechamente relacionados.

La codificación o clasificación de diagnósticos médicos surge a partir de la intención de tener una terminología estandarizada que permita transformar el significado de un diagnóstico a una representación independiente del lenguaje, de modo que permita que la información esté disponible para propósitos de compatibilidad e integración. Esta codificación de datos permite su agrupación y comparación estadística, ya sea entre diferentes escenarios o en un mismo escenario pero en diferentes momentos.

Clasificaciones:

Clasificación Internacional de Enfermedades (CIE): La CIE fue publicada por la Organización Mundial de la Salud. Se utiliza a nivel internacional para fines estadísticos relacionados con morbilidad y mortalidad, los sistemas de reintegro y soportes de decisión automática en medicina.

Este sistema está diseñado para promover la comparación internacional de la recolección, procesamiento, clasificación y presentación de estas estadísticas. Es una clasificación diagnóstica

internacional, estándar para todos los propósitos epidemiológicos generales y muchos otros de administración de salud. Esto incluye el análisis de la situación general de salud de grupos de población y el seguimiento de la incidencia y prevalencia de enfermedades y otros problemas de salud en relación con otras variables, tales como las características y circunstancias de los individuos afectados.

La lista de códigos CIE-10 es la décima versión de la Clasificación estadística internacional de enfermedades y otros problemas de salud (en inglés ICD, siglas de International Statistical Classification of Diseases and Related Health Problems) y determina los códigos utilizados para clasificar las enfermedades y una amplia variedad de signos, síntomas, hallazgos anormales, denuncias, circunstancias sociales y causas externas de daños y/o enfermedad. Cada condición de salud puede ser asignada a una categoría y recibir un código de hasta seis caracteres de longitud (en formato de X00.00). Cada una de tales categorías puede incluir un grupo de enfermedades similares.

Las Enfermedades de Declaración Obligatoria (EDO): Son enfermedades que por su importancia y repercusión se consideran como de constante vigilancia epidemiológica por tratados internacionales o intereses del país. Son enfermedades transmisibles que los médicos están obligados a notificar al centro de salud pública correspondiente por ser de especial importancia para la comunidad. Ejemplo de ellas se tiene el dengue, la tuberculosis, la hepatitis, la sífilis, la varicela, el SIDA, entre otras.

El programa de declaración se basa en distintas normas. Así, existe un Reglamento Sanitario Internacional que obliga a declarar enfermedades o situaciones que requieren una respuesta inmediata desde la perspectiva de Salud Pública, como también numerosas reglamentaciones europeas, nacionales y comunitarias que son de obligado cumplimiento por las distintas instituciones dedicadas a la salud.

El Registro de Enfermedades de Declaración Obligatoria (EDO) es de utilidad por varios motivos:

- Proporciona información necesaria para adoptar las adecuadas medidas de prevención y control ante una alerta comunitaria.
- Configura el perfil epidemiológico de las enfermedades objeto de notificación.
- Posibilita la evaluación de la efectividad de las acciones preventivas desarrolladas para controlar enfermedades transmisibles.

- Proporciona una orientación acerca de las prioridades sanitarias de la población.

Clasificación Internacional de la Atención Primaria (CIAP): El Comité Internacional de Clasificación de la WONCA desarrolló la primera versión de la Clasificación Internacional de la Atención Primaria - CIAP (International Classification of Primary Care - ICPC) en 1987, que se tradujo al español en 1990. Actualmente se ha traducido a más de 20 idiomas.

Dirigida para registrar la actividad realizada en Atención Primaria, pero solamente una se ha diseñado para describirla de forma global. Permite la recogida y análisis de tres importantes componentes de la consulta médico-paciente: la razón de consulta, el problema atendido, y el proceso de atención.

La CIAP-2 se publicó en 1998 en inglés y un año más tarde en español. Las dos grandes diferencias de la segunda versión de la CIAP con su predecesora son que incorpora criterios de inclusión / exclusión y correlaciona muchas rúbricas e incluye la correspondencia con la décima edición de la Clasificación Internacional de Enfermedades que la OMS editó en 1992 (CIE-10). La experiencia acumulada con la CIAP-1 obliga a que las clasificaciones no deban ser modificadas por el usuario, ya que se pierde una de sus funciones, la comparabilidad. Independientemente de las limitaciones comentadas, en el empleo inteligente y en la explotación de la historia clínica electrónica, el primer paso obligado es tener la CIAP-2 E, en español.

1.3 Estado del arte

Durante los últimos años las instituciones cubanas y el propio MINSAP, han desarrollado sistemas encaminados a lograr la informatización de la salud. En todos los casos el objetivo ha sido abastecer al SNS de información confiable, consistente y oportuna para la toma de decisiones y el mejoramiento de los procesos médicos asistenciales, garantizando de esta manera el incremento en la calidad y seguridad de la atención médica a la población.

La informatización del Sistema Nacional de Salud Pública está dada por un conjunto de métodos, técnicas, procederes y actividades gerenciales dirigidas al manejo de la información en la salud, la cual comprende la información sobre el estado de salud de la población, la información sobre el conocimiento de las ciencias de la salud y la información en general para la toma de decisiones, clínico-epidemiológicas, operativas y estratégicas.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

La informatización del SNS cubano comenzó por la Atención Primaria de la Salud (APS), ya que es el nivel más bajo de la cadena de atención médica a la población. Tiene como principales objetivos contar con módulos especializados en la gestión de los procesos generados en el policlínico con programas asociados a la misma, que faciliten la correcta toma de decisiones en todos sus niveles. El centro de gestión de toda la información en salud estará en el policlínico y contendrá la información que generen los Equipo Básico de Salud (EBS) por cada Grupo Básico de Trabajo (GBT) y de los servicios propios del policlínico.

Durante los últimos 20 años un grupo de instituciones cubanas desarrollan sistemas encaminados a lograr determinados niveles de informatización de la salud. Estas soluciones carecían de integración y de una definición generalizable, aparte de que no existían los recursos tecnológicos necesarios para su ejecución en el SNS. A partir de 1997 se concibe una primera estrategia de informatización como respuesta del sector de la salud a los lineamientos estratégicos para la informatización de la sociedad cubana, con la finalidad de coordinar esfuerzos para el desarrollo de este proceso en el SNS.

En este año se inició una etapa planeada de introducción de las nuevas tecnologías de la información y las comunicaciones en el marco del proceso de informatización de la sociedad cubana, que da paso a una primera estrategia de informatización del sector de la salud. En el período 1998-2000, se trabajó la primera etapa planeada a través de los objetivos estratégicos maestros y los planes de acción para la capacitación de los recursos humanos, la seguridad informática e informatización de los servicios.

Desde el inicio de 2001 se realizaron profundos cambios en métodos y estilos de trabajo en el sector de la salud que propiciaron un salto cualitativo en el desarrollo estratégico de la informatización. En la actualidad se trabaja en proyectos basados en nuevas tecnologías de Internet, software libre y otros que garantizan una explotación integrada y compatible y que serán introducidos utilizando como infraestructura de la Red Telemática de la Salud.

Los sistemas informáticos están dirigidos a mejorar la calidad de atención, aumentar la productividad y disminuir los costos con el objetivo de obtener clientes contentos, compradores que han pagado por el desarrollo de aplicaciones que deben ser ágiles, sencillas, intuitivas y usables. En la actualidad, aún las aplicaciones de escritorio siguen siendo la opción más potente y cómoda de usar.

Estas aplicaciones presentan algunas diferencias con respecto a las aplicaciones Web convencionales, se caracterizan por mejorar la experiencia del usuario en cuanto a audio, video y comunicaciones. Son más interactivas que las aplicaciones Web convencionales, ya que mantienen un contacto permanente entre los procesos internos del programa y lo que sucede en la interfaz de usuario. Permiten acciones tales como arrastrar y pegar documentos, textos e imágenes.

Uno de los mayores problemas que presentan es que la aplicación junto con sus actualizaciones, deben ser instaladas en cada computador utilizado. El mundo del escritorio es muy amplio y el objetivo concreto de la aplicación puede determinar la tecnología a elegir. Si la aplicación debe ser multiplataforma se recomienda Java, si la aplicación debe tener una fuerte integración con el SO, entonces se recomienda aplicaciones nativas como C/C++.

Estrategias para optimizar el uso de aplicaciones de escritorio:

- **Minimizar los accesos a recursos externos.**

Con el paso de los años han ido surgiendo una serie de bases de datos ligeras, capaces de ejecutarse en procesos Java e ideales para incrustar dentro de aplicaciones de escritorio. Uno de los usos más interesantes de estas bases de datos es el de servir como sistemas de “backup” cuando se pierde la conexión con una base de datos central. Utilizar una de estas bases de datos incrustadas, si la aplicación no tuviese conexión con el servidor de base de datos, almacenaría las operaciones realizadas, guardando cambios, almacenando propiedades y nuevos registros, o simplemente realizando consultas sobre un histórico. Una vez que la conexión se restablezca, en ese momento se volcaría de nuevo toda la información.

- **Utilizar la ayuda de algún framework.**

Normalmente, lo que menos tienen los desarrolladores es precisamente tiempo. Existen en la actualidad varias “suites” de componentes de interfaz de usuario listas para utilizar, y que se deben aprovechar siempre que sea posible.

- **Utilizar hilos de trabajo de manera extensiva.**

Los procesos que no dependen para nada del usuario, y que no interfieran en sus futuras acciones, se deben intentar ejecutarlos de un modo que no molesten. Una de las formas de realizar esto es

ejecutarlos en un hilo en segundo plano. De este modo, el usuario puede seguir trabajando con la aplicación mientras la tarea costosa se realiza de manera transparente.

- **No cargar la información que no se necesita.**

Las aplicaciones de escritorio tienen una filosofía de funcionamiento muy diferente de las aplicaciones web. Al disponer de una cantidad mayor y más completa de componentes de interfaz de usuario, se tiene la oportunidad de poder descargarse gran cantidad de información de una tacada y desplegar dicha información sobre los árboles, tablas, listas, entre otros, que forman el interfaz de usuario.

- **Precargar toda la información útil que se pueda.**

La idea de precargar la información que el usuario va a usar, proporciona un ahorro importante en cuanto a tiempo de acceso a servidor, base de datos, entre otros, ya que los datos se obtendrán de golpe y pocos accesos a base de datos.

- **Presentar el interfaz de la aplicación tan rápido como sea posible.**

Mostrar la interfaz de usuario lo más rápido que sea posible. Si se muestra la pantalla de inicio de la aplicación, la pantalla de entrada al sistema, o la ventana principal de la aplicación de una manera rápida, el usuario estará mucho más satisfecho, y además estará entretenido mientras va cargando en segundo plano todos los datos que se necesita.

- **Avisar al usuario de lo que está pasando.**

El usuario debe saber en todo momento lo que se está realizando, o al menos que se está ejecutando algo.

- **Evitar el presentar enormes cantidades de datos.**

Evitar realizar consultas demasiado genéricas para agilizar la aplicación y el funcionamiento general del sistema. Garantizar que se realicen consultas mucho más específicas, concretas, y sobre todo mucho más rápidas y que impactan mucho menos en el rendimiento de los servidores. Si los usuarios se encuentran familiarizados con esta forma de trabajar, con resultados paginados, se puede intentar ofrecer una interfaz similar.

- **¨Conocer a tu framework como a ti mismo¨.**

Siempre que se implante alguna librería, aunque parezca tan inofensiva, se debería comprobar diferentes parámetros de funcionamiento de la aplicación, como el rendimiento, el consumo de memoria, el acceso a disco, memoria virtual utilizada; y poder, utilizar algún “profiler” (JMemProf, JProbe y JMP) que dé datos exactos y no meras intuiciones.

Intentar seguir estas estrategias para optimizar el uso de aplicaciones de escritorio, ayuda a crearlas mucho más eficientes y agradables al usuario. Esto significa, que la creación de interfaces de usuario no es arrastrar dos botones sobre un panel con el maravilloso entorno de desarrollo. La creación de interfaces de usuario es también planificar la creación de hilos en segundo plano, el decidir cómo, cuantos y que datos se van a precargar, el meditar como se van a visualizar los informes de las aplicaciones, el pensar cual framework pueden ahorrar trabajo, o el planear el método de visualización de cargas masivas. Todo ese análisis, hará que realmente tenga éxito una aplicación de escritorio.

La necesidad de integrar, consolidar y coordinar la información, incorporando técnicas de gerenciamiento como indicadores de calidad y ofreciendo herramientas de avanzada como sistemas de soporte para la toma de decisiones, exige la implementación de estándares. La diversidad de información crea ineficiencia o impide efectividad, por lo que se necesitan de estándares.

Tradicionalmente, las instituciones de salud fueron concebidas como unidades independientes donde los pacientes recibían atención médica en el nivel primario, secundario o en el terciario, sin ningún tipo de comunicación bi-direccional ni coordinación efectiva. La falta de coordinación entre los distintos escenarios y actores del sistema se traduce en ineficiencias en el cuidado. Es vital estandarizar identificadores para individuos, profesionales de la salud, proveedores, financiadores y en general toda la información que se requiera un sistema, para que todos puedan ser reconocidos a través de estos u otros sistemas informáticos.

Cuando se lleva a cabo el proceso de estandarización en un sistema informático, de alguna manera, se está configurando ese sistema, es decir, se trazan pautas y se definen los objetivos generales, en ese sentido, se estandariza. Este es uno de los objetivos más sustanciales que aporta a la configuración.

De esta manera, la configuración concibe que cada parte de una aplicación cumpla una función específica encomendada, adaptando una aplicación de software o un elemento de hardware al resto

de los elementos del entorno y a las necesidades específicas del usuario. Tener disponible la información relevante de forma inmediata y hacer llegar la información a los usuarios adecuados en el momento preciso, es una de las necesidades más básicas en la actualidad, como también, es un reto optimizar los servicios o recursos que se brinda a un sistema.

La configuración será la que determine a través de qué medios y con qué recursos funcionará, pero sin embargo, este conjunto de informaciones puede ser alterado si se considera necesario, tanto para corregir un error, como para dar nuevas funciones o redefinir el elemento en diferentes modos.

Hay dos tipos principales de configuración, las predeterminadas y las personalizadas. La configuración predeterminada es la que está dada y que puede existir automáticamente, la configuración personalizada es aquella realizada por el usuario con un objetivo específico. Salvo algunos casos, nunca es recomendado mantener una configuración predeterminada, ya que además de no seguir los intereses o necesidades personales, también puede ser fácilmente alterada por agentes externos como virus y hackers. En cambio, la configuración personalizada transformará al elemento en cuestión, en algo mucho más útil y a la vez seguro.

Un ejemplo en ámbito internacional donde se demuestre el análisis realizado sobre las aplicaciones de escritorio y de las configuraciones que se lleven a cabo en los sistemas informáticos dirigidos a la salud, es el Sistema de Información para la Gerencia Hospitalaria (SIGHO), liberado para el apoyo a la gestión de todos los Hospitales del sector salud en México.

Software basado en la Norma Oficial Mexicana NOM-168-SSA1-1998 referente al resguardo y uso del expediente clínico electrónico para facilitar las actividades de gerencia dentro del hospital y se apoya de estándares internacionales para el diagnóstico de enfermedades y realización de procedimientos, tales como el CIE-10 y CIE9MC. Se compone de 14 módulos.

Su módulo de configuración contiene las opciones de parámetros necesarios para el correcto funcionamiento de otros módulos del SIGHO, concentra los catálogos de información que se requieren en común en módulos como Estados, Jurisdicciones, Municipios, Localidades, Servicios y Áreas médicas, por citar algunas. Este sistema se desarrolló en Visual Basic, lenguaje de programación que no recibe soporte desde abril del 2008 por parte de la empresa Microsoft. Utiliza como Gestor de Bases de Datos SQLServer2005 y el sistema tiene como características que requiere de una conectividad permanente.

Otro software en este ámbito es el Sistema Integral de Administración de la Salud, reconocido en el mundo como el ÁNGEL. Este es el primer software médico completo y gratuito en la Argentina que permite establecer un sistema electrónico integral de historias clínicas para la provisión de los servicios sanitarios. El sistema permite que toda decisión o acción se almacene en un registro médico electrónico e incorporar datos socioeconómicos y ambientales para focalizar programas de prevención y asistencia en los sectores sociales más necesitados.

El sistema es operable sobre Windows y Linux. Los requerimientos mínimos de equipamiento: Pentium II 64 Mb. Es compatible con HL7. Puede funcionar en red, pero no es requisito para su funcionamiento. El sistema se implementó en Java y es independiente de base de datos. El software es gratuito, pero el código no es abierto para modificaciones. No tiene módulos para todas las especialidades y no posee un módulo de configuración. Y entre otras características la falta de clasificaciones de enfermedades como (CIAP) y (EDO), el Ángel solo realiza búsquedas por (CIE).

Un ejemplo en el ámbito nacional por automatizar los procesos en la salud, es el Registro Automatizado de Gerencia de APS (APUS). Este permite el diagnóstico de salud en sectores como consultorios y área del policlínico, y el análisis demo-sociológico y clínico-epidemiológico de los problemas de salud del individuo, la familia, la vivienda y el ambiente. Facilita el control de las actividades y servicios de salud prestados, la vigilancia de salud comunitaria, y el monitoreo y evaluación de los programas de APS.

APUS mejoró algo en la eficiencia de trabajo del policlínico, no así de los consultorios, este confrontó una difícil codificación clínica de muchos problemas de salud debido al desajuste a las clasificaciones estándares de enfermedades y de problemas de salud en APS y la poca aceptación médica debido al llenado más cuidadoso y lento de datos en sus modelos. El sistema no posee módulo de configuración y se implementó en Borlan Delphi, lenguaje propietario de la empresa Embarcadero Technologies, y además utiliza como gestor de bases de datos SQLServer2000.

Todos estos sistemas informáticos, dirigidos a la adquisición, procesamiento e interpretación de datos de pacientes, de alguna manera u otra se lleva a cabo la configuración, la mayoría de ellos o son propietarios o no tienen el código fuente libre para modificaciones, esto puede traer como consecuencia que para su uso se tendría que invertir una gran suma de dinero y la carencia de sistemas que abarquen las funciones esenciales de la APS con enfoque integral y el desajuste al SNS.

A pesar de los intentos por automatizar las actividades que se llevan a cabo en la Atención Primaria para la Salud, unido a una configuración que haga uso de estándares y clasificaciones internacionales de los problemas de salud, hacia el correcto manejo de la información, no se pudo obtener una solución confortable. Si se realizara cambios en alguna de ellas para obtener de esta una solución mucho más confortable, se invertiría dinero, tiempo y esfuerzo, y es exactamente lo que se debe ahorrar en estos tiempos. Por lo que no es factible el uso de estos sistemas para solucionar el problema existente en la APS.

1.4 Problema a resolver y situación problémica

Durante el transcurso de los años, el país ha logrado altos indicadores en la esfera de la salud pública gracias a un sistema de salud en constante desarrollo y evolución. El gobierno se ha dado a la tarea de formar un elevado número de médicos y enfermeras y ha priorizado todos los aspectos relacionados con la informatización de la Salud Pública. A nivel mundial, la APS presenta una serie de problemas relacionados principalmente con la gestión de la información, impedimentos que frenan en gran medida la capacidad de respuesta de estos centros a la población, trayendo como resultado que se vea afectada la eficacia asistencial ofrecida y con ello la calidad de vida de los pacientes.

Un problema común al que se enfrenta la APS en Cuba en los últimos años es la demanda excesiva. Ésta puede ser definida como la que conlleva masificación y, como consecuencia, una disminución del tiempo de atención deseable para cada paciente o un alargamiento excesivo de la duración de la consulta a demanda, en detrimento de otras actividades del centro.

La cual se manifiesta a través de dos indicadores:

- La presión o sobrecarga asistencial que no es más que el número total de pacientes atendidos por el profesional sanitario al día.
- La frecuentación que es el número de visitas por paciente al año.

La sobrecarga asistencial tiene como consecuencias principales una disminución del tiempo de consulta dedicado al paciente, muy por debajo actualmente de los 10 minutos que reclaman la mayoría de los profesionales, incapacidad de desarrollar programas de prevención y promoción de la salud y una mayor prescripción, lo que a su vez genera un aumento de la frecuentación por parte del

usuario, con lo que se entra en una espiral de deterioro progresivo de la calidad de la asistencia que se presta en los Centros de Salud.

Además, la recopilación de información se realiza de forma manual y en papel, demandando una porción muy significativa de tiempo. Según estudios realizados, se demostró que por cada hora de cuidado directo sobre el paciente se generaba entre 30 y 60 minutos de registro en papel, aumentando la cantidad de documentación acumulada y provocando el deterioro y la pérdida de información, de gran importancia para el establecimiento de actividades de promoción, investigación, prescripción, predicción y prevención de las enfermedades del paciente, cómo necesidad indispensable para la atención familiar, haciendo que el trabajo sea mucho más complejo y engorroso.

Por otra parte, el territorio nacional presenta lugares donde la tecnología de la informática y las comunicaciones aún no han llegado, por lo intrincada que se encuentran estas zonas, con carreteras estrechas y curvas peligrosas. A estas regiones se les conoce como "zonas de silencio" de Cuba, están situadas fundamentalmente en las regiones montañosas más remotas del país, que poseen alrededor de unos 19 000 Km2 de zonas montañosas las cuales están pobladas aproximadamente por 820 000 habitantes. [2]

Estas zonas no solo se caracterizan por falta del fluido eléctrico o comunicación telefónica, sino que en este caso particular falta el cableado de fibra óptica necesaria para establecer una conexión. Como consecuencia, algunos centros de salud estarán totalmente aislados sin conexión. Muchas de estas unidades de salud carecen de computadoras o que estas presentan bajas prestaciones, o sea, cuentan en su generalidad con 256 Mb de RAM y un microprocesador de 2.0 GHz.

Esto trae como consecuencia que el médico actualmente trabaja en modo desconectado, las unidades que puedan realizar en algún momento del día la conexión, lo harán a través del módem, que en su mayoría, poseen un ancho de banda de 56 Kbps.

Otro problema actual de la APS está enmarcado en la dificultad de informatizar los servicios sanitarios que se prestan a la población, problema analizado a fondo en busca de soluciones. La complejidad de este proceso en el SNS es sumamente alta, debido a la variabilidad de unidades de APS, con objetivos concretos y enfocados a la función que desempeña la institución, enmarcadas en

un lugar físico diferente, siendo un elemento esencial la comunicación e intercambio de información entre dichas entidades.

La inexistencia de sistemas informáticos que abarquen todos los procesos principales del espectro de la atención primaria de salud, es un problema a resolver. Debido a la gran variedad de estructuras o formas organizativas que presenta el primer nivel de atención, el MINSAP y el Ministerio de la Informática y las Comunicaciones (MIC), con la colaboración de la Universidad de Ciencia Informáticas (UCI) y el Centro Especializado en Soluciones para la Informática Médica (CESIM), deciden el desarrollo y completamiento de la informatización de la APS.

El Sistema Integral para la Atención Primaria de Salud (SIAPS) tiene como objetivo la solución a múltiples inconvenientes vinculados al sector de la salud y abarcar todos los procesos principales del espectro de la atención primaria de salud en la comunidad, dando un paso de avance en la prevención, investigación, recuperación y predicción de los problemas de salud del paciente. Este incluye módulos como: Clínico Quirúrgico, Medicina Familiar, Medios de Diagnóstico, Enfermería y otros. Se deberá tratar a los procesos que se lleva a cabo en dichas estructuras como módulos en el sistema informático, cada uno presenta características y objetivos específicos de acuerdo a la función que realice en los centros de salud, poniendo en práctica el concepto de interoperabilidad.

Pero en dicho sistema no se gestiona los nomencladores médicos tales como CIE, CIAP, EDO y CPAPS. No se garantiza la seguridad del sistema y no existe una interoperabilidad que permita el intercambio de datos en forma precisa, efectiva y consistente, y además que haga uso de la información intercambiada, centrada en un registro médico electrónico de los pacientes con uso de un repositorio de datos clínico común.

El consenso sobre qué estándares utilizar y cómo utilizarlos juega un rol fundamental, ya que no existe un axioma sobre cómo llegar a la interoperabilidad. Actualmente existe deficiencias en cuanto al uso preciso de estándares y de clasificaciones internacionales de los problemas de salud, proceso de codificación en el cual se precisen y estandarizan cada una de las entidades a medir mediante un vocabulario controlado, todas estas deficiencias hacen engorrosa la labor que realizan los especialistas.

De acuerdo con las dificultades recogidas anteriormente en el proceso de asistencia médica, los avances en las TIC benefician en gran medida el desarrollo de un sistema informático, en el cual se

espera obtener una solución que frene los impedimentos que afectan en gran medida la capacidad de respuesta de los centros salud, trayendo como resultado que se eleve la eficacia asistencial y con ello la calidad de vida de los pacientes.

Como también, mejorar la eficiencia de los servicios, dotando al sistema de un aumento del acceso a la información unificada, permitiendo su integridad, control y su seguridad, para que llegue de forma rápida, actualizada y de manera eficiente. Del mismo modo, se espera lograr una interoperabilidad o interrelación entre aplicaciones de la APS definidos como módulos, garantizando el buen funcionamiento en el intercambio de los datos. Y además, permitir el manejo adecuado de la codificación de los problemas de salud y la utilización de estándares reconocidos internacionalmente.

1.5 Tecnologías actuales a considerar y utilizar

1.5.1 Arquitectura de Software

La Arquitectura del Software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán, y los principios que orientan su diseño y evolución.

La Arquitectura del Software es el diseño de más alto nivel de la estructura de un sistema, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información. Establece los fundamentos para que analistas, diseñadores, programadores, etc. trabajen en una línea común que permita alcanzar los objetivos del sistema de información, cubriendo todas las necesidades.

1.5.2 Arquitectura en Tres Capas

El uso de una arquitectura en capas, representa uno de los estilos que aparecen con mayor frecuencia. De forma general, se define el estilo en capas como una organización jerárquica, que separa la lógica del negocio de la lógica del diseño, de forma tal que cada capa, suministra servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. [3]

Las 3 capas fundamentales son:

- **Capa de Presentación:** es con la que interactúa el usuario, presenta el sistema al usuario, le comunica la información y captura la información del usuario en un mínimo de proceso. Esta

capa se comunica únicamente con la capa de negocio. También es conocida como interfaz gráfica y debe tener la característica de ser entendible y fácil de usar para el usuario.

- **Capa de Negocio:** es donde se encuentran los programas que se ejecutan, se reciben las peticiones del usuario y se envían las respuestas tras el proceso. Se nombra así porque es aquí donde se implantan las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos para almacenar o recuperar datos de él. [4]
- **Capa de Datos:** es donde residen los datos y es la encargada de acceder a los mismos. Está formada por uno o más gestores de bases de datos que realizan todo el almacenamiento de datos, reciben solicitudes de almacenamiento o recuperación de información desde la capa de negocio. [5]

Ventajas de la Arquitectura por Capas

- Centralización de los aspectos de seguridad y transaccionalidad, que serían responsabilidad del modelo.
- No replicación de lógica de negocio en los clientes: esto permite que las modificaciones y mejoras sean automáticamente aprovechadas por el conjunto de los usuarios, reduciendo los costos de mantenimiento.
- Mayor sencillez de los clientes.

La ventaja principal de este estilo es que se puede desarrollar en varios niveles y, en caso de que ocurra algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado. Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de niveles.

1.5.3 Arquitectura Basada en Componentes

La arquitectura basada en componentes describe una aproximación de ingeniería de software al diseño y desarrollo de un sistema. Esta arquitectura se enfoca en la descomposición del diseño en componentes funcionales o lógicos que expongan interfaces de comunicación bien definidas. Esto provee un nivel de abstracción mayor que los principios de orientación por objetos y no se enfoca en

asuntos específicos de los objetos como los protocolos de comunicación y la forma como se comparte el estado.

El estilo de arquitectura basado en componentes tiene las siguientes características:

- Es un estilo de diseño para aplicaciones compuestas de componentes individuales.
- Pone énfasis en la descomposición del sistema en componentes lógicos o funcionales que tienen interfaces bien definidas.
- Define una aproximación de diseño que usa componentes discretos, los que se comunican a través de interfaces que contienen métodos, eventos y propiedades.

Los siguientes son los principales beneficios del estilo de arquitectura basado en componentes:

- **Facilidad de Instalación:** Cuando una nueva versión esté disponible, usted podrá reemplazar la versión existente sin impacto en otros componentes o el sistema como un todo.
- **Costos reducidos:** El uso de componentes de terceros permite distribuir el costo del desarrollo y del mantenimiento.
- **Facilidad de desarrollo:** Los componentes implementan un interface bien definida para proveer la funcionalidad definida permitiendo el desarrollo sin impactar otras partes del sistema.
- **Reusable:** El uso de componentes reutilizables significa que ellos pueden ser usados para distribuir el desarrollo y el mantenimiento entre múltiples aplicaciones y sistemas.
- **Mitigación de complejidad técnica:** Los componentes mitigan la complejidad por medio del uso de contenedores de componentes y sus servicios. Ejemplos de servicios de componentes incluyen activación de componentes, gestión de la vida de los componentes, gestión de colas de mensajes para métodos del componente y transacciones.

1.5.4 Patrones Arquitectónicos y de diseño

1.5.4.1Modelo Vista Controlador

Modelo Vista Controlador (MVC) es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee

de datos dinámicos a la página. El modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio, y el controlador es el responsable de recibir los eventos de entrada desde la vista.

Elementos del patrón:

- **Modelo:** Datos y reglas de negocio.
- **Vista:** Muestra la información del modelo al usuario.
- **Controlador:** Gestiona las entradas del usuario.

1.5.5 Patrones de Diseño

1.5.5.1 Singleton

El patrón de diseño **Singleton** (instancia única) está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella.

La instrumentación del patrón puede ser delicada en programas con múltiples hilos de ejecución. Si dos hilos de ejecución intentan crear la instancia al mismo tiempo y ésta no existe todavía, sólo uno de ellos debe lograr crear el objeto. La solución clásica para este problema es utilizar exclusión mutua en el método de creación de la clase que implementa el patrón.

El patrón Singleton provee una única instancia global gracias a que:

- La propia clase es responsable de crear la única instancia.
- Permite el acceso global a dicha instancia mediante un método de clase.
- Declara el constructor de clase como privado para que no sea instanciable directamente.

1.5.5.2Lazy

El Lazy Loading es un patrón de diseño útil y simple que se basa en ir cargando los distintos componentes de una clase cuando se vaya a utilizar. Este es comúnmente utilizado en programación de computadoras para aplazar la inicialización de un objeto hasta el punto en que sea necesario. Puede contribuir a la eficiencia en la operación del programa, si se utilizan adecuadamente.

1.5.5.3 Abstract Factory

El patrón Abstract Factory ofrece una interfaz para la creación de familias de productos relacionados o dependientes sin especificar las clases concretas a las que pertenecen. Su objetivo es soportar múltiples estándares (MS-Windows, Motif, Open Look). Es extensible para futuros estándares. Entre sus principales restricciones está cambiar el Look-and-Feel sin recompilar y cambiar el Look-and-Feel en tiempo de ejecución. El problema que intenta solucionar este patrón es el de crear diferentes familias de objetos.

Ventajas:

- Aísla las clases de implementación, ayuda a controlar los objetos que se creen y encapsula la responsabilidad de creación.
- Hace fácil el intercambio de familias de productos sin mezclarse, permitiendo configurar un sistema con una de entre varias familias de productos:

  Cambio de factory ⇨ Cambio de familia.

- Fomenta la consistencia entre productos.

1.5.6 Lenguaje de Programación

Un lenguaje de programación es cualquier lenguaje artificial que puede utilizarse para definir una secuencia de instrucciones para su procesamiento por un ordenador o computadora. Es complicado definir qué es y qué no es un lenguaje de programación. Se asume generalmente que la traducción de las instrucciones a un código que comprende la computadora debe ser completamente sistemática. Normalmente, es la computadora la que realiza la traducción.

1.5.6.1 Java

Es un lenguaje de programación sencillo, orientado a objetos, de propósito general e independiente de la plataforma de desarrollo, cuya sintaxis es muy parecida al lenguaje C o C++, fue creado por James Gosling, cumpliendo así con el slogan de la compañía "compilar una vez y ejecutar en cualquier parte".

Java es un lenguaje de programación con el que se puede realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de

Internet como en la informática en general. Está desarrollado por la compañía Sun Microsystems con gran dedicación y siempre enfocado a cubrir las necesidades tecnológicas más punteras.

Entre sus principales características se tiene que es un lenguaje independiente de la plataforma lo que posibilita que un programa en Java pueda funcionar en cualquier ordenador, la independencia de plataforma es una de las razones por las que Java es interesante para Internet. [6]

Ventajas frente a otros sistemas operativos:

- Simple y poderoso.
- Seguro.
- Orientado por Objetos.
- Robusto.
- Interactivo.
- Independiente de arquitectura de hardware.
- Interpretado y rápido.
- Fácil de aprender.
- Herramientas poderosas: Threads, excepciones, APIs, RMI, Beans.

1.5.7 Software libre

Software Libre se refiere a la libertad de los usuarios para ejecutar, copiar, distribuir, estudiar, cambiar y mejorar el software. De modo más preciso, se refiere a cuatro libertades de los usuarios del software: [7]

- La libertad de usar el programa, con cualquier propósito (libertad 0).
- La libertad de estudiar el funcionamiento del programa, y adaptarlo a las necesidades (libertad 1). El acceso al código fuente es una condición previa para esto.
- La libertad de distribuir copias, con lo que puede ayudar a otros (libertad 2).

- La libertad de mejorar el programa y hacer públicas las mejoras, de modo que toda la comunidad se beneficie (libertad 3). De igual forma que la libertad 1 el acceso al código fuente es un requisito previo.

Un programa es software libre si los usuarios tienen todas estas libertades. Así pues, deberías tener la libertad de distribuir copias, sea con o sin modificaciones, sea gratis o cobrando una cantidad por la distribución, a cualquiera y a cualquier lugar. El ser libre de hacer esto significa, entre otras cosas que no tienes que pedir o pagar permisos. [8]

1.5.8 Sistema Operativo Linux

Linux, sistema operativo derivado de UNIX que manteniendo la generalidad de sus características, como el ser multitarea y basado en bibliotecas dinámicas, puede ser ejecutado en computadoras u ordenadores personales aunque su potencia sea limitada. [9]

En sus orígenes fue desarrollado, en 1990, por el informático finlandés Linus Torvalds, que publicó su código como un denominado código abierto, esto es, accesible para toda la comunidad, sin restricciones para modificarlo y ampliarlo. Este planteamiento, favorecido por su estructura modular (basado en la instalación de diversos paquetes), generó una nueva visión de desarrollo informático, ya que su expansión fue debida a la aportación, generalmente voluntaria y sin ánimo de lucro, de multitud de desarrolladores independientes.

En la actualidad este sistema operativo ha obtenido un cierto apoyo por parte de la industria, de forma que empresas como IBM o Hewlett-Packard lo integran en algunas de sus computadoras y prestan el soporte técnico correspondiente, normalmente, como parte de los sistemas servidores. Su implantación en sistemas para usuarios finales, aún no ha alcanzado la extensión que tiene en algunos de los ámbitos más profesionales, muy especialmente en servidores de Internet.

1.5.9 Sistemas de Gestión de Base de Datos (SGBD)

Los sistemas gestores de Base de Datos son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. Se compone de un lenguaje de definición de datos, un lenguaje de manipulación de datos y un lenguaje de consulta.

Estos permiten manejar con facilidad grandes volúmenes de información en muy poco tiempo y ofrecen independencia y seguridad en el tratamiento de información.

Existen varios objetivos que deben cumplir los SGBD, dentro de los cuales se consideran más importantes los siguientes:

- **Seguridad:** Los SGBD deben disponer de un sistema de permisos a usuarios y grupos de usuarios, que permitan restringir los accesos no autorizados.
- **Integridad:** Se debe proteger la información almacenada ante fallos de hardware, datos introducidos por descuido o cualquier situación que pueda alterar la integridad de los datos almacenados.
- **Tiempo de respuesta:** Se debe minimizar el tiempo que el SGBD tarda en proporcionar la información solicitada y en guardar los cambios efectuados.
- **Abstracción de la información:** Los usuarios deben ser ajenos a detalles sobre la localización física donde han sido almacenados los datos.
- **Redundancia mínima:** Un diseño correcto de una BD debe evitar la duplicación o redundancia de la información.
- **Independencia:** Debe ser posible modificar la BD sin necesidad de realizar cambios en las aplicaciones que la accedan.

El propósito general de los sistemas de gestión de Base de Datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante, para un buen manejo de datos.

1.5.9.1HSQLDB 1.8.0

HSQL es un motor de bases de datos SQL ligero (611 Kb aprox.), OpenSource e implementado completamente en Java. Ideal para realizar pruebas sin tener que conectarse a un gestor de bases de datos, o para aplicaciones en donde en vez de querer administrar la información en ficheros binarios y hacer los métodos de consulta, borrado, etc. HSQL permite la creación de índices, control de la integridad referencial, sentencias de definición de datos, etc. Ahorra bastante tiempo de desarrollo y al trabajar sobre bases de datos es flexible a la hora de realizar cambios en la información a tratar o cuando los requisitos de la información a tratar cambien.

HSQLDB 1.8.0 apoya el dialecto de SQL definido por las normas de SQL 92, 99 y 2003. Esto significa que cuando una característica de la norma es compatible, por ejemplo, externa izquierda, la

sintaxis es la que especifica el texto estándar. Muchas de las características de SQL-92 y el 99 hasta el nivel avanzado, son compatibles y no hay apoyo de la mayoría de SQL 2003 Fundación y varias características opcionales de esta norma.

HSQLDB también soporta algunas palabras y expresiones que no son parte del estándar SQL como mejoras.

Es una base de datos que según sus creadores puede llegar a manejar varios GB de DATA (8GB para tablas de disco), que trabaja con estándar SQL92 y que soporta la creación de procedimientos almacenados (escritos en Java). Cuenta con una herramienta para la ejecución de sentencias SQL (Manager Swing) creada en Java Swing. Y finalmente cuenta con un Driver para el trabajo con Java, que es capaz de ejecutarse dentro del espacio de memoria de la propia aplicación. Las tablas y datos se cargan desde un fichero de script SQL. Esto permite una velocidad muy alta, el ahorro de tener que poner un MySQL o un Oracle para cualquier cosa, y la posibilidad de que sólo se permitan conexiones desde dentro de la propia máquina virtual de la aplicación, lo que aporta un extra de seguridad.

Características básicas

- Escrito por completo en Java.
- Completo sistema gestor de bases de datos relacional.
- Tiempo de arranque mínimo y gran velocidad en las operaciones: SELECT, INSERT, DELETE y UPDATE.
- Sintaxis SQL estándar.
- Integridad referencial (claves foráneas).
- Procedimientos almacenados en Java.
- Triggers.
- Tablas en disco de hasta 8GB.

1.5.9.2RazorSQL 4.5

RazorSQL es una herramienta de consulta de base de datos universal, programación y edición de SQL, navegador de base de datos, y herramienta de administración con capacidades de conexión

embebidas para DB2, Derby, Firebird, FrontBase, HSQLDB, Informix, Microsoft SQL Server, MySQL, OpenBase, Oracle, PostgreSQL, SQL Anywhere, SQLite, y Sybase. También soporta cualquier base de datos compatible con JDBC u ODBC (solo para Windows).

RazorSQL incluye un motor de base de datos relacional integrado totalmente listo para usar y que no requiere administración del usuario final. Algunas de las principales características de RazorSQL es que son herramientas visuales para crear, editar, describir, cambiar, eliminar, y visualizar objetos de la base de datos; herramientas para importar y exportar datos; un navegador de base de datos para visualizar objetos y estructuras de la base de datos; y un robusto editor de programación con soporte para SQL, PL/SQL, TransactSQL, SQL PL, PHP, Java, XML, HTML, y otros doce lenguajes de programación.

También incluye un constructor de consultas, herramientas para crear, editar, y ejecutar procedimientos almacenados, disparadores, y funciones, una herramienta para comparar tablas y otros resultados de consultas y el historial SQL. Y funcionalidad con el complemento API.

1.5.10 Metodologías y lenguajes para el desarrollo de software

El progreso de un software no es una tarea fácil. Como resultado a esta dificultad surgió desde hace mucho la Metodología. Las metodologías imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente.

Hoy en día existen numerosas propuestas metodológicas que inciden en distintas dimensiones del proceso de desarrollo. Un ejemplo de ellas son las propuestas tradicionales centradas específicamente en el control del proceso. Estas han demostrado ser efectivas y necesarias en un gran número de proyectos, sobre todo aquellos proyectos de gran tamaño respecto a tiempo y recursos.

Sin embargo, la experiencia ha demostrado que las metodologías tradicionales no ofrecen una buena solución para proyectos donde el entorno es volátil y donde los requisitos no se conocen con exactitud, porque no están pensadas para trabajar con incertidumbre.

Aplicar metodologías tradicionales obliga a forzar al cliente a que tome la mayoría de las decisiones al principio. Luego el costo de cambio de una decisión tomada puede llegar a ser muy elevado si se aplica metodologías tradicionales.

Por lo antes mencionado se han detectado varios problemas que a continuación se mencionan:

- Retrasos en la planificación: Llegada la fecha de entregar el software éste no está disponible.
- Sistemas deteriorados: El software se ha creado pero después de un par de años el costo de su mantenimiento es tan complicado que definitivamente se abandona su producción.
- Tasa de defectos: El software se pone en producción pero los defectos son tantos que nadie lo usa.
- Requisitos mal comprendidos: El software no resuelve los requisitos planificados inicialmente.
- Cambios de negocio: El problema que resolvía el software ha cambiado y aún se ha adaptado.
- Falsa riqueza: El software hace muchas cosas técnicamente muy interesantes y divertidas, pero no resuelven el problema del cliente, ni hace que éste gane más dinero.
- Cambios de personal: Después de unos años de trabajo los programadores comienzan a odiar el proyecto y lo abandonan.

Como respuesta a estos problemas surgieron otras metodologías que tratan de adaptarse a la realidad del desarrollo de software.

Surgen así las metodologías ágiles las cuales aportan nuevas técnicas y métodos de trabajo para el desarrollo de cada etapa de un software. En general estas metodologías hacen un balance entre los procesos y el esfuerzo, ya que tratan de centrarse en las cuestiones necesarias sin perderse en las burocráticas. Además, tienen como prioridad satisfacer al cliente mediante tempranas y continuas entregas de software que le aporte un valor, capturan los cambios para que el cliente tenga una ventaja competitiva, construir el proyecto en torno a individuos motivados y lograr que el personal del negocio y los desarrolladores trabajen juntos a lo largo del proyecto.

1.5.10.1 Proceso Unificado de Software (RUP)

El Proceso Unificado de Software (RUP) es una metodología de la ingeniería de software que va más allá de un sencillo análisis y diseño orientado a objetos para proporcionar una familia de técnicas que soportan el ciclo completo de desarrollo de software. RUP es un proceso basado en componentes, dirigido por los casos de uso, centrado en la arquitectura e iterativo e incremental. [10]

Divide en 4 fases el desarrollo del software:

- Inicio: El Objetivo en esta etapa es determinar la visión del proyecto.
- Elaboración: En esta etapa el objetivo es determinar la arquitectura sólida.
- Construcción: En esta etapa el objetivo es llevar a obtener la capacidad operacional inicial.
- Transición: El objetivo es llegar a obtener el realce del proyecto.

Cada una de estas etapas es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes.

RUP es una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software dado por la particularidad de que en cada ciclo de iteración, se hace exigente el uso de artefactos.

1.5.10.2 Lenguaje Unificado de Modelado (UML)

Lenguaje Unificado de Modelado es uno de los más conocidos y utilizados en la actualidad; aun cuando todavía no es un estándar oficial, está respaldado por el OMG (Object Management Group). Ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

Es importante resaltar que UML es un lenguaje para especificar, construir, visualizar y documentar los artefactos de un sistema de software Orientado a Objetos (OO) y no para describir métodos o procesos. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software, pero no especifica en sí mismo qué metodología o proceso usar. Es una técnica de modelado de objetos y como tal supone una abstracción de un sistema para llegar a construirlo en términos concretos. [11]

Algunas de las propiedades de UML como lenguaje de modelado estándar son: [12]

- Concurrencia, es un lenguaje distribuido y adecuado a las necesidades de conectividad actuales y futuras.
- Ampliamente utilizado por la industria desde su adopción por OMG.

- Reemplaza a decenas de notaciones empleadas con otros lenguajes.
- Modela estructuras complejas.
- La estructura más importante que soporta es que tiene su fundamento en las tecnologías orientadas a objetos, tales como objetos, clase, componentes y nodos.
- Emplea operaciones abstractas como guía para variaciones futuras, añadiendo variables si es necesario.
- Comportamiento del sistema: casos de uso, diagramas de secuencia y de colaboraciones, que sirven para evaluar el estado de las máquinas.

1.5.11 Herramientas a utilizar

1.5.11.1 Visual Paradigm 6.4

Los procesos de análisis y diseño del presente trabajo se desarrollaron en Visual Paradigm. Visual Paradigm para UML es una de las herramientas CASE, considerada muy completa y fácil de usar, con soporte multiplataforma y que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. Fue creada para el ciclo vital completo del desarrollo del software que lo automatiza y acelera, permitiendo la captura de requisitos, análisis, diseño e implementación. Es muy sencilla de usar, fácil de instalar y actualizar. Genera código para varios lenguajes. También proporciona características tales como generación del código, ingeniería inversa y generación de informes.

Tiene la capacidad de crear el esquema de clases a partir de una base de datos y crear la definición de base de datos a partir del esquema de clases. Permite invertir código fuente de programas, archivos ejecutables y binarios en modelos UML al instante, creando de manera simple toda la documentación. Cabe destacar igualmente su robustez, usabilidad y portabilidad. Está diseñada para usuarios interesados en sistemas de software de gran escala con el uso del acercamiento orientado a objeto. Incorpora el soporte para trabajo en equipo, esto permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios realizados por sus compañeros.

1.5.11.2 Netbeans IDE 6.7.1

El IDE NetBeans es un galardonado entorno de desarrollo integrado disponible para Windows, Mac, Linux y Solaris. NetBeans es un IDE de código abierto y una plataforma de aplicaciones que permiten a los desarrolladores crear rápidamente web, escritorio y aplicaciones móviles utilizando la

plataforma Java, así como JavaFX, PHP, JavaScript y Ajax, Ruby y Ruby on Rails, Groovy y Grails, y C / C + +. El proyecto NetBeans consiste en un IDE de código abierto y una plataforma de aplicaciones que permiten a los desarrolladores la creación rápida de Internet, empresas, equipos de escritorio y aplicaciones móviles utilizando la plataforma Java, así como JavaFX, PHP, JavaScript y Ajax, Ruby y Ruby on Rails, Groovy y Grails, y C / C + +.

El proyecto NetBeans es apoyada por una comunidad de desarrolladores vibrante y ofrece la documentación extensa y los recursos de formación, así como una variada selección de plugins de terceros.

NetBeans es una plataforma de ejecución de aplicaciones, es decir, facilita la escritura de aplicaciones Java, proporcionando una serie de servicios comunes, que a su vez están disponibles a través del IDE.

En este capítulo se realizó un estudio y análisis de los conceptos necesarios para el entendimiento de la creación del sistema desarrollado, centrando especial interés en las nuevas tendencias principalmente en aquellas vinculadas a la solución propuesta.

Se fundamentó la metodología de desarrollo del software, el lenguaje de programación y el sistema gestor de base de datos a utilizar, llegando a la conclusión de la factibilidad de empleo de estas tecnologías durante el ciclo de desarrollo del Sistema. Siempre siguiendo las políticas definidas por el Ministerio de Salud Pública (MINSAP) para la informatización del Sistema Nacional de Salud (SNS).

Capítulo 2. Características del Sistema

En este capítulo se describe la propuesta del sistema, se expone el modelo de dominio o Modelo Conceptual, el cual sirve de apoyo para la especificación de condiciones, capacidades y cualidades que el sistema debe cumplir. Se justifican los patrones de diseño a utilizar en el desarrollo de la aplicación, se presentan los requisitos de software tanto los funcionales como los no funcionales, y el modelamiento del sistema.

2.1 Modelo de Dominio

Al no identificarse negocio en el sistema de configuración, se emplea un modelo de dominio el cual permite captar los tipos más importantes de objetos que existen o los eventos que suceden en el entorno donde estará el sistema. En el modelo de dominio se describen los conceptos fundamentales, permitiendo mostrar al cliente el entorno de la información que se maneja y de esta manera contribuir a la comprensión del contexto del sistema. También se representa el diagrama del modelo de dominio, el cual permite de forma gráfica ver la relación entre los objetos o clases.

2.1.1 Conceptos Fundamentales

Área de Salud:

Estructura fundamental del sistema sanitario, responsable de la gestión unitaria de los centros y establecimientos del servicio de salud en su demarcación territorial y de las prestaciones sanitarias y programas sanitarios por ellos desarrollados.

Personal de Salud:

Personas que realizan las actividades y brindan los servicios como profesionales de salud en cada una de las unidades dentro de un área de salud.

Grupo Etáreo:

Persona o grupo de personas que tienen la misma edad.

Unidad de Salud:

Diferentes lugares donde se prestan servicios de salud. Estructuras físicas de las Zonas Básicas de Salud.

Estado del Paciente:

Distintos estados en que puede encontrarse un paciente según las definiciones de la Atención Primaria de Salud.

Ubicación Geográfica:

Es la localización precisa de un sitio con base en sus características geográficas, incluyendo: georeferencia (coordenadas) y esquemas del sitio con manzanas, calles, localidades, municipio, provincia y nación.

Clasificación Internacional de Atención Primaria:

Taxonomía de los términos y expresiones utilizadas habitualmente en medicina general. Recoge los motivos (o razones) de consulta, los problemas de salud y el proceso de atención. Es un tipo de clasificación de terminología médica de ámbito internacional.

Para más información sobre los principales conceptos asociados y la descripción de estos, en cada uno de los modelos de dominio de los Nomencladores Asistenciales, Ubicación Geográfica, Grupo Etáreo, Estado del Paciente, Unidades de Salud, Personal de Salud y Seguridad del Módulo de Configuración para Escritorio remitirse al Expediente de Proyecto. [13]

2.2 Conceptos fundamentales de las Áreas de Salud

Centro Laboral: Hace referencia a los centros laborales que se encuentran dentro de un área de salud.

Dirección: Hace referencia a la ubicación lógica (calles, entre calles, números, etc.) de las viviendas y centros laborales de un área de salud.

Zona_ APS: Hace referencia a zonas básicas (territorio de influencia de un centro de salud) de un área de salud.

Localidades_ Áreas: Hace referencia a las localidades atendidas por un área de salud.

Tipo_ Centro: Hace referencia a los tipos de centros laborales que pueden existir, ejemplo: estudiantil, laboral, entre otros.

Tipo_ Zona_ APS: Hace referencia a los tipos de zonas básicas de un área de salud en dependencia de si comprenden viviendas o centros laborales, ejemplo: comunidad, centro laboral, entre otros.

Ubicación_ Zona_ APS: Hace referencia a la ubicación geográfica de las zonas básicas de un área de salud, ejemplo: rural, urbana, entre otros.

Unidad_ Base: Hace referencia a las unidades base (hospital u otra institución superior) de un área de salud en caso de que la tenga, ejemplo: Hospital Gineco Obstétrico González Coro, entre otros.

2.2.1 Diagrama del modelo de dominio de Área de Salud

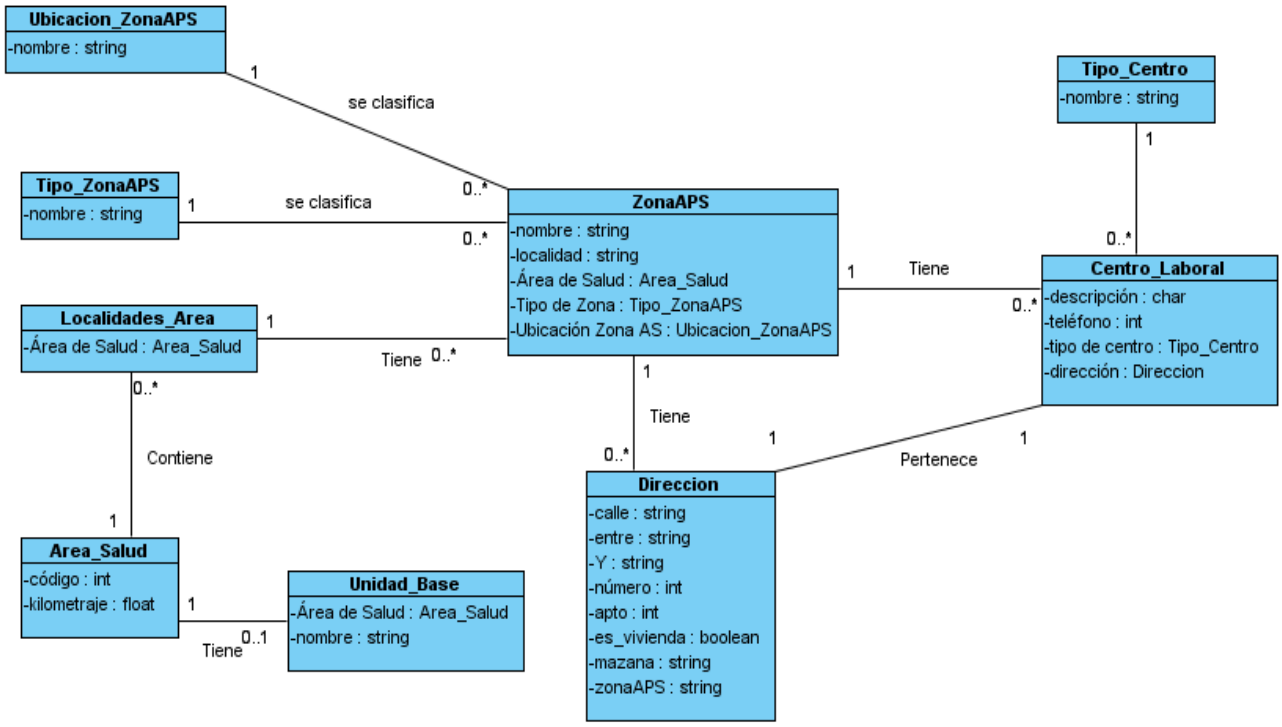

Figura # 1 Modelo de Dominio Área de Salud

Para obtener más información sobre los Modelos de Dominio de los Nomencladores Asistenciales, Ubicación Geográfica, Grupo Etáreo, Estado del Paciente, Unidades de Salud, Personal de Salud y Seguridad del Módulo de Configuración para Escritorio, remitirse al Expediente de Proyecto.[14]

2.3 Especificación de Requerimientos de Software

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA

Un Requerimiento de Software es la condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente. Los requerimientos de Software se dividen en requerimientos funcionales y requerimientos no funcionales.

2.3.1 Requerimientos Funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir.

No	Requisitos Funcionales	Descripción
RF1	AS_ Configurar Área Salud.	Registrar y modificar los datos del codificador área de salud.
RF1.1	AS_ Agregar Datos Área de Salud.	Registrar los datos de un área de salud determinada.
RF1.2	AS_ Modificar Datos Área de Salud.	Modificar los datos de un área de salud determinada.
RF2	AS_ Gestionar Datos Dirección.	Registrar, obtener, eliminar y modificar los datos del codificador datos dirección.
RF2.1	AS_ Agregar Datos Dirección.	Registrar los datos de la dirección.
RF2.2	AS_ Buscar Datos Dirección.	Obtener los datos de una o todas las direcciones en dependencia de la opción de búsqueda seleccionada.
RF2.3	AS_ Eliminar Datos Dirección.	Eliminar los datos de una o todas las direcciones.
RF2.4	AS_ Modificar Datos Calle.	Modificar los datos de una dirección.
RF2.5	AS_ Mostrar Datos Calle.	Mostrar los datos de una dirección.

RF3	AS_ Gestionar Zona APS.	Registrar, obtener, eliminar y modificar los datos del codificador zona APS.
RF3.1	AS_ Agregar Zona APS.	Registrar los datos de la zona APS.
RF3.2	AS_ Buscar Zona APS.	Obtener los datos de una o todas la zona APS en dependencia de la opción de búsqueda seleccionada.
RF3.3	AS_ Eliminar Zona APS	Eliminar los datos de una o todas las zonas APS.
RF3.4	AS_ Modificar Zona APS.	Modificar los datos de una zona APS.
RF4	AS_ Gestionar Centro Laboral.	Registrar, obtener, eliminar y modificar los datos del codificador centro laboral.
RF4.1	AS_ Agregar Centro Laboral.	Registrar los datos del centro laboral.
RF4.2	AS_ Buscar Centro Laboral.	Obtener los datos de una o todos los centros laborales en dependencia de la opción de búsqueda seleccionada.
RF4.3	AS_ Eliminar Centro Laboral.	Eliminar los datos de uno o todos los centros laborales.
RF4.4	AS_ Modificar Centro Laboral.	Modificar los datos del centro laboral.
RF4.5	AS_ Mostrar Centro Laboral.	Mostrar los datos del centro laboral.
RF5	AS_ Gestionar Codificador Tipo Zona.	Registrar, obtener, eliminar y modificar los datos del codificador tipo de zona.
RF5.1	AS_ Agregar Datos Tipo Zona.	Registrar los datos del tipo de zona.

RF5.2	AS_ Buscar Datos Tipo Zona.	Obtener los datos de uno o todos los tipos de zona en dependencia de la opción de búsqueda seleccionada.
RF5.3	AS_ Eliminar Datos Tipo Zona.	Eliminar los datos de uno o todos los tipos de zonas.
RF5.4	AS_ Modificar Datos Tipo Zona.	Modificar los datos del tipo de zona.
RF6	AS_ Gestionar Datos Ubicación Zona.	Registrar, obtener, eliminar y modificar los datos del codificador ubicación geográfica.
RF6.1	AS_ Agregar Datos Ubicación Zona.	Registrar los datos de la ubicación zona.
RF6.2	AS_ Buscar Datos Ubicación Zona.	Obtener los datos de una o todas las ubicaciones de zonas en dependencia de la opción de búsqueda seleccionada.
RF6.3	AS_ Eliminar Datos Ubicación Zona.	Eliminar los datos de una o todas las ubicaciones de zonas.
RF6.4	AS_ Modificar Datos Ubicación Zona.	Modificar los datos de la ubicación zona.
RF7	AS_ Gestionar Codificador Tipo de Centro.	Registrar, obtener, eliminar y modificar los datos del codificador tipo de centro.
RF7.1	AS_ Agregar Datos Tipo de Centro.	Registrar los datos del tipo de centro.
RF7.2	AS_ Buscar Datos Tipo de Centro.	Obtener los datos de uno o todos los tipos de centros en dependencia de la opción de búsqueda seleccionada.
RF7.3	AS_ Eliminar Datos Tipo de Centro.	Eliminar los datos de uno o todos los tipos de centro.

RF7.4	AS_ Modificar Datos Tipo de Centro.	Modificar los datos del tipo de centro.
RF8	UG_ Gestionar Datos Nación.	Registrar, obtener, eliminar y modificar los datos de la nación.
RF9	UG_ Gestionar Datos Provincia.	Registrar, obtener, eliminar y modificar los datos de la provincia.
RF10	UG_ Gestionar Datos Municipio.	Registrar, obtener, eliminar y modificar los datos del municipio.
RF11	UG_ Gestionar Datos Localidad.	Registrar, obtener, eliminar y modificar los datos de la localidad.
RF12	UG_ Gestionar Datos Calle.	Registrar, obtener, eliminar y modificar los datos de la calle.
RF13	UG_ Gestionar Datos Manzana.	Registrar, obtener, eliminar y modificar los datos de la manzana.
RF14	PS_ Gestionar Datos Personal Salud.	Registrar, obtener, eliminar y modificar los datos del personal de salud.
RF15	PS_ Gestionar Codificador Tipo Profesional.	Registrar, obtener, eliminar y modificar los datos del codificador tipo profesional.
RF16	PS_ Gestionar Codificador Nivel Técnico.	Registrar, obtener, eliminar y modificar los datos del codificador nivel técnico.
RF17	PS_ Gestionar Codificador Función Médica.	Registrar, obtener, eliminar y modificar los datos del codificador ubicación.
RF18	PS_ Gestionar Codificador Ubicación.	Registrar, obtener, eliminar y modificar los datos del codificador ubicación.

RF19	PS_ Gestionar Codificador Estado Laboral.	Registrar, obtener, eliminar y modificar los datos del codificador estado laboral.
RF20	US_ Gestionar Datos Unidad de Salud.	Registrar, obtener, eliminar y modificar los datos de las unidades de salud.
RF21	US_ Gestionar Codificador Organismos.	Registrar, obtener, eliminar y modificar los datos del codificador organismos.
RF22	US_ Gestionar Codificador Grupos.	Registrar, obtener, eliminar y modificar los datos del codificador grupos.
RF23	US_ Gestionar Codificador Subgrupo.	Registrar, obtener, eliminar y modificar los datos del codificador subgrupos.
RF24	US_ Gestionar Codificador Tipo de Unidad.	Registrar, obtener, eliminar y modificar los datos del codificador tipo de unidad.
RF25	US_ Gestionar Codificador Tipo de Departamento.	Registrar, obtener, eliminar y modificar los datos del codificador tipo de departamento.
RF26	US_ Gestionar Codificador Departamento.	Registrar, obtener, eliminar y modificar los datos del codificador departamento.
RF27	US_ Gestionar Codificador Servicios.	Registrar, obtener, eliminar y modificar los datos del codificador servicios.
RF28	US_ Gestionar Codificador Especialidades.	Registrar, obtener, eliminar y modificar los datos del codificador especialidades.
RF29	GE_ Gestionar Grupo Etáreo.	Registrar, obtener, eliminar y modificar los datos de grupo etáreo.

RF30	EP_ Gestionar Estado del Paciente.	Registrar, obtener, eliminar y modificar los datos del estado del paciente.
RF31	CPAPS_ Gestionar Clasificación del Grupo de Problemas de Salud.	Registrar, obtener, eliminar y modificar los datos de los problemas de salud.
RF32	EDO_ Gestionar Datos Enfermedad.	Registrar, obtener, eliminar y modificar los datos de la enfermedad.
RF33	EDO_ Gestionar Datos Grupo.	Registrar, obtener, eliminar y modificar los datos grupo.
RF34	CIAP_ Gestionar Datos Problemas de Salud.	Registrar, obtener, eliminar y modificar los datos problemas de salud.
RF35	CIAP_ Gestionar Datos Capítulos.	Registrar, obtener, eliminar y modificar los datos capítulo.
RF36	CIAP_ Gestionar Datos Componente.	Registrar, obtener, eliminar y modificar los datos componente.
RF37	CIAP_ Gestionar Datos Subcomponente.	Registrar, obtener, eliminar y modificar los datos subcomponente.
RF38	CIAP_ Gestionar Datos Abreviatura	Registrar, obtener, eliminar y modificar los datos abreviatura.
RF39	CIE_ Gestionar Datos Capítulos.	Registrar, obtener, eliminar y modificar los datos del capítulo.
RF40	CIE_ Gestionar Datos Grupos.	Registrar, obtener, eliminar y modificar los datos del grupo.
RF41	CIE_ Gestionar Datos Categoría.	Registrar, obtener, eliminar y modificar los

		datos de la categoría.
RF42	CIE_ Gestionar Datos Subcategoría.	Registrar, obtener, eliminar y modificar los datos de la subcategoría.
RF43	CIE_ Gestionar Datos Problemas de Salud.	Registrar, obtener, eliminar y modificar los datos de los problemas de salud.
RF44	Seguridad_ Gestionar Roles.	Registrar, obtener, eliminar y modificar los roles de los usuarios.
RF45	Seguridad_ Gestionar Funcionalidad.	Registrar, obtener, eliminar y modificar las funcionalidades de los usuarios.
RF46	Seguridad_ Gestionar Codificador Perfil de Usuario.	Registrar, obtener, eliminar y modificar los perfiles de usuario.
RF47	Seguridad_ Gestionar Usuario	Registrar, obtener, eliminar y modificar usuario.

2.3.2 Requerimientos no Funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable.

2.3.2.1 Usabilidad

El sistema estará diseñado de manera que los usuarios adquieran las habilidades necesarias para explotarlo en un tiempo reducido. Cuenta con una apariencia similar a las aplicaciones de Windows para facilitar la adaptación del personal que lo utilice.

2.3.2.2 Eficiencia

El sistema minimizará el volumen de datos en las peticiones y además optimizará el uso de recursos críticos como la memoria. Para ello se potenciará como regla guardar en la memoria caché, datos y recursos de alta demanda, o sea, funcionalidades de más uso por el usuario.

El sistema respetará buenas prácticas de programación para incrementar el rendimiento en operaciones costosas para la máquina virtual como la creación de objetos. Se deberá usar siempre que sea posible el patrón Singleton, destruir referencias que ya no estén siendo usadas, optimizar el trabajo con cadenas, entre otras buenas prácticas que ayudan a mejorar el rendimiento.

2.3.2.3 Restricciones de diseño

La capa de presentación contendrá todas las vistas y la lógica de la presentación. El flujo a través de las ventanas y la interacción de los componentes se manejarán mediante las plataformas Swing Application Framework y BeansBindings. La capa del negocio manejará el negocio de manera simple a los desarrolladores apoyándose en la jerarquía de clases que proporcionará las funcionalidades de impresión e interacciones CRUD con la BD. La capa de acceso a datos contendrá las entidades y los objetos de acceso a datos correspondientes a las mismas. El acceso a datos está basado en el estándar JPA y particularmente en la implementación del motor de persistencia Hibernate.

2.3.2.4 Interfaz

Las ventanas del sistema contendrán los datos claros y bien estructurados, además de permitir la interpretación correcta de la información. La interfaz contará con teclas de función que faciliten y aceleren su utilización y contará con una barra de menú desplegable para facilitar el acceso a los diferentes módulos, cambiar el idioma, apariencia y contenido, así como para acceder a la ayuda. El sistema tendrá además un menú lateral izquierdo para realizar la navegación de manera directa y sencilla entre las funcionalidades del módulo. La entrada de datos incorrecta será detectada claramente e informada al usuario. Todos los textos y mensajes en pantalla aparecerán en idioma español.

2.3.2.5 Requerimientos de soporte

Se permitirá la creación de usuarios, otorgamiento de privilegios y roles, asignación de perfiles y activación de permisos.

- **Seguridad de acceso y administración de usuarios.**

Se mantendrá un nivel de seguridad entre las estaciones de trabajo, garantizando únicamente la ejecución de las aplicaciones que hayan sido definidas para la estación en cuestión.

- **Respaldo y recuperación de base de datos.**

Se permitirá realizar copias duras de seguridad de la base de datos hacia otro dispositivo de almacenamiento externo, además de recuperar la base de datos a partir de los respaldos realizados mediante scripts comprimidos.

- **Configuración de parámetros.**

Se permitirá configurar la aplicación así como el funcionamiento de sus módulos.

Se permitirá establecer parámetros de diseño como títulos, ventanas de diálogo y Splash.

Se permitirá establecer parámetros de configuración del sistema y actualización de nomencladores.

2.3.2.6 Requerimientos de hardware

Para las consultas del Sistema para la Atención Primaria alas SIAPS, se escogieron estaciones de trabajo de 256 Mb de memoria RAM y un microprocesador de 2.0 GHz.

2.3.2.7 Requerimientos de software

El sistema debe correr en sistemas operativos Windows o Unix, utilizando la plataforma JAVA (Java Runtime Enviroment y HSQLDB Hypersonic).

En este capítulo después de haber analizado el dominio del problema, se identificaron los principales conceptos asociados tales como Área de Salud, Personal de Salud, Grupo Etáreo, Unidades de Salud, Estado del Paciente, Ubicación Geográfica, los Nomencladores Asistenciales y los elementos necesarios para conformar los requerimientos funcionales y no funcionales, imprescindibles para la realización del análisis y el diseño del módulo que se analizará en el capítulo siguiente.

Capítulo 3. Diseño del Sistema

El Diseño de un sistema de software da una mayor comprensión de los aspectos relacionados con los requerimientos no funcionales y contribuye a la definición de una arquitectura estable, sólida, creando básicamente, un plano del modelo de implementación.

En este capítulo se justifican los patrones de diseño a utilizar en el desarrollo de la aplicación y los diagramas de clases del diseño, el diagrama de clases persistentes, el modelo de datos, además de la explicación de sus tablas y sus atributos. Como resultado del flujo de trabajo Diseño se generan los artefactos mencionados anteriormente, que son el punto de partida para la implementación del sistema.

3.1 Modelo de Diseño

El modelo de diseño permite 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 y concurrencia y tecnologías de interfaz de usuario. Crea una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.

Descompone los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo. Captura las interfaces entre los subsistemas en el ciclo de vida del software, lo cual es muy útil cuando se utilizan interfaces como elementos de sincronización entre diferentes equipos de desarrollo. Los artefactos generados en el modelo de diseño son: Modelo de Despliegue, Descripción de la Arquitectura, Clase del Diseño, Subsistema de Diseño e Interfaz.

3.1.1 Patrones de Diseño

En la estructura de los Diagramas de Clases del Diseño se muestra la aplicación de los patrones de diseño, así como las limitaciones que establecen sobre la arquitectura definida. Estos diagramas están estructurados siguiendo el patrón MVC, el cual permite la separación en capas de los objetos que componen el diseño del sistema, e introduce la necesidad de crear tres categorías de objetos principales: vistas, controladores y modelos.

La capa de persistencia o datos en el patrón MVC también forma parte del modelo. En esta capa se utilizan los patrones de diseño Abstract Factory, Singleton y Lazy.

Otros de los patrones de diseño que se utilizan son los patrones GRASP, los cuales tienen como objetivo la descripción de los principios fundamentales de diseño de objetos para la asignación de responsabilidades y dentro de estos, los patrones Experto, Creador, Alta cohesión, Bajo acoplamiento y Controlador.

Mediante la asignación a cada clase de las tareas o responsabilidades que estas pueden realizar, en dependencia de la información que contienen, se evidencia el uso de los patrones Experto y Creador. Estos conservan el encapsulamiento y definen quién será el responsable de crear una instancia de una clase respectivamente. Al utilizar los patrones Alta cohesión y Bajo acoplamiento se permite la colaboración entre clases o elementos del diseño sin que se afecte su reutilización y entendimiento cuando se encuentren aislados. El patrón Controlador sirve como intermediario entre una determinada interfaz y el algoritmo que lo implementa, de tal forma que es el que recibe los datos del usuario y los envía a las distintas clases según el método llamado.

3.1.2 Estructura de Diseño

En el Modelo de diseño se establece la realización de las funcionalidades en clases incluyendo una orientación hacia el entorno de implementación. Está constituido por los diagramas de clases. Se realizará un diagrama de clases por cada funcionalidad simple y compleja en el caso de los codificadores.

Para la nomenclatura de estas clases en los Diagramas de Clases se siguió la siguiente estructura: los diagramas de clases del diseño se nombrarán: DCD_<Nombre de la funcionalidad>, para la clase interfaz se define SiapsFRGestionar(opción).java, para la clase control se define CC(opción).java y para las entidades (Tb,Tr,Tn)(opción).java.

En los diagramas aparecen las clases entidades y sus relaciones con las clases controladoras. La clase Entity Manager representa la clase con la que se relacionan las controladoras para llevar a cabo las operaciones de persistencia de datos.

Se definieron por la utilización del Framework Swing las clases, Swing Component, Swing Contenedores y Swing App Framework, así como los paquetes Framework JPA, Framework Hibernate y Framework Jasper Report.

3.1.3 Diagramas de Clases del Diseño

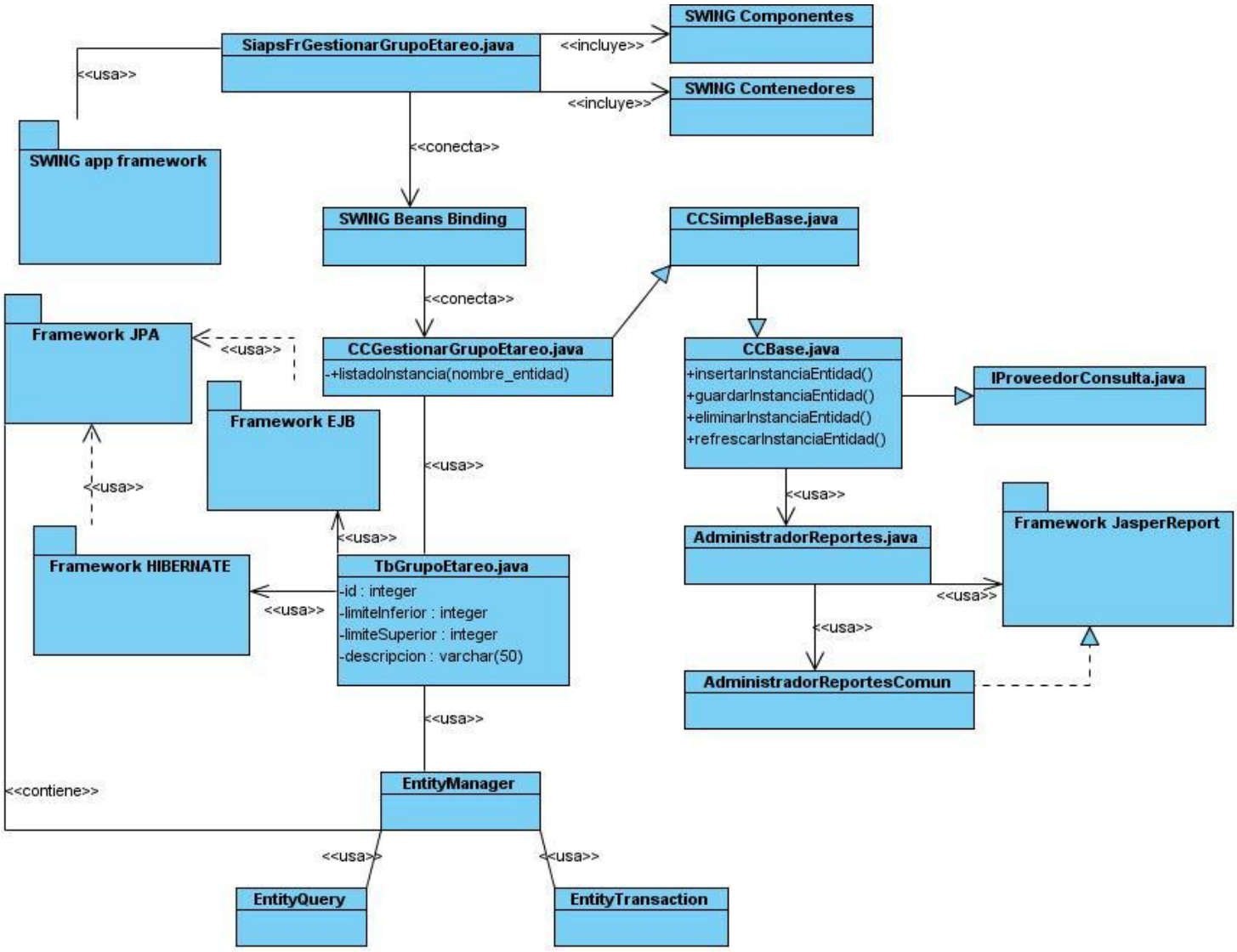

Figura # 2. DCD Gestionar Grupo Etáreo.

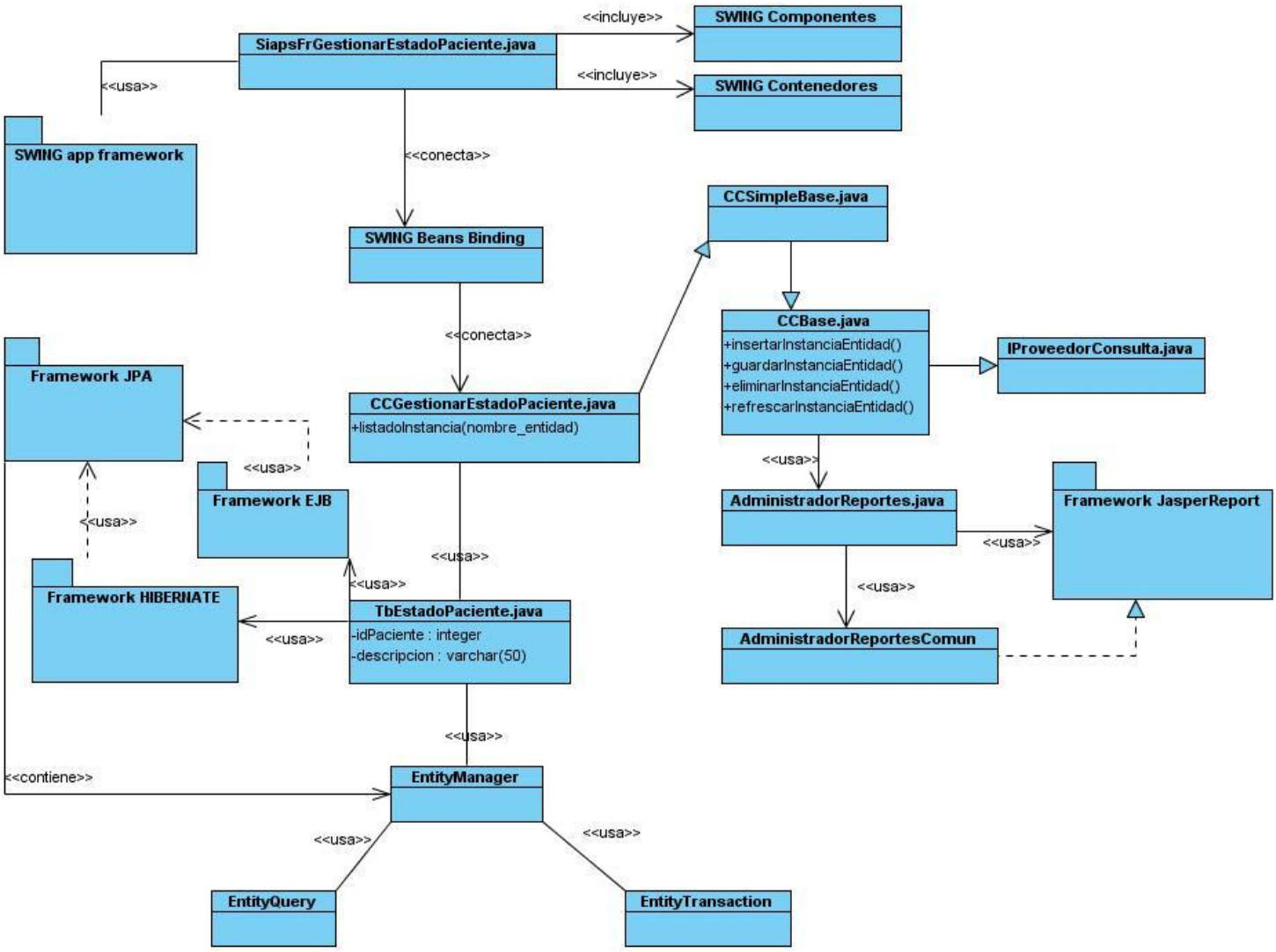

Figura # 3. DCD Gestionar Estado del Paciente.

3.1.4 Descripción de clases y atributos

Posteriormente serán explicadas algunas de las clases que han sido identificadas para su futura implementación. De esta manera, se tendrá una comprensión mayor del funcionamiento que tendrá el sistema en desarrollo.

La clase **SiapsFrGestionarGrupoEtareo.java** es un formulario desktop que permite capturar los datos que serán persistidos en la base de datos, además de mostrar información útil al usuario.

La clase **CCGestionarGrupoEtareo.java** es una clase que permite darle respuesta a las peticiones que se desencadenan en la vista a través de los métodos que contienen. Hace uso del Framework EJB que encapsula la lógica de negocio, integrándose con la vista a través del Framework SWING.

La clase **TbGrupoEtareo.java** es una clase que se ejecuta del lado del servidor. Representa una tabla en el modelo de datos relacional y cada instancia de esta entidad corresponde a un registro en esa tabla. Es persistida por las clases servidoras para darle una respuesta a las páginas clientes. Hace uso del Framework Hibernate y JPA.

Clase **Swing Componentes**: conjunto de componentes visuales (jtextbox, jlabel, jradiobutton, etc.) que permiten la construcción de la vista. Forman parte del Framework SWING.

Clase **Swing Contenedores**: conjunto de componentes visuales (jpanel, jframe, etc.) que permiten agrupar los Swing Componentes para la construcción de la vista. Forman parte del Framework SWING.

Clase **SWING Beans Binding**: librería que permite el enlace dinámico entre los componentes visuales y los datos. Forman parte del Framework SWING.

Paquete SWING App Framework: es una especificación que provee la implementación prototipo de un conjunto de clases Java que permite tener un sistema de ventanas, componentes gráficos, independiente del sistema operativo y la librería de dibujo que estén disponibles en la máquina destino.

Clase **CComplejoBase.java:** es la clase abstracta que encapsula las principales funcionalidades de la cuál heredan todas las clases controladoras para los CRUD complejos. (Cuando hay más de una entidad relacionada en el manejo de los datos o la funcionalidad).

Clase **CSimpleBase.java:** es la clase abstracta que encapsula las principales funcionalidades de la cual heredan todas las clases controladoras para el CRUD simple. (Cuando hay solo una entidad relacionada en el manejo de los datos o la funcionalidad).

Clase **CBase.java**: es la clase abstracta base de la jerarquía de la cual heredan las clases CCComplejoBase y CCSimpleBase. Implementa la interfaz IProveedorConsulta.

Interfaz IProveedorConsulta.java: es una interfaz que define la forma en que se van a realizar las consultas a la BD.

Clase **AdministradorReportes.java**: es la clase que permite construir los reportes usando la librería JasperReport y usa la clase AdministradorReportesComun.

Clase **AdministradorReportesComun**: es la clase que permite navegar por los datos a mostrar en el reporte y acceder a ellos a través de sus nombres. Hereda funcionalidades del framework JasperReport.

Framework JasperReport: es una librería de clases de Java de código abierto desarrollada para facilitar el agregar capacidades de reporte a las aplicaciones Java. Permite realizar reportes de código abierto que tiene como función el llevar documentos ricos en contenido a la pantalla, a la impresora, o a archivos PDF, HTML, XLS, CSV y XML.

Paquete Framework EJB: es una plataforma para construir aplicaciones de negocio portables, escalables, y reutilizables utilizando el lenguaje de programación Java. El objetivo de Enterprise JavaBeans (EJB) 3.0 es simplificar el desarrollo de aplicaciones Java y estandarizar el API de persistencia para la plataforma Java. Forma parte de la especificación JEE 5.

Clase **EntityManager:** permite realizar las operaciones CRUD (Crear, Leer, Actualizar y Eliminar) que impliquen entidades.

Clase **EntityQuery:** Permite encontrar objetos persistentes manejando cierto criterio de búsqueda. Permite realizar peticiones a la base de datos y controla cómo se ejecuta dicha petición. Se utiliza para enlazar los parámetros de la petición, limitar el número de resultados devueltos por la petición y para ejecutar dicha petición.

Clase **EntityTransaction:** permite realizar operaciones sobre datos persistentes de manera que agrupados formen una unidad de trabajo transaccional, en el que todo el grupo sincroniza su estado de persistencia en la base de datos o todos fallan en el intento, en caso de fallo, la base de datos quedará con su estado original. Maneja el concepto de todos o ninguno para mantener la integridad de los datos.

Paquete Framework HIBERNATE: Conjunto de clases agrupadas en componentes que constituyen una herramienta de Mapeo objeto/relacional ó ORM de código abierto (Object Relational Mapping) y un generador de sentencias SQL. Permite diseñar objetos persistentes que podrán incluir polimorfismo, relaciones, colecciones, y un gran número de tipos de datos. De una manera muy rápida y optimizada permite generar BBDD en cualquiera de los entornos soportados: Oracle, PostgresSQL, DB2, MySql, entre otras.

Paquete Framework JPA: Conjunto de clase agrupadas en componentes que constituyen la API de persistencia desarrollada para la plataforma Java EE e incluida en el estándar EJB 3.0 como parte de JSR 220, aunque su uso no se limita a los componentes software EJB. Permite unificar la manera en que funcionan las utilidades que proveen un mapeo objeto-relacional. El objetivo que persigue el diseño de esta API es no perder las ventajas de la orientación a objetos al interactuar con una base de datos.

Para obtener más información sobre los Diagramas del Diseño y la descripción de las clases de los Nomencladores Asistenciales, Área de Salud, Ubicación Geográfica, Estado del Paciente, Unidades de Salud, Personal de Salud y Seguridad del Módulo de Configuración para Escritorio, remitirse al Expediente de Proyecto.[15]

En este capítulo se analizaron todos los diagramas de clases para cada funcionalidad del sistema a implementar, como ejemplo se tiene al Diagrama de Clases del Diseño para el Gestionar Grupo Etáreo. Se analizaron los patrones de diseño utilizados en el sistema como Abstract Factory, Singleton y Lazy. Lo que permite crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases, los cuales serán analizados en el posterior capítulo.

Capítulo 4. Implementación y Prueba

En este capítulo se pretende llevar a cabo la implementación del sistema. Se expone la necesidad de la Integración del módulo Configuración con otros componentes del SISalud. Se muestran artefactos como el Diagrama de Despliegue y el Diagrama de Componentes. Se describen detalladamente los estándares de Diseño, Codificación y Tratamientos de Errores.

4.1 Propuesta de integración entre módulos

Módulo de Medicina Familiar

Este módulo utiliza los Nomencladores Asistenciales y lo referente al Área de Salud.

Módulo de Enfermería

Este módulo utiliza los departamentos ubicados dentro de la Entidad.

Módulo Clínico Quirúrgico

Este módulo utiliza las especialidades y lo correspondiente a los médicos, ubicados en la Entidad además de los Nomencladores Asistenciales.

Módulo de Medios de Diagnostico

Este módulo utiliza los departamentos y servicios localizados dentro de la Entidad.

4.2 Propuesta de seguridad del módulo

Para el correcto funcionamiento del Módulo Configuración para escritorio, se tuvieron en cuenta una serie de requisitos de seguridad con el objetivo que no se pueda afectar el funcionamiento del sistema.

El sistema contará con un componente de seguridad encargado de la autenticación, auditoría y control de usuarios. Habrá una relación entre las tablas usuarios_roles y funcionalidad_roles permitiendo el acceso a la aplicación por tipo de usuario y dándole solo visibilidad a las áreas establecidas en dependencia del rol que posea.

Las bitácoras le permitirán al sistema llevar una traza de todas las operaciones llevadas a cabo por cada usuario mediante un registro de actividades con el objetivo de realizar reportes de todos los cambios realizados.

4.3 Implementación

En el flujo de trabajo de Implementación se comienza a partir del resultado obtenido en el diseño y se implementa el sistema en términos de componentes, es decir, ejecutables, ficheros, códigos de fuente y similares, necesarios para la implantación y despliegue del sistema. El objetivo fundamental es desarrollar la arquitectura y el sistema como un todo.

4.3.1 Diagrama de Despliegue

Un Diagrama de Despliegue es un grafo de nodos unidos por conexiones de comunicación. Un nodo es un recurso de ejecución tal como un computador, un dispositivo o memoria, muestra las relaciones físicas entre los componentes hardware y software en el sistema final, es decir, la configuración de los elementos de procesamiento en tiempo de ejecución y los componentes software (procesos y objetos que se ejecutan en ellos).

A continuación se presenta el Diagrama de Despliegue correspondiente al Módulo de Configuración:

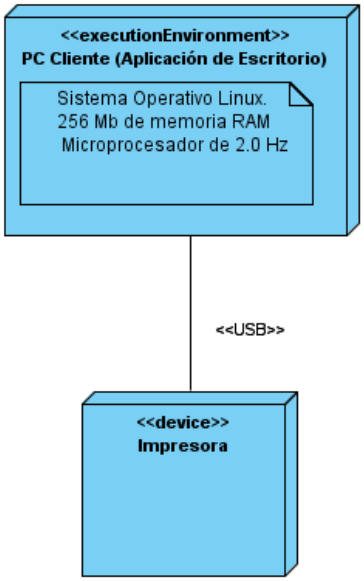

Figura # 4 Diagrama de Despliegue en modo desconectado.

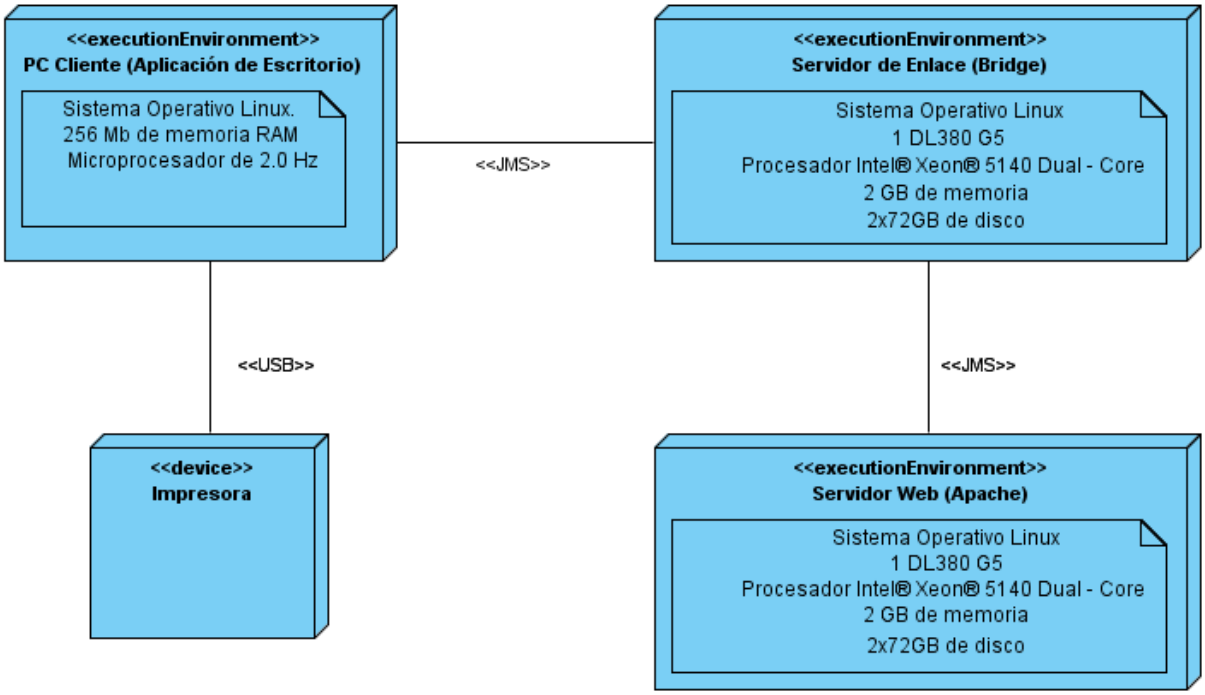

Figura # 5 Diagrama de Despliegue una vez establecida la conexión.

4.3.2 Estándares de diseño, codificación y tratamiento de excepciones

Para que exista igualdad entre los módulos de SIAPS, se establecieron elementos similares tanto para el diseño como para la codificación y tratamiento de errores en los códigos fuentes de las aplicaciones y los mensajes que se emitan, facilitando el trabajo con ellos.

4.3.2.1 Estándares de diseño

En el Modelo de diseño se establece la realización de las funcionalidades en clases incluyendo una orientación hacia el entorno de implementación. El mismo está constituido por los diagramas de clases. Se realizará un diagrama de clases por cada funcionalidad.

La estructura del diagrama de clases responderá a la arquitectura definida, la cual corresponde al patrón Modelo Vista Controlador implícito en los Frameworks que se utilizan y el patrón 3 capas para dividir las capas de presentación, negocio y persistencia.

Nomenclatura

Los diagramas de clases del diseño: DCD_<Nombre de la funcionalidad>.

Nomenclatura de los métodos

La nomenclatura de los métodos listar incluidos en las clases controladoras CC<nombre de la opción> para los CRUD Simple y CRUD Complejo será la siguiente:

- listar<nombre de la entidad>()

En el caso de los métodos para el resto de las funcionalidades son heredados de la clase CCBase.java y la nomenclatura es la siguiente:

- insertarInstanciaEntidad()
- guardarInstanciaEntidad()
- eliminarInstanciaEntidad()
- refrescarInstanciaEntidad()

En el nombre de los métodos la primera palabra del nombre debe escribirse con letra minúscula y la primera letra del resto de las palabras con letra mayúscula.

4.3.2.2 Estándares de codificación

Actualmente se hallan estándares de codificación para la mayoría de los lenguajes existentes. El uso de ellos partiendo de las convenciones definidas permite una mejor comunicación entre los programadores creando las condiciones para la reusabilidad y el mantenimiento de los sistemas. A continuación se describen algunos de los elementos que demuestran lo antes enunciado:

El idioma utilizado es el español.

La indentación tiene como objetivo lograr una estructura uniforme para los bloques de código así como para los diferentes niveles de anidamiento, se debe emplear cuatro espacios como unidad de indentación y los tabuladores deben ser exactamente cada 8 espacios.

- Longitud de la línea deben ser inferiores a los 80 caracteres.
- Rompiendo Líneas cuando una expresión no entre en una línea, se debe romper después de una coma o antes de un operador, se debe alinear la nueva línea con el comienzo de la expresión al mismo nivel de la línea anterior.

Además, se tuvieron en cuenta comentarios, separadores, líneas, espacios en blanco y márgenes para establecer un modo común al comentar el código, de manera que sea comprensible con sólo leerlo una vez.

Comentarios:

Los comentarios no deben encerrarse en grandes cuadrados dibujados con asteriscos u otros caracteres.

Los comentarios nunca deben incluir caracteres especiales como backspace.

Los comentarios de bloque se podrán usar al comienzo de cada fichero o antes de cada método.

Un comentario de una sola línea debe ir precedido de una línea en blanco.

Los comentarios de fin de línea // pueden convertir en comentario una línea completa o una parte de una línea.

Cada comentario de documentación se encierra con los delimitadores de comentarios /**...*/.

Declaraciones:

Se recomienda una declaración por línea, ya que facilita los comentarios, no poner diferentes tipos en la misma línea. Intentar inicializar las variables locales donde se declaran y situar las declaraciones solo al principio de los bloques.

Declaraciones de Clases e Interfaces:

Ningún espacio en blanco entre el nombre de un método y el paréntesis "(" que abre su lista de parámetros. La llave de apertura "{" aparece al final de la misma línea de la sentencia declaración.

La llave de cierre "}" empieza una nueva línea para ajustarse a su sentencia de apertura correspondiente, excepto cuando no existen sentencias entre ambas, que debe aparecer inmediatamente después de la de apertura "{". Los métodos se separan con una línea en blanco.

Sentencias:

Cada línea debe contener como mucho una sentencia. Una sentencia return con un valor no debe usar paréntesis a menos que hagan el valor de retorno más obvio de alguna manera.

Las sentencias if usan siempre llaves {}.

Inicio y fin de Bloque {} Lo mismo sucede para el caso de las instrucciones if, else, for, while, do while, switch, try, catch, etc.

Líneas en Blanco.

Se deben usar siempre dos líneas en blanco entre las secciones de un fichero fuente y las definiciones de clases e interfaces.

Se debe usar siempre una línea en blanco entre métodos**,** variables locales de un método y su primera sentencia, antes de un comentario de bloque o de un comentario de una línea y entre las distintas secciones lógicas de un método para facilitar la lectura.

Espacios en Blanco.

Todos los operadores binarios excepto "." se deben separar de sus operandos con espacios en blanco. Las expresiones en una sentencia for se deben separar con espacios en blanco. Los "Cast"s deben ir seguidos de un espacio en blanco.

Convenciones de Nombres

El prefijo del nombre de un paquete se escribe siempre con letras en minúsculas, y debe ser uno de los nombres de dominio de alto nivel.

Los nombres de las clases e interfaces deben ser sustantivos, cuando son compuestos tendrán la primera letra de cada palabra que lo forma en mayúsculas.

Excepto las constantes, todas las instancias y variables de clase o método empezarán con minúscula.

4.3.2.3Tratamiento de excepciones

Para prever que no existan errores en el funcionamiento del sistema es indispensable el tratamiento de errores, por tanto, desde el inicio se realizan operaciones y se cumplen tareas para evitar la ocurrencia de estos. Durante la entrada de datos en los formularios, los componentes tendrán en cuenta la cantidad de letras permisibles, ejemplo de esto en el Carnet de Identidad de una persona, que este no puede exceder de 11 caracteres. Se validará el tipo de carácter, para este mismo ejemplo solo permite números. En caso de que no se permita la duplicación de los datos, mostrará un mensaje al usuario, en el caso de los Nomencladores Asistenciales como CIE, CIAP entre otros, que son estándares, no permite que se repita una enfermedad.

Se le mostrará un mensaje al usuario al eliminar datos con Integridad Referencial, o sea, que posea una información asociada a otra funcionalidad.

Para verificar un error es necesario llevar a cabo una excepción que en caso de ser recibida rápidamente permite gestionar el error. Una excepción es un suceso que ocurre durante la puesta en marcha de un programa, provoca la interrupción del flujo normal de las sentencias. Controlan los errores sin confundir el código con muchas instrucciones de control de error.

En este capítulo se realizó la implementación del módulo de configuración para escritorio. La aplicación proporciona la gestión de información, se brinda de esta manera a los usuarios del sistema una solución funcional de fácil interacción con nuevas opciones para la visualización de la información. Se mostraron los Artefactos Diagrama de Despliegue y de Componentes. Se estableció una línea de estándares de diseño y codificación mediante la cual se rigen todos los procesos realizados para lograr igualdad en el trabajo.

Conclusiones

La realización del presente trabajo ha posibilitado cumplir con los objetivos propuestos, por lo que se pueden plantear las siguientes conclusiones:

1. Un análisis minucioso de las aplicaciones existentes arrojó como resultado que no existe una solución informática que se ajuste a las políticas del SNS, por lo que la aplicación concebida es la primera de su tipo en el mundo.
2. El uso de las tecnologías definidas por el CESIM permitió la elaboración de un producto que responde a las políticas de desarrollo establecidas por el MINSAP, posibilitando así la reducción de costos.
3. La implementación de las funcionalidades del Módulo de Configuración garantiza la gestión de nomencladores médicos, la seguridad del sistema e interoperabilidad entre los módulos del SIAPS.

Por todo lo expuesto anteriormente, se logró implementar el Módulo Configuración para Escritorio, que permite dar soporte a todos los demás módulos; gestiona todas las configuraciones: usuarios, área de salud, nomencladores médicos, gestión de codificadores y estándares internacionales y garantizará la seguridad del sistema.

Recomendaciones

Las recomendaciones de la investigación están dirigidas a sugerir acciones para complementar el producto obtenido. Por lo que para el buen desempeño y puesta en marcha de la aplicación, se hace las siguientes recomendaciones:

1. Utilizar el Framework JavaFX, el cual constituye una tecnología novedosa para proporcionar un entorno visual del producto más atractivo.
2. Reemplazar la versión 1.8 del HSQLDB Hypersonic por la 2.0 que permite la integridad referencial entre esquemas.
3. Establecer un plan de capacitación en aras de adiestrar al personal médico que va a interactuar con el sistema.
4. Actualizar las funcionalidades necesarias para permitir la interoperabilidad de los módulos existentes con los que posteriormente se integrarán al sistema.

Referencias Bibliográficas

1. Álvarez, D. M. G. Sistema de Información Estadística de Salud Cubano.
2. González, Maricarmen http://www.cubavision.cubaweb.cu/comentarios_detalles.asp?ID=263
3. Aplicación en capas. [En línea] [Citado el: 2008 de Marzo de 2008.] http://oness.sourceforge.net/proyecto/html/ch03s02.html.
4. Ídem a la referencia 3.
5. Ídem a la referencia 3.
6. García de Jalón, Javier, Rodríguez Iñigo, Mingo José Ignacio, Alfonso Brazález Aitor Imaz, Larzabal Alberto, Calleja Jesús, García Jon. Aprenda Java como si estuviera en primero. Escuela Superior de Ingenieros Industriales. Universidad de Navarra. España. Enero 2000
7. GNU Proyecto, Free Software Foundation. [En línea] [Citado el: 25 de Marzo de 2008.] http://www.gnu.org/philosophy/free-sw.es.html.
8. Ídem a la referencia 7.
9. Cabello Albino, Clifton. 2007. Sitio Web: El mundo Linux. Definición de Linux. [En línea] 2007. [Citado el: 20 de Marzo de 2008.] http://www.elmundolinux.com/definicionlinux.php.
10. APSI_SW_DI_017_alasSIAP_Metodología_Enfocado_Procesos V1.0.doc.
11. Ídem a la referencia 10.
12. Rumbaugh James, Jacobson Ivar, Booch Grady. El Lenguaje Unificado de Modelado. Manual de Referencia. Pearson Education. 2007. [Consultado el 18 de febrero de 2010].
13. **Gómez, Yaney.** Repositorio de documentos para el Departamento de Atención Primaria de Salud del Centro de Infomática Médica. [En línea] 2010. https://repositorio.cesim.prod.uci.cu/svn/aps/configuracion/DESKTOP/EXPEDIENTE_DE_PROYECTO/
14. Ídem a Referencia 4.
15. Ídem a Referencia 4.

Bibliografía

1. Aplicaciones de Escritorio Eficientes. http://www.javahispano.org/contenidos/es/aplicaciones_de_escritorio_eficientes/

2. Aplicación en capas. [En línea] [Citado el: 2008 de Marzo de 2008.] http://oness.sourceforge.net/proyecto/html/ch03s02.html.

3. APSI_SW_DI_017_alasSIAP_Metodología_Enfocado_Procesos V1.0.doc.

4. Álvarez, D. M. G. Sistema de Información Estadística de Salud Cubano.

5. Beans Binding. http://www.elguille.info/colabora/NET2005/Percynet_EnlaceDatos_Parte1.htm.

6. Cabello Albino, Clifton. 2007. Sitio Web: El mundo Linux. Definición de Linux. [En línea] 2007. [Citado el: 20 de Marzo de 2008.] http://www.elmundolinux.com/definicionlinux.php.

7. Delgado Ramos, Ariel y Vidal Ledo, María. 2006. Informática en la salud pública cubana. http://bvs.sld.cu/revistas/spu/vol32_3_06/spu15306.htm

8. Definición de Configuración. http://www.definicionabc.com/tecnologia/configuracion.php

9. Definición de Configurar. http://www.mastermagazine.info/termino/4404.php

10. González, Maricarmen http://www.cubavision.cubaweb.cu/comentarios_detalles.asp?ID=263

11. García de Jalón, Javier, Rodríguez Iñigo, Mingo José Ignacio, Alfonso Brazález Aitor Imaz, Larzabal Alberto, Calleja Jesús, García Jon. Aprenda Java como si estuviera en primero. Escuela Superior de Ingenieros Industriales. Universidad de Navarra. España. Enero 2000

12. GNU Proyecto, Free Software Foundation. [En línea] [Citado el: 25 de Marzo de 2008.] http://www.gnu.org/philosophy/free-sw.es.html.

13. **Gómez, Yaney.** Repositorio de documentos para el Departamento de Atención Primaria de Salud del Centro de Infomática Médica. [En línea] 2010. https://repositorio.cesim.prod.uci.cu/svn/aps/configuracion/DESKTOP/EXPEDIENTE_DE_PROYECTO/

14. Gómez Fernández, Vanesa Ing., Rivero Alemán, Rubén Ing. 2009. “Estadística Descriptiva del Registro de Fallecidos”.

BIBLIOGRAFÍA

15. Martínez, German, 28 de octubre de 2009. Sistema de Información para la Gerencia Hospitlaria (SIGHO). http://sigho.ses-gro.gob.mx/ y http://www.sigho.gob.mx/quees.htm

16. Pompa Sourd, Frank Ing. Sistema Informático para la Atención Primaria de Salud (APUS). http://www.hab2001.sld.cu/arrepdf/00196.pdf , http://www.sld.cu/instituciones/cedisap/atepri1.htm

17. Rumbaugh James, Jacobson Ivar, Booch Grady. El Lenguaje Unificado de Modelado. Manual de Referencia. Pearson Education. 2007. [Consultado el 18 de febrero de 2010].

18. Sánchez González, Miguel Angel. 2002. Historia, teoría y método de la medicina: introducción al pensamiento médico. http://books.google.com.cu/books?id=USvkOQBxZfMC&pg=PA458&lpg=PA458&dq=Nomenclaturas+y+clasificaciones+m%C3%A9dicas&source=bl&ots=IZnnzhKFl2&sig=L-BcUghV5y5J0qqxs3UVtk5vBwc&hl=es&ei=5DWoS6ztDsX_lgeftIGqAQ&sa=X&oi=book_result&ct=result&resnum=2&ved=0CAkQ6AEwAQ#v=onepage&q=Nomenclaturas%20y%20clasificaciones%20m%C3%A9dicas&f=false

19. Stusser Beltranena, Rodolfo J. y Rodríguez Díaz, Alfredo. 2006. La informatización de la atención primaria de salud. http://bvs.sld.cu/revistas/mgi/vol22_4_06/mgi12406.htm

20. Sistema Integral de Administración en Salud ANGEL. http://www.conmed.com.ar/angel.html y http://www.proyectoangel.net/

21. SWING. http://msdn.microsoft.com/es-es/library/96etzbdw%28VS.80%29.aspx y http://www.dcc.uchile.cl/~lmateu/CC60H/Trabajos/edavis/swing.html

Glosario de Términos

Aplicación o Sistema Informático: Programas con los cuales el usuario final interactúa a través de una interfaz y que realizan tareas útiles para éste.

Cliente Servidor: Modelo para construir sistemas de información, que se sustenta en la idea de repartir el tratamiento de la información y los datos por todo el sistema informático, permitiendo mejorar el rendimiento del sistema global de información.

Concurrencia: Ejecución simultánea de dos o más actividades durante el mismo intervalo de tiempo.

Componente: Parte física y reemplazable de un sistema que se ajusta a, y proporciona la realización de, un conjunto de interfaces.

Deficiencia: Es toda pérdida o anormalidad de una estructura o función psicológica, fisiológica o anatómica.

Dominio: Área de conocimiento o actividad caracterizada por un conjunto de conceptos y terminología comprendidos por los practicantes de ese dominio.

Equipos Básicos de Salud: Binomio conformado por el médico y enfermera de la familia, que atiende una población geográficamente determinada, que puede estar ubicado en la comunidad, centros laborales o educacionales.

Interoperabilidad: Condición necesaria para que los usuarios (humanos o mecánicos) tengan un acceso completo a la información disponible. Entre las iniciativas recientes más destacadas para dotar a la Web de interoperabilidad se encuentran los servicios Web y la Web semántica.

Informática: Disciplina que estudia el tratamiento automático de la información utilizando dispositivos electrónicos y sistemas computacionales.

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

Morbilidad: Es el estudio de los efectos de una enfermedad en una población en el sentido de la proporción de personas que enferman en un sitio y tiempo determinado.

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

GLOSARIO DE TÉRMINOS

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

Servicio Web: es una colección de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.

Software: Conjunto de programas y procedimientos necesarios para hacer posible la realización de una tarea específica, en contraposición a los componentes físicos del sistema.

Software Libre: Es el software que, una vez obtenido, puede ser usado, copiado, estudiado, modificado y redistribuido libremente.

Subsistema: Agrupación de elementos, de los que algunos constituyen una especificación del comportamiento ofrecido por los elementos contenidos.

Splash: Especie de pantalla de presentación que se muestra usualmente durante la carga de una aplicación con el objetivo de entretener e informar al usuario sobre temas como el copyright y el nombre del autor, para evitar que el usuario piense que el proceso de carga no está llevándose a cabo correctamente o que se haya paralizado por algún motivo.

Unidad de Salud: Centro de trabajo que pertenece al Ministerio de Salud Pública (MINSAP).