Universidad de las Ciencias Informáticas

Facultad 7

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

ADQUISICIÓN DE DATOS TRANSMITIDOS POR

MÁQUINAS DE ANESTESIA PARA EL SISTEMA ALAS HIS

Autores: Diana Lázara Durán Dor

Yunier Silva Quevedo

Tutores: Ing. Yoandy González Martínez

Ing. Rodney Ledo Ramírez

Cotutor: Ing. Leitniz Pérez Buján

La Habana, junio 2013 “Año 55 de la Revolución”

Datos de contacto

Ing. Yoandy González Martínez

Ingeniero en Ciencias Informáticas, graduado en la Universidad de las Ciencias Informáticas en el año 2007. Posee la categoría docente de Profesor Asistente. Ha impartido asignaturas de Práctica Profesional 1, Programación Web, Gráficos por Computadora y Prueba de nivel de programación. Ha participado en varios proyectos de desarrollo vinculados al perfil de salud. Actualmente labora en el Departamento de Sistemas de Gestión Hospitalaria del Centro de Informática Médica (CESIM), desempeñándose como Jefe de Departamento.

Correo Electrónico: ygonzalezm@uci.cu

Ing. Rodney Ledo Ramírez:

Graduado en la Universidad de Ciencias Informáticas (UCI) como Ingeniero en Ciencias Informáticas en el año 2009. Profesor Instructor del departamento Sistemas de Gestión Hospitalaria. Se ha desempeñado como líder del equipo de desarrollo del módulo de Admisión y Visor HC. Ha impartido la asignatura de Práctica Profesional. Ha tutorado varios trabajos diploma que cuya calificación ha sido de 5 puntos.

Correo electrónico: rledo@uci.cu

Ing. Leitniz Pérez Buján:

Ingeniero en Ciencias Informáticas, graduado en el año 2012 en la Universidad de las Ciencias Informáticas. Es recién Graduado en Adiestramiento, ha impartido la asignatura de Gráficas por Computadora como alumno ayudante. Actualmente labora en el Departamento de Sistemas de Gestión Hospitalaria del Centro de Informática Médica (CESIM).

Correo Electrónico: lbujan@uci.cu

Agradecimientos

Queremos darle un especial agradecimiento a Daniel Thevenet por toda su ayuda y preocupación en este trabajo, a los médicos del Hospital “Calixto García”, en especial a Piedra y Alfre. A nuestros tutores, en especial al co-tutor Leitniz, al profesor Rabelo por su apoyo en la revisión del documento siempre que lo necesitábamos, a Nadiezka por ser más que oponente tutora y ayudarnos en la perfección del trabajo, a los demás profesores del proyecto que siempre estuvieron dispuestos a ayudarnos y a todos los demás que pusieron un granito de arena en nosotros para poder cumplir el sueño de ser ingenieros informáticos.

A todos nuestros compañeros de estos 5 años por estar siempre juntos en esta etapa difícil y a la vez linda de la vida.

De Diana:

Quiero agradecerle a mi madre por estar siempre a mi lado, por haberme apoyado en estos años y siempre, por ser mi guía, por ser la persona que siempre está ahí cuando lo necesito, por ser mi razón de ser.

A mis tías Gilda y Jacqueline, por apoyarme y ayudarme en todo.

A mi abuela por su preocupación en todo este tiempo.

A mi novio, por su apoyo, comprensión y por estar siempre a mi lado desde que nos conocimos.

A mis hermanitas Jane, Marilyn y Magdi, y en especial a Jane por ayudarme y estar siempre a mi lado en todo este tiempo.

A mi papá y a mis tíos, por su cariño.

A mis primos que los quiero mucho.

A mi compañero de tesis y amigo, por pasar por esta etapa juntos y no dejarme sola nunca.

A todos mis amigos y en especial a Héctor y a Lily.

De Yunier:

Especialmente quiero agradecerles a mis padres, pues sin sus esfuerzos hubiera sido imposible mi estancia en la UCI durante cinco años y haber logrado graduarme como siempre aspiré. Por apoyarme en todo momento y darme fuerzas para seguir adelante en momentos difíciles.

A mi hermano que siempre me ha apoyado y me ha inspirado a seguir sus pasos para hacerme ingeniero.

A mis abuelos que están a mi lado y mi abuela que ya no está pero seguro le gustaría mucho verme graduado.

A mis tíos y en especial a mi tía Leydis que siempre me ayudó mucho.

A todos mis primos, por su cariño.

A mi compañera de tesis por su entrega, dedicación y confianza en mí.

A todos mis amigos y compañeros de la UCI y de mi municipio que de una forma u otra me ayudaron y apoyaron durante estos cinco años, especialmente a Jorge Luis y a Pedro.

Dedicatoria

De Diana:

A mi madre por todo el apoyo y la confianza depositada en mí.

De Yunier:

A mis padres por su esfuerzo y dedicación para hacer cumplir mis sueños.

Resumen

La investigación tiene como propósito crear un sistema para la adquisición de datos transmitidos por máquinas de anestesia para el sistema alas HIS. Los datos obtenidos deberán ser almacenados y mostrados gráficamente en tiempo real. Para cumplir los objetivos trazados se realizó un análisis de los procesos asociados a los equipos de anestesia, se estudiaron sistemas con objetivos similares, además de tecnologías y herramientas para su implementación.

El tema aborda la conexión entre una máquina de anestesia y un ordenador a través del puerto serie estándar RS 232. A partir de la lectura, se propone una solución para decodificar los datos asociados a los indicadores de anestesia. Para implementar el componente se realizó un estudio del protocolo Medibus, encargado de establecer la comunicación.

Para la realización del sistema se utilizó Java como lenguaje de programación y su API (javax.comm) de comunicación serie con los dispositivos externos. Se usó además NetBeans 7.1 como Entorno de Desarrollo Integrado (IDE por sus siglas en inglés), la librería JFreeChart 1.0.14 para la realización de gráficos y PostgreSQL 9.1 como sistema gestor de base de datos.

La herramienta desarrollada consiste en una aplicación de escritorio que emplea interfaces gráficas intuitivas y usables, basadas en la experiencia de usuario para este tipo de aplicaciones. Proporciona funcionalidades que permiten establecer la comunicación entre un ordenador y máquinas de anestesia cuyo protocolo sea Medibus. Los datos adquiridos son decodificados y representados de manera gráfica, facilitando su interpretación. Estos son además registrados en la base de datos.

Palabras clave:

Adquisición, anestesia, protocolo de comunicación, puerto serie.

Índice de contenido

Índice de tablas

Índice de figuras

Introducción

En el Hospital General de Massachusetts en el año 1846 se logró realizar una intervención quirúrgica sin presencia de dolor, de esta forma el arte quirúrgico se transformaba en ciencia. Se había conseguido la síntesis de la anestesia como arma eficaz de lucha contra el dolor y la cirugía adquiría su verdadera significación. Desde entonces se le atribuyó el nombre de anestésico a la mezcla de medicamentos que se le suministraba al organismo de acuerdo al estado físico del paciente y al procedimiento quirúrgico a practicar. Los anestésicos permitieron a los cirujanos realizar intervenciones por más tiempo y sin presencia de dolencias.

Luego de este avance alcanzado por la ciencia surgieron otras necesidades en materia de técnicas e instrumentos, puesto que debían ser cada vez más precisos y eficaces. De esta forma se comenzó a demandar más de la industria, que a su vez sufrió transformaciones para adactarse a las nuevas exigencias médicas. La creación de medios auxiliares provocó un nuevo paradigma tecnológico que los profesionales del ramo deberían enfrentar.

A partir del año 1846 ante la necesidad de supervisar el comportamiento de los indicadores vitales del paciente durante el proceso de anestesia, se fabricaron las máquinas de anestesia. Inicialmente fueron equipos capaces de administrar gases anestésicos aprovechando la absorción pulmonar. Luego evolucionaron hacia formas más sofisticadas, capaces de vaporizar líquidos y de controlar las funciones respiratorias, incorporando capacidades de monitorización de las variables fisiológicas del paciente como el VT (Volumen tidal), VM (Volumen minuto), PIP (Presión pico), entre otras. (1)

Estos equipos se encuentran en la mayoría de las áreas de cirugía y son controlados por los anestesiólogos. Las máquinas de anestesia, como hoy día son nombradas, son equipos compuestos por elementos mecánicos, neumáticos y electrónicos. Administran de manera segura y por vía pulmonar, con ventilación espontánea o mecánica, gases como el oxígeno, el óxido nitroso, el aire y vapores anestésicos que permitan realizar un correcto anestesiado, monitorizando además todas las funciones requeridas. (2)

El desarrollo de las máquinas de anestesia y el avance tecnológico de la informática conllevó a la idea de crear sistemas para la adquisición de datos. Uno de ellos son los SCADAs **(**del inglés *Supervisory Control and Data Acquisition)*, conjunto de software orientados a monitorear, controlar y automatizar los diferentes procesos registrados por distintos tipos de dispositivos. (3)

En Cuba existen hospitales que adquieren sus máquinas de anestesia en otros países y a costos muy elevados, por esta razón no compran los software responsables de mostrar y almacenar la información que generan estas máquinas. En estos centros hospitalarios no existe un sistema que supervice y controle en tiempo real los datos transmitidos por los equipos de anestesia, por lo que los datos se gestionan manualmente, aumentando la posibilidad de extravíos de información valiosa y sensible de los pacientes.

La informatización de la atención primaria, secundaria y terciaria de la salud requiere de múltiples proyectos de investigación científica multidisciplinaria, de desarrollo tecnológico y de abundante intercambio académico. Esto se debe no solo a los grandes retos de carácter informático, sino también a los enormes desafíos médicos y científicos-tecnológicos de la toma de decisiones, vinculadas a cada nivel de atención y de especialización.

La informatización de la salud pública en Cuba constituye una de las tareas principales del Instituto Central de Investigación Digital (ICID). Desde hace varios años esta institución ha comenzado a desarrollar sistemas para la adquisición de datos transmitidos por equipos médicos, sin embargo no ha sido desarrollado ningún sistema informático que permita interpretar y procesar la información expedida por las máquinas de anestesia.

La Universidad de las Ciencias Informáticas (UCI) es una institución dedicada a la formación universitaria y a la creación de software en ramas como la educación, el deporte, las telecomunicaciones, la industria y la salud. Su proceso productivo está organizado estructuralmente en centros de desarrollo, uno de ellos el Centro de Informática Médica (CESIM) que se encarga de desarrollar proyectos vinculados a la salud, entre los que se encuentra el Sistema de Información Hospitalaria alas HIS.

El sistema alas HIS cuenta con diversos módulos, uno de ellos es Bloque Quirúrgico que incluye entre sus funciones el almacenamiento de los indicadores mostrados por las máquinas de anestesia. Esta funcionalidad presenta inconvenientes ya que los anestesiólogos deben introducir los datos de forma manual en el sistema, siempre en una estructura de formulario. Aunque pueden ser consultados los datos del paciente no se visualizan de manera gráfica, que contituye una forma más rápida y ágil de monitorear los signos vitales del paciente durante la intervención quirúrgica.

La opción actual de entrada de la información al sistema está expuesta a posibles errores humanos, tanto por la lectura, como por la introducción de los datos. La aplicación no cuenta además con funcionalidades gráficas que describan el comportamiento de los indicadores del paciente respecto al

tiempo. Estas gráficas representarían un soporte para las futuras investigaciones clínicas, herramientas para tomar decisiones y un respaldo en casos médico-legales.

Basado en lo anterior se define como **problema a resolver**: ¿Cómo adquirir en tiempo real las variables fisiológicas provenientes de las máquinas de anestesia para su uso por el sistema alas HIS?

Se define como **objeto de estudio:** el proceso de adquisición de datos provenientes de equipos médicos, enmarcado **en el campo de acción:** proceso de adquisición de variables fisiológicas provenientes de las máquinas de anestesia.

Para resolver el problema identificado se propone el siguiente **objetivo general:** Implementar un componente que adquiera las variables fisiológicas provenientes de las máquinas de anestesia para su interpretación por el sistema alas HIS.

Para dar cumplimiento al objetivo anteriormente planteado se definen las siguientes **tareas a desarrollar**:

1. Caracterizar las soluciones informáticas relacionadas con la comunicación con dispositivos externos con el fin de evaluar las tendencias actuales vinculadas al estado del arte de la investigación.
2. Definir mediante una revisión bibliográfica, el protocolo de comunicación indicado para la aplicación, que permita la implementación de sus especificaciones técnicas para la comunicación.
3. Desarrollar las funcionalidades necesarias que permitan la representación gráfica de los datos para su interpretación y almacenamiento, con fines investigativos, docentes y de carácter médico-legal.
4. Desarrollar, basado en el Proceso Unificado de Desarrollo, los artefactos resultantes del Modelo de Dominio, Modelado de Sistema, la Gestión de Requisitos, el Diseño y la Implementación.
5. Obtener una solución, que adquiera automáticamente los datos procedentes de máquinas de anestesia con el fin de reducir los posibles errores humanos en la introducción de los datos al sistema alas HIS.

Con el desarrollo de las funcionalidades asociadas a la adquisición de datos desde máquinas de anestesia se esperan obtener los siguientes resultados:

1. Mejorar el proceso de registro por parte de los anestesiólogos de las variables fisiológicas generadas por la máquina de anestesia.
2. Facilitar la interpretación de la información del paciente adquirida durante la intervención quirúrgica.
3. Aumentar la credibilidad de la información almacenada, pues esta podrá ser procesada libre de errores de lectura humana o introducción manual de los datos.

Para un mayor entendimiento el documento estará estructurado de la manera siguiente:

Capítulo 1. Fundamentación teórica del sistema de adquisición de datos: Resume lo relacionado con el campo de acción. Presenta el estudio realizado de los sistemas de adquisición de datos desde equipos médicos, además de abordar los temas de la comunicación por puerto serie y de los protocolos que utilizan las diferentes máquinas de anestesia. Se fundamentarán las tecnologías, metodologías y herramientas de desarrollo a utilizar.

Capítulo 2. Características del sistema de adquisición de datos: Descripción del negocio a automatizar. Se muestra un marco conceptual de la información que será procesada por el sistema, así como las funcionalidades y los requerimientos deseados.

Capítulo 3. Arquitectura y diseño del sistema de adquisición de datos: Descripción y análisis de la solución que se propone para dar respuesta a la problemática planteada.

Capítulo 4. Características de la implementación del sistema de adquisición de datos: Se exponen las estrategias de codificación, la descripción del flujo de implementación y estilos utilizados.

CAPÍTULO 1. Fundamentación teórica del sistema de adquisición de datos

En el presente capítulo se abordan una serie de conceptos relacionados con la especialidad de anestesiología, se muestran breves descripciones de sistemas actuales de control y adquisición de datos transmitidos por equipos médicos, dando de esta forma el estado del arte y sus características esenciales. Se fundamenta la selección de tecnologías, metodologías y un grupo de herramientas utilizadas en el desarrollo del componente para la conexión.

1.1 Conceptos básicos relacionados con el dominio del problema

En un acto quirúrgico la suministración de anestesia facilita la realización de la cirugía con ausencia de dolor y origina un riesgo mínimo para el paciente, proporcionando una recuperación óptima. La anestesia suministrada al organismo está compuesta por medicamentos seleccionados en cantidades determinadas, esto depende del estado físico del paciente y del procedimiento quirúrgico que se le llevará a cabo.

Las formas de proporcionar anestesia a los pacientes son: (1)

- General
- Local
- Regional

El proceso de suministración de anestesia al paciente se realiza en tres etapas fundamentales, inducción, mantenimiento y reversión. En la inducción los anestesiólogos comienzan el proceso de sedación una vez suministrados los fármacos utilizando técnicas psicológicas, respuestas a estímulos u otros métodos para verificar el efecto de las drogas. En la etapa de mantenimiento la principal función es conservar el correcto funcionamiento de los sistemas vitales, ya que el paciente se encuentra inconsciente. La etapa de reversión se denomina al espacio de tiempo en que el paciente vuelve a un estado de consciencia y se recuperan los reflejos.

Las máquinas de anestesia registran todo lo relacionado con la vía aérea del paciente. Las señales ventilatorias y los parámetros calculados que monitoriza son: (5)

- VT. Volumen corriente: Volumen que entra y sale del paciente en cada ciclo respiratorio.
- VM. Volumen minuto: Volumen que entró y salió del paciente al cabo de un minuto y se calcula,

VT x F.

- PIP. Presión pico: Pico máximo de presión en la vía aérea.
- MEDIA. Presión media: Valor medio de la curva de presión.
- PEEP. Presión positiva al final de la espiración: Presión que queda al final de la espiración, usualmente se mantiene constante durante las espiraciones.
- F. Frecuencia respiratoria: Cantidad de ciclos respiratorios por minuto.
- ALARMAS: Alarmas activadas durante la ventilación.
- P: curva en tiempo real de presión de vía aérea en función del tiempo.

1.2 Antecedentes

Se ha realizado un estudio de algunos de los sistemas actuales que existen relacionados con la adquisición de datos, presentando un resumen general de sus funcionalidades para así conformar parte del estado del arte de la investigación.

Sistemas informáticos analizados:

Sistema alas HIS

Es un Sistema de Información Hospitalaria que posee varios módulos que permiten la gestión de la información de las distintas áreas en las instituciones hospitalarias. Contiene un componente denominado “Adquisición de datos” que permite la conexión con los equipos médicos que se encuentran en los laboratorios clínicos y la transferencia de datos al sistema alas HIS. Cuenta con 11 equipos conectados al sistema y utilizan una arquitectura cliente-servidor. Para la implementación usan C++ y algunas de las bibliotecas del *framework boost*.

El subsistema está formado por dos componentes:

Manejadores de dispositivos (*Drivers*): Son módulos independientes en forma de bibliotecas dinámicas que implementan los protocolos para la comunicación con los equipos.

Módulo de Recolección (*HISDataServer*): Responsable de almacenar en la base de datos del HIS los resultados provenientes de los equipos médicos, así como hacerle llegar a los manejadores las solicitudes creadas en el sistema.

Monitor

Programa que grafica los datos provenientes de máquinas de anestesia. Tiene una interfaz intuitiva y muy versátil. Existe una versión para ordenadores y otra versión para MAC (Macintosh). La versión para MAC permite imprimir la hoja del registro anestésico. En ambas versiones se pueden grabar los datos en formato ASCII y exportarlos a una hoja de Excel. La interfaz con la máquina de anestesia se realiza a través de un puerto serial. Como la mayoría de las computadoras actuales no cuentan con este puerto, se hace necesace necesario un convertidor serial-USB, lo mismo se aplica para la versión MAC. Las variables obtenidas de las máquinas de anestesia no pueden editarse, lo que representa una gran ventaja puesto que brinda mayor seguridad a los datos del paciente. (6)

Características del sistema

- Posee una interfaz amigable.
- Los datos pueden ser guardados como un archivo Excel o como un archivo del sistema "monitor".
- El número máximo de dispositivos conectados es ocho.
- Se visualiza en tiempo real las variables fisiológicas del paciente.
- Los marcadores de eventos se pueden introducir en tiempo real o luego de la intervención.
- Se pueden visualizar los datos anteriores sin interrumpir la grabación de los nuevos.

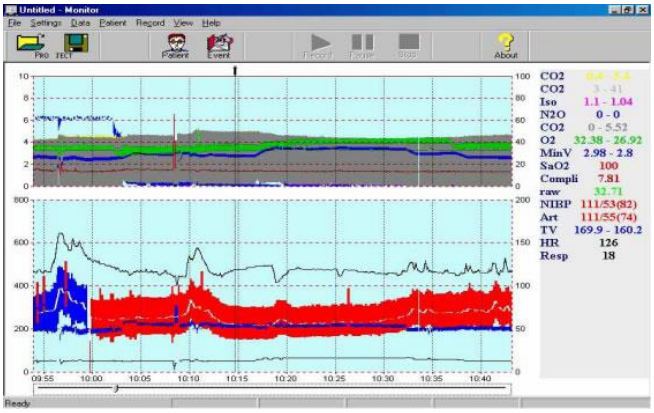

Figura 1:Interfaz de monitor

Regane

Es una solución integral que captura las señales ventilatorias de la máquina de anestesia marca Medec, permitiendo su análisis y revisión posteriormente. REGANE registra los datos de pacientes para asociarlos en la historia clínica electrónica en tiempo real. Implementa su propio protocolo de comunicación y el producto debe ser ejecutado en plataformas MS Windows XP/Vista. La empresa utilizó como lenguaje de programación MS Visual Basic 2008. La interfaz entre la máquina de anestesia y el ordenador consiste en un cable serial DB9-M a DB9-F directo si el ordenador dispone de puerto serie, en caso contrario se pueden utilizar adaptadores USB-RS 232. Para la adquisición de datos, este presenta en pantalla y almacena en disco los datos que recibe por el puerto serie. Para esto se definió una rutina de adquisición que se basa en detectar cada paquete, identificar la información contenida, interpretarla, mostrarla en pantalla y finalmente almacenarla. (7)

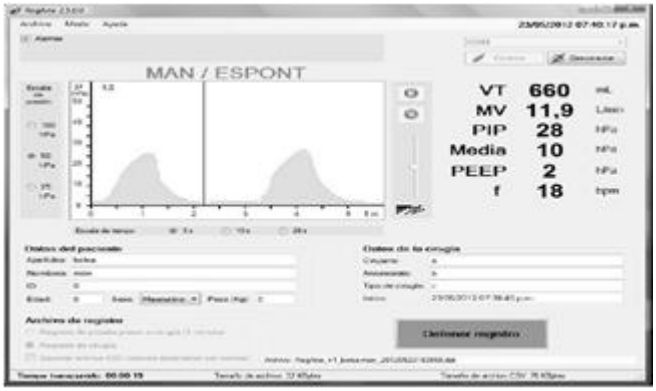

Figura 2: Interfaz de regane

Sistemas SCADA

Una de las utilizaciones de los sistemas SCADAs es en la automatización de los laboratorios clínicos de los hospitales. En este ambiente también se emplean dispositivos autoanalizadores, que le envían los datos a los sistemas SCADA. La empresa líder a nivel mundial en la fabricación de autoanalizadores es Roche. Esta compañía está dividida en dos líneas de producción: la farmacéutica y la de diagnóstico; dispone por tanto de una amplia gama de productos dirigidos a los laboratorios clínicos entre los que se encuentran: (8)

- HITACHI/ROCHE 902
- HITACHI/ROCHE 912
- HITACHI ROCHE 917

- COBAS B121
- COBAS B121 BGE
- ELECSYS 2010

Sistema de información clínica web (Innovian® Solution Suite)

Recopila, presenta y almacena datos automáticamente de monitores con señales vitales, ventiladores y otros dispositivos médicos conectados, a los que luego podrá acceder vía web. Está diseñado para los médicos que necesitan acceder a la información del paciente tanto en la cabecera como en ubicaciones remotas. Captura continuamente los datos vitales de los dispositivos médicos y los sistemas de información del bloque preoperatorio y cuidados intensivos, e integra dichos datos en una interfaz de fácil navegación a la que se puede acceder en cualquier lugar mediante la red del hospital.

Para la recopilación continua de datos este captura las señales vitales mientras se transporta al paciente. Así que la recopilación de datos sigue siendo continua independientemente de si transporta al paciente del quirófano a la Unidad de Cuidados Intensivos (UCI), lo lleva a radiología o a la sala de hemodinámica, o simplemente si cambia las camas en la UCI. Está integrado con monitores *Infinity*, que permiten separar el monitor de la estación de trabajo y llevarlo con el paciente. Cuando el paciente vuelve o se cambia a otra cama se conecta de nuevo el monitor y los datos recopilados durante el transporte se cargan automáticamente en la base de datos *Innovian*. (9)

Driver EPP (Encriptado Pin Pad)

El proyecto consta de dos desarrollos de software en diferentes niveles, un driver de un dispositivo EPP y una aplicación cliente de *test*. Esta forma parte de un conjunto de *drivers* de diferentes dispositivos utilizados conjuntamente en Terminales Auto consulta ATM que respetan un estándar de comunicación homogéneo utilizando XMLs para ejecutar el modelo de Petición-Respuesta. El driver está realizado en el lenguaje C/C++ y se comunica con el EPP a través de un enlace RS232. Este le envía al EPP todos los comandos necesarios para ejecutar sus funciones utilizando el protocolo de paquetes de comunicación específico del EPP.

El sistema permite comunicarse con su aplicación Cliente remotamente, otorgando mayor flexibilidad y facilidad de control. Trabaja bajo el sistema operativo Windows XP SP2. El protocolo utilizado en el *driver* para comunicarse con su aplicación cliente es por medio de XMLs. Estos XMLs tienen en sus

tags el comando requerido de ejecución por parte de la aplicación cliente al EPP, intermediando el *driver* para interpretarlo y reprocesarlo en el protocolo de comunicación del EPP. (10)

Luego del estudio de los sistemas de adquisición de datos antes mencionados se llegó a la conclusión que estos almacenan los datos adquiridos en tiempo real, muestran una interfaz con los datos claros y bien estructurados, y son sistemas de fácil manipulación por los anestesiólogos. Pero también son sistemas difíciles de adquirir por su costo en el mercado, y presentan dificultades para integrarse al sistema alas HIS.

Los sistemas anteriores no son diseñados para trabajar como componentes de HIS, puesto que los datos adquiridos por ellos son almacenados en archivos locales de su propio sistema y no existe una solicitud de intervención realizada con anterioridad. Los datos del paciente, del anestesiólogo y del cirujano deben introducirse manualmente, lo que puede conllevar a errores.

1.3 Comunicación serial

La comunicación entre equipos se realiza a través de un puerto de comunicación. En el caso de las máquinas de anestesia la transferencia de datos se realiza mediante el puerto de comunicación RS 232. El mismo es un conector DB-25, aunque actualmente se encuentra la versión de 9 pines DB-9.

El término serial viene del hecho de que el puerto serie serializa los datos. Esto significa que toma un byte de datos y transmite los ocho bits del byte uno a la vez. La ventaja del puerto serie es que necesita únicamente un solo cable para transmitir los ocho bits, mientras que un puerto paralelo necesita ocho. La desventaja es que dura ocho veces más para transmitir el dato que si hubieran ocho cables. (11)

Antes de cada byte de información, el puerto serial manda un bit de comienzo (*start* bit), el cual es un bit con valor de cero. Después de cada byte de datos, este manda un bit de parada (*stop* bit) para indicar que el byte ha sido completado. Algunas veces también se manda un bit de paridad.

Los puertos serie, también llamados puertos de comunicación (COM), son bi-direccionales. Lo que permite a cada dispositivo recibir datos, así como también transmitirlos. Los dispositivos seriales usan distintos pines para recibir y transmitir datos. Esto permite que la comunicación sea *full-dúplex*, en la cual la información puede viajar en ambas direcciones al mismo tiempo.

Este estándar se basa en una comunicación asíncrona. Los datos pueden ser transmitidos en cualquier momento por lo que deben tomarse precauciones para sincronizar la transmisión y recepción.

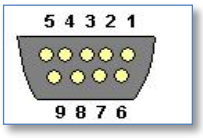

Figura 3: Líneas del puerto serial COM

Los dispositivos que utilizan cables serie para comunicarse se dividen en dos categorías:

- DCE (*Data Communicatios Equipment*). Equipo de comunicación de datos.
- DTE (*Data Terminal Equipment*). Equipo terminal de datos.

Usualmente la comunicación serial se utiliza para transmitir datos en formato ASCII. Para realizar la comunicación se utilizan las tres líneas fundamentales: el pin dos para la recepción de los datos, el pin tres para la transmisión de estos y el pin cinco como señal de tierra.

Señal	Siglas	DB-9
***Data CarrierDetect* (Detecta la portadora)**	DCD	1
***Recieved Data* (Recibe datos)**	RXD	2
***Transmit Data* (Transmite datos)**	TXD	3
***Data Terminal Ready* (Terminal de datos listo)**	DTR	4
***SignalGround* (Tierra)**	SG	5
***Data Set Ready* (Equipo de datos listo)**	DSR	6
***RequestToSend* (Solicita enviar)**	RTS	7
***Clear ToSend* (Disponible para enviar)**	CTS	8
***Ring Indicator* (Indica llamada)**	RI	9

Tabla 1: Funciones de los pines del puerto RS 232

Debido a que la transmisión es asíncrona, es posible enviar datos por una línea mientras se reciben datos por otra. Las características más importantes de la comunicación serial son la velocidad de transmisión, los bits de datos, los bits de parada y la paridad. Para que dos puertos se puedan comunicar es necesario que las características sean iguales.

Velocidad de transmisión (*baudrate*): Indica el número de bits por segundo que se transfieren y se mide en baudios (*bauds*).

Bits de datos*:* Se refiere a la cantidad de bits en la transmisión. Cuando la computadora envía un paquete de información, el tamaño de ese paquete no necesariamente será de ocho bits. Las cantidades más comunes de bits por paquete son cinco, siete y ocho bits.

Bits de parada: Usado para indicar el fin de la comunicación de un solo paquete. Los valores típicos son uno, uno punto cinco o dos bits. Debido a la manera como se transfiere la información a través de las líneas de comunicación y que cada dispositivo tiene su propio reloj, es posible que los dos dispositivos no estén sincronizados. Por lo tanto, los bits de parada no sólo indican el fin de la transmisión sino además dan un margen de tolerancia para esa diferencia de los relojes. Mientras más bits de parada se usen, mayor será la tolerancia a la sincronía de los relojes, sin embargo la transmisión será más lenta.

Paridad: Es una forma sencilla de verificar si hay errores en la transmisión serial. Existen cuatro tipos de paridad: par, impar, marcada y espaciada. La opción de no usar paridad alguna también está disponible.

Para la comunicación con el puerto serie RS 232 es necesario contar solamente con las líneas de recepción, transmisión y terminal de datos. Pues estas constituyen las más importantes en el proceso de transmisión. En este tipo de comunicación otra de las características que se debe tener en cuenta es que si por ambos lados el cable tiene el mismo conector los pines dos y tres deben ser intercambiados, esto trae consigo que se pueda pasar el flujo de información de un lado a otro sin problema en la transmisión o en la recepción.

1.4 Manejadores de dispositivos para la adquisición de datos

Para la comunicación de los sistemas operativos con los equipos se hace necesario la utilización de manejadores de dispositivos (*driver*). Los cuales están encargados de establecer el entendimiento

entre ambas entidades. Debido al desarrollo de las tecnologías existen diversos dispositivos que se pueden conectar a un ordenador para transmitir información. Estos deben de ser reconocidos y controlados por el sistema al cual estén conectado. Por lo que el manejador será específico para un dispositivo determinado, dependiendo del modelo y de la marca.

El manejador que se pretende desarrollar debe ser capaz de obtener,interpretar, graficar y almacenar los datos provenientes del equipo de anestesia. Teniendo en cuenta porque vía son transmitidos los datos y en que formato serán recibidos.

1.5 Protocolos

El estudio de las máquinas de anestesia evidenció que la comunicación de estas con sistemas informáticos es posible a través de un protocolo definido por el fabricante. Un protocolo es el conjunto de reglas que gobiernan el intercambio de datos entre dos entidades. (12)

Las funciones de un protocolo se pueden agrupar en:

Encapsulamiento: Añadir datos de información de control. Los datos se aceptan o generan por una entidad y se encapsulan en la unidad de datos de protocolo (PDU) junto con la información de control.

Segmentación y ensamblado: Cada protocolo puede dividir los datos de un nivel superior en bloques más pequeños. Denominaremos unidad de datos del protocolo (PDU, *Protocol Data Unit*) al bloque de datos a intercambiar entre dos entidades. El procedimiento contrario a la segmentación se denomina ensamblado. Los datos tendrán que ensamblarse recuperando el formato de los mensajes originales para ser entregados a la entidad de destino.

Control de la conexión: La transferencia de datos puede ser no orientada a la conexión, cada PDU se tratará de forma independiente de los PDUs recibidos con anterioridad. La orientada a la conexión se establece una asociación lógica o conexión entre dos entidades.

Entrega en orden:

- Control del flujo: El control de flujo es una operación realizada por la entidad receptora para limitar la velocidad y la cantidad de datos a enviar.
- Control de errores: Recuperar pérdidas o deterioro de los datos.
- Direccionamiento: La dirección del nivel de red se utiliza para encaminar los PDUs a través de la red hasta el sistema destino.

- La multiplexación entre capas: Puede realizarse de dos formas distintas:
    - La multiplexación ascendente consiste en que varias conexiones del nivel superior compartan o se multiplexen sobre una única conexión del nivel inferior.
    - La multiplexación descendente consiste en establecer una única conexión del nivel superior utilizando varias conexiones del nivel inferior.

Servicios de transmisión:

- Prioridad.
- Calidad del Servicio.
- Seguridad.

Para llevar a cabo la realización de un manejador de dispositivo lo primero que se debe tener en cuenta es el protocolo de comunicación que definió su fabricante. Este es el encargado de esclarecer como es la comunicación que establece el equipo con el medio exterior, así de cómo es que inicia la comunicación, cómo envía las tramas y que significan cada una de ella, y cómo concluye su transmisión.
Algunos de los protocolos de comunicación más usados para la automatización de procesos relacionados con el monitoreo de datos de equipos médicos se nombran: Modbus, Vitalink y Medibus.

Protocolo de comunicación Modbus

Es desarrollado por la empresa norteamericana MODICON, es público, sencillo de implementar y flexible. Se ha convertido en uno de los protocolos de comunicaciones más populares en sistemas SCADA.

Las comunicaciones MODBUS se pueden realizar en modo ASCII o en modo RTU. En modo ASCII los bytes se envían codificados en ASCII, es decir, que por cada byte a transmitir se envían dos caracteres ASCII de 2 bytes con su representación hexadecimal, lo que permite leer las tramas con un simple editor de texto. En modo RTU se envían en binario. En el modo ASCII las tramas comienzan por 3AH (carácter ':'), y terminan en 0DH-0AH y cada byte se envía como dos caracteres ASCII. En modo RTU no se utiliza indicador de inicio y final de trama. (13)

Protocolo de comunicación Vitalink

Vitalink recoge los parámetros y datos en forma de ondas de ventiladores y monitores Drager a través de un puerto serie de comunicaciones. Entre estos datos están: los límites actuales de las alarmas, las alarmas activas actuales, la fecha y la hora actual, la identificación del dispositivo, la configuración actual de este y otros datos. Para la verificación de los errores especifica que la paridad puede ser impar, par o ninguna. La cantidad de bits de datos transmitidos están entre los siete u ocho bits y para indicar el fin de la transmisión de un paquete, indica que los bits de parada pueden ser uno o dos. (14)

Protocolo de comunicación MEDIBUS

Medibus es un protocolo desarrollado por la empresa alemana Drager. Brinda valores medidos y curvas en tiempo real, establece valores y alarmas disponibles. Los datos que ofrece son públicos y este se emplea generalmente para el propósito de conectar monitores adicionales para transferencias de datos o para sistemas de gestión de datos. Es unidireccional, lo que significa que el control de un dispositivo externo no es la intención. La interfaz serie utilizada (RS 232) tiene una tasa de transmisión limitada, lo que limita los dispositivos médicos para transmitir sólo dos o tres curvas en tiempo real a una velocidad de muestreo de 16ms/20ms. (15)

Como se había mencionado anteriormente un protocolo de comunicación no es más que un conjunto de reglas de programación que están definidas para la interpretación de señales que son transmitidas a través de una conexión física.

El protocolo de enlace de datos está diseñado para enviar mensajes orientados a carácter. Los caracteres restringidos se colocan para que se pueda conocer cuándo es que comienza a llegar un mensaje.

Las señales enviadas por las máquinas de anestesia mediante el protocolo Medibus viajan codificadas, cuyos valores son hexadecimales específicos para cada variable o alarma que se quiera recibir desde estas, así mismo cada petición que se les hacen tienen un código asociado.

Medibus es el protocolo de comunicación que utilizan las máquinas de anestesia marca Fabius GS que actualmente son encontradas en los hospitales de Cuba. Estos equipos son fabricados también por la empresa alemana Drager.

A continuación se reflejan códigos y su significado a la hora de ser decodificados.

	Comandos de Solicitudes
Código	**Especificación**
24H	Solicitar actuales datos medidos
25H	Solicitar actuales límites de alarma bajo
26H	Solicitar actuales límites de alarma alta
27H	Solicitar las alarmas actuales
28H	Solicitar fecha y hora actuales
2AH	Solicitar mensajes de texto actuales
30H	No hacer nada (NOP)
52H	Petición de identificación del dispositivo
55H	Detener comunicación

Tabla 2:Comandos de solicitudes

Leyenda:

M= Los datos de medida.

LL= Límites de alarma baja.

HL= Límites de alarma alta.

Datos y límites

Datos relativos a las vías respiratorias y los límites					
Código	**Descripción**	**Formato**	**M**	**LL**	**HL**
73H	La media de la presión respiratoria	_XX_	x	x	
74H	Meseta Presión	_XX_	x		
78H	PEEP Respiración Presión	_XX_	x		

7DH	Pico Respiración Presión	_XX_	x		
82H	L Volumen Tidal	X.XX	x		
B9H	L volumen minuto respiratorio	XX.X	x	x	x
D7H	frecuencia respiratoria	XX_	x		

Tabla 3: Datos relativos a las vías respiratorias y los límites

Datos y límites relacionados al O2					
Código	**Descripción**	**Formato**	**M**	**LL**	**HL**
70H	El % inspiratorio O2	XXX_	x	x	x

Tabla 4: Datos y límites relacionados al O2

Datos y límites relacionados al Freshgas					
Código	**Descripción**	**Formato**	**M**	**LL**	**HL**
DDH	El flujo N2O	XXXX	x		
DEH	El flujo de aire	XXXX	x		
E2H	El flujo O2	XXXX	x		

Tabla 5: Datos y límites relacionados al Freshgas

Mensajes de alarmas en vías respiratorias			
Código	**Prior**	**Descrpción**	**Español**
19	22	Minute Volume < low limits	MIN VOL Bajo
9B	13	Minute Volume > high limits	MIN VOL Alto

DA	7	PEEP > high limits	PEEP Alto
08	31	Inspiratory Oxygen <Low limit	O2 INSP Bajo
37	12	Inspiratory Oxygen > high limit	O2 INS Alto
BE	8	O2 Measurement inoperable	O2 INSP INOP

Tabla 6: Mensajes de alarmas en vías respiratorias

1.6 Lenguaje de programación

Java

Para el desarrollo de la aplicación se hizo necesario utilizar el lenguaje de programación Java, pues con este se asimilaría el lenguaje de programación que utiliza el departamento de Gestión de Información Hospitalaria, al cual se va a asociar este sistema.

Java es un lenguaje de alto nivel orientado a objetos, lo que le da una gran ventaja frente a otros lenguajes porque permite la reutilización de código. Trabaja sobre un JVM (*java virtual machine*), lo que lo hace compatible con cualquier sistema operativo. El mismo permite el desarrollo de aplicaciones bajo el esquema de Cliente-Servidor, como de aplicaciones distribuidas, lo que lo hace capaz de conectar dos o más computadoras, ejecutando tareas simultáneamente, y de esta forma logra distribuir el trabajo a realizar. Como otras de las ventajas que presenta este lenguaje de programación es que presenta una buena gestión de memoria utilizando el proceso "***Garbage Collector***", el cual es ejecutado de manera trasparente al programador, posee facilidades de aprendizaje, además de venir acompañado de una serie de librerías estándar para realizar multitud de operaciones comunes a la hora de programar. (16)

1.7 Metodología de desarrollo de software

Como metodología de desarrollo de software se selecciona al Proceso Unificado de Desarrollo (RUP). Para el lenguaje de modelado se propone el Lenguaje Unificado de Modelado (UML).

Proceso Unificado de Desarrollo (RUP)

RUP es una plataforma flexible de procesos de desarrollo de software que ayuda brindando guías

consistentes y personalizadas de procesos. Divide en cuatro fases el desarrollo del software y cada una de estas es desarrollada mediante un ciclo de iteraciones. En cada ciclo se hace exigente el uso de artefactos, siendo por este motivo una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software.

El ciclo de vida de RUP tiene la característica de ser orientado a casos de uso, iterativo e incremental y centrado en la arquitectura, donde sus cuatro fases para el desarrollo de un software se definen en:

• **Inicio:** Se define y se delimita el proyecto.

• **Elaboración**: Su objetivo es obtener la arquitectura candidata del sistema.

• **Construcción**: Obtención del producto, apoyado en una fuerte documentación.

• **Transición:** Se alcanza la liberación del producto, listo para su instalación.

RUP es una metodología robusta y define de manera ordenada las tareas a desarrollar. Al utilizar esta metodología queda garantizada la planificación, desarrollo y mantenimiento del sistema. Es aplicable tanto a pequeños proyectos (como el del sistema actual), como para grandes proyectos. (17)

Lenguaje Unificado de Modelado (UML)

Es un lenguaje estándar utilizado para especificar, visualizar y documentar esquemas de sistemas de software orientado a objetos. No es un método de desarrollo, por lo que no sirve para determinar qué hacer en primer lugar o cómo diseñar el sistema, sino que simplemente le ayuda a visualizar el diseño y a hacerlo más accesible para otros. UML está diseñado para su uso con software orientado a objetos y se utilizan para crear diagramas que representen alguna parte o punto de vista del sistema. (18)

1.8 Herramientas y tecnologías de desarrollo utilizadas

A continuación se muestran las herramientas y tecnologías escogidas para la realización del sistema teniendo en cuenta las utilizadas en el departamento.

NetBeans IDE

Es un entorno de desarrollo integrado libre y gratuito, creado específicamente para Java, pero permite la codificación de programas en C, C++ y otros (20). La Plataforma NetBeans es una base modular, lo

que significa que las aplicaciones creadas en el entorno pueden ser extendidas fácilmente por otros desarrolladores de software. Es usado como una estructura de integración para crear aplicaciones de escritorio grandes. Ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación. Entre las características de la plataforma están: (19)

- Administración de las interfaces de usuario (ej. menús y barras de herramientas).
- Administración de las configuraciones del usuario.
- Administración del almacenamiento (guardando y cargando cualquier tipo de dato).
- Administración de ventanas.
- *Framework* basado en asistentes (diálogo paso a paso).

Sus funciones son:

- Editor de código sensible al contenido.
- Con soporte para autocompletar el código, coloreado de etiquetas, auto tabulación y uso de abreviaturas para varios lenguajes de programación.
- Soporte para Java, C, C++, XML y lenguajes HTML.
- Soporte para JSP, XML, RMI, CORBA, JINI, JDBC y tecnologías Servet.
- Incluye CVS (control de versiones) y ANT (compilación avanzada).
- Posibilidad de utilizar otras versiones de compiladores y depuradores.
- Creación visual de componentes gráficos.
- Herramientas con asistentes para facilitar la escritura de código.

Bibliotecas

Para la conexión con dispositivos externos mediante el lenguaje java a través del puerto serie RS 232 existen bibliotecas que proveen de funcionalidades capaces de poder transmitir información entre un equipo y un ordenador. Para poder arribar a una conclusión se realizó un estudio de las bibliotecas Giovynet y javax.com para decidir cuál es la que se ajusta en todos los sentidos a la solución del problema. Otra biblioteca a la cual se le realizará un estudio es a la jFreeChart, la misma facilita la

creación de gráficas en el lenguaje java.

Giovynet

La biblioteca Giovynet provee de métodos para enviar y recibir datos de un dispositivo externo a través de los puertos serial y paralelo de un ordenador. Es utilizada mediante el lenguaje de programación Java. Provee de facilidades al programador para establecer la comunicación con los puertos serial, configurando así los parámetros de conexión y teniendo un control de los datos que son transmitidos por este. Solo puede ser utilizada por el sistema operativo Windows. (20)

Java Communications API

La API Java Communications es una extensión de Java que facilita el desarrollo de aplicaciones independientes de la plataforma de comunicaciones para tecnologías inteligentes. También conocido como javax.com, el cual proporciona acceso a las aplicaciones de hardware RS-232 (puerto serie). Las implementaciones de la API están disponibles para Solaris SPARC, Solaris x86, x86 y Linux. (21)

Java Communications API cuenta con:

- Enumeración de puertos.
- Configuración del puerto (velocidad de transmisión, la velocidad, bits de parada, paridad).
- El acceso a las señales DTR estándar, CD, CTS, RTS y DSR.
- Transferencia de datos a través de puertos RS-232.
- Opciones de control de flujo de hardware y software.
- Control de umbral para recibir *buffer*.
- Opción evento asíncrono para la notificación de:
    - Datos disponibles sobre un puerto RS-232.
    - Puerto de hardware cambios de nivel de línea
    - Cambios de propiedad de puertos dentro de una única JVM.

Luego del estudio realizado de las librerías Giovynet y Com se pudo identificar que para la realización del sistema de adquisición de datos se utilizará la librería Com. Esta provee de diversas funcionalidades y es utilizada en sistemas operativos como Windows y Linux, a diferencia de la

Giovynet que solo es posible ejecutarla en el sistema operativo Windows.

API jFreeChart'S

Esta API facilita el trabajo, proporcionando una amplia gama de gráficos de calidad profesional para la implementación del sistema. JFreeChart'S es consistente y bien documentada, sus gráficos son flexibles por lo que son fáciles de extender. Presenta soporte para muchos tipos de salida, incluyendo componentes Swing, archivos de imagen (incluyendo PNG y JPEG) y gráficos vectoriales de formatos de archivo (incluyendo PDF, EPS y SVG) y es de código abierto. Trabaja con GNU Classpath, una implementación en software libre de la norma estándar de biblioteca de clases para el lenguaje de programación Java. (22)

Visual Paradigm

Es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Tiene soporte para UML 2.x y todos los diagramas asociados. La forma de trabajo es intuitiva, con soporte multiplataforma que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones. 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 las clases.

Permite invertir código fuente de programas, archivos ejecutables y binarios en modelos UML al instante, creando de manera simple toda la documentación. Incorpora el soporte para trabajo en equipo, que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros. Una de las ventajas que tiene su uso es que brinda la posibilidad de intercambiar información mediante la importación y exportación de ficheros con aplicaciones. (23)

PostgreSQL 9.1 como servidor de base de datos

Es un sistema de base de datos relacional de código abierto. Iniciado como un proyecto universitario en la década de 1980, ha llevado varias décadas de desarrollo y cuenta con una arquitectura confiable, estabilidad de procesamiento e integridad en el manejo de datos. Sus principales características son:

- Consultas complejas.
- Claves foráneas.
- Desencadenantes o disparadores.

- Vistas.
- Integridad transaccional.
- Control de concurrencia multiversión.

Además PostgreSQL puede ser ampliado por el usuario en muchos aspectos, por ejemplo, la adición de nuevos:

- Tipos de datos.
- Funciones.
- Operadores.
- Funciones de indexado.
- Métodos de índice.
- Lenguajes procedurales.

Debido a la licencia libre, PostgreSQL puede ser utilizado, modificado y distribuido por todo el mundo de forma gratuita para cualquier propósito, sea comercial, privado o académico. (24)

PgAdmin como aplicación cliente para manejar la base de datos

PgAdmin III es una aplicación gráfica para trabajar con el gestor de bases de datos PostgreSQL. Es la más completa y popular con licencia Open Source. Está escrita en C++. Usa la librería gráfica multiplataforma wxWidgets, lo que permite que se pueda usan en Linux, FreeBSD, Solaris, Mac OS X y Windows. Es capaz de gestionar versiones a partir de la PostgreSQL 7.3 ejecutándose en cualquier plataforma, así como versiones comerciales de PostgreSQL como Pervasive Postgres, EnterpriseDB, Mammoth Replicator y SRA PowerGres.

PgAdmin III está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar *scripts* programados, soporte para el motor de replicación Slony-I y mucho más. La conexión al servidor puede hacerse mediante conexión TCP/IP o *Unix Domain Sockets* (en plataformas Unix), y puede encriptarse mediante SSL para mayor seguridad. (25)

Conclusiones

El el capítulo anterior se abordaron los conceptos fundamentales de la especialidad de anestesiología

y se explicó brevemente cada una de las variables fisiológicas que son transmitidas por las máquinas de anestesia. Se realizó el estudio de los sistemas de adquisición de datos que existen actualmente en el mundo, abordando en cada uno de ellos sus características y dejando claro el por qué de su difícil integración con el sistema alas HIS. En los epígrafes “Protocolos”, “Comunicación serial” y “Manejador de dispositivo” quedaron plasmados los conceptos fundamentales de cada uno de ellos y se puntualizó su necesidad en la elaboración del sistema de adquisición de datos. Otro de los temas abordados y de los cuales se describió su utilización fueron las herramientas y la metodología utilizada.

CAPÍTULO 2. Características del sistema de adquisición de datos

En este capítulo se describen los procesos actuales que se llevan a cabo en un acto quirúrgico cuando se le suministra al paciente fármacos anestésicos. Para tener una representación gráfica del negocio a automatizar, se muestra el modelo de dominio con una breve descripción de los conceptos identificados en el problema. Se especifican los requisitos funcionales y no funcionales del sistema, además de los casos de uso a desarrollar como respuesta al problema actual.

2.1 Modelo de dominio

El modelo de dominio es utilizado por el analista como un medio para comprender las actividades que son objeto de automatización en el sistema. Los objetos o conceptos incluidos en este modelo describen entidades que están vinculadas al problema en cuestión. Cuando se realiza la programación orientada a objetos, el funcionamiento interno del software representa los flujos de actividades de la realidad, por lo que el mapa de conceptos del modelo de dominio constituirá una primera versión del sistema.

2.2 Conceptos fundamentales del dominio

Para brindar una mejor comprensión del diagrama del Modelo de dominio se realiza una breve descripción de los conceptos encontrados en el problema.

Acto quirúrgico: Procedimiento aplicado a un paciente con necesidad quirúrgica durante un espacio de tiempo en un local establecido para este fin.

Paciente: Persona que recibe servicios médicos. Ej: Se somete a una intervención quirúrgica.

Anestesiólogo: Médico especializado en la administración y selección de la anestesia aplicada al paciente, además del control del equipo de anestesia.

Máquina de anestesia: Equipo que a través de la administración de gases como el oxígeno, el óxido nitroso y vapores anestésicos monitoriza todas las funciones requeridas en el paciente y las transmite mediante el puerto de comunicación RS 232.

Puerto serie RS 232: Puerto que permite la comunicación entre la máquina de anestesia y el ordenador. Está compuesto por nueve pines y de ellos los tres más importantes son: transmisión(pin 3), recepción(pin 2) y tierra(pin 5).

Ordenador: Máquina electrónica que se comunica con el equipo de anestesia y procesa los datos que este le envía mediante el puerto serie RS 232.

Interfaz: Sistema intermediario entre la máquina de anestesia y el ordenador. Este decodifica con ayuda del protocolo de comunicación los datos que el equipo transmite y almacena esta información en un registro.

Protocolo: Es el que define el intercambio de datos entre la máquina de anestesia y el ordenador. A través de este se logra interpretar los datos que transmite la máquina de anestesia.

Variables fisiológicas: Funciones en el paciente que monitoriza la máquina de anestesia como:

- VT: Volumen corriente
- VM: Volumen minuto
- PIP: Presión pico
- Media: Presión media
- PEEP: Presión positiva al final de la espiración.
- f: Frecuencia respiratoria
- Alarmas: Alarmas activadas durante la ventilación

Base de datos de alas HIS: Espacio virtual donde se almacena toda la información de un paciente ya decodificada por el sistema.

Diagrama del modelo de dominio

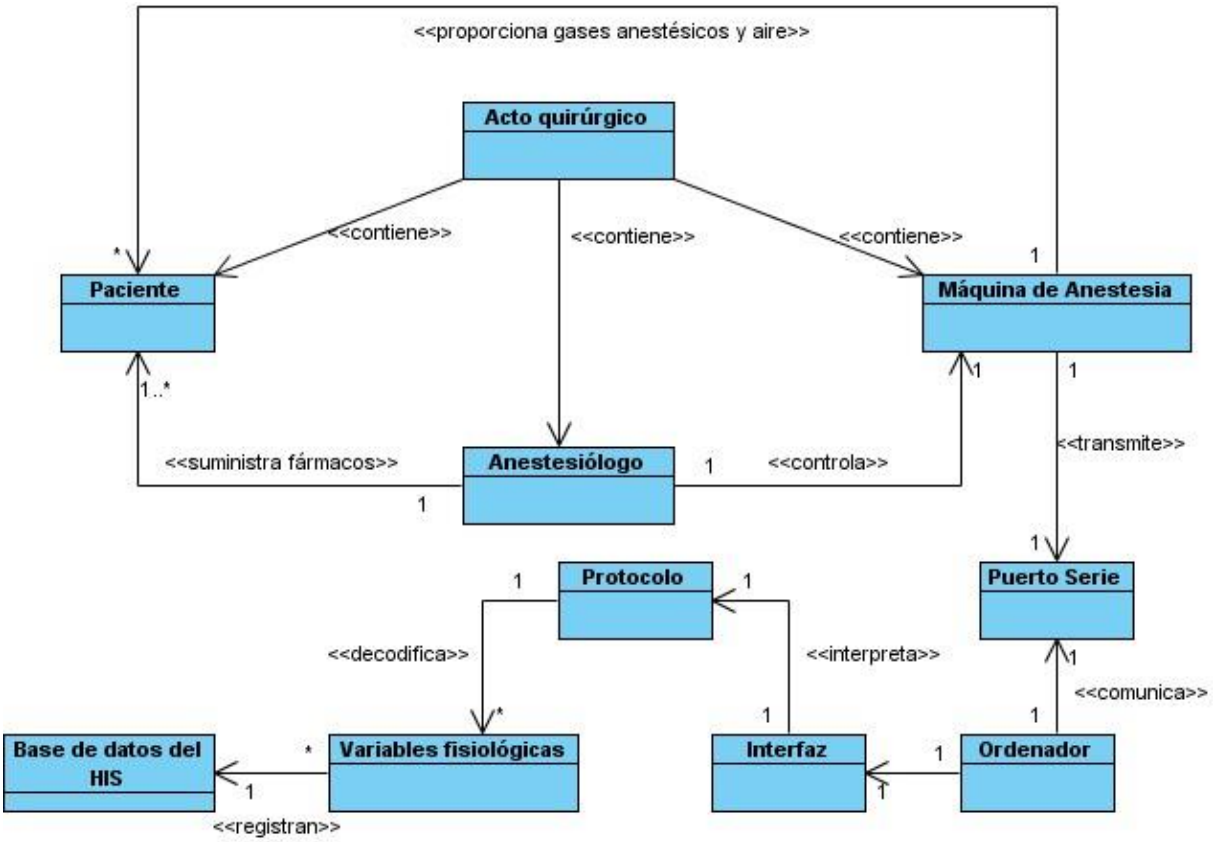

Figura 4: Diagrama del modelo de dominio

La adquisición de variables fisiológicas se obtiene durante la realización de un acto quirúrgico, en el cual coinciden el paciente, el anestesiólogo y la máquina de anestesia. El anestesiólogo le suministra fármacos al paciente según sea necesario y a la vez controla el equipo de anestesia encargado de proporcionar gases anestésicos y aire. Este equipo transmite a través del estandar RS 232 variables fisiológicas monitorizadas desde el paciente, estas son recibidas por un ordenador mediante este medio. El ordenador decodifica las variables fisiológicas que le envió el equipo de anestesia a través de una interfaz que interpreta el protocolo de comunicación Medibus y las registra en la base de datos del sistema alas HIS.

2.3 Propuesta del sistema

La aplicación que se propone tiene como objetivo principal mostrar y almacenar en tiempo real las variables fisiológicas que transmiten las máquinas de anestesia para el procesamiento retrospectivo de

éstas. El sistema permitirá conectar el equipo médico al ordenador donde este se encuentra ejecutándose, el mismo, decodificará y almacenará los datos recibidos para la interpretación de los usuarios. Este garantizará una interfaz intuitiva, además de integridad y confiabilidad de los datos almacenados y mostrados.

Brindará una serie de beneficios a los usuarios entre los que se encuentran: la visualización y el almacenamiento de datos asociados a los pacientes en tiempo real. Lo anterior permitirá a los médicos una mejor comprensión de los indicadores mediante gráficas que describan el comportamiento del paciente durante la cirugía y contribuyendo a evitar posibles errores humanos. Para una mejor comprensión de lo antes expuesto se relacionan en los anexos una serie de imágenes que representan las principales interfaces gráficas de usuario de la **¡Error! No se encuentra el origen de la referencia.**.

Ver anexos de la aplicación(**Imágenes**)

2.4 Especificación de los requisitos del software

A partir de los conceptos relacionados en el modelo de dominio se identificaron las funcionalidades que el sistema debe automatizar para dar solución al problema planteado. Estas funcionalidades conforman los requisitos del software, que son condiciones que necesitan los clientes y debe cumplir el sistema para satisfacer un contrato o un documento formal. El conjunto de todas estas necesidades son la base para el desarrollo del componente o software.

Los requisitos se clasifican en funcionales y no funcionales.

Requisitos funcionales

Describen la funcionalidad o los servicios que se espera que el sistema provea, sus entradas, salidas y excepciones.

Los requisitos funcionales que se determinaron fueron:

	Requisitos funcionales
RF1	Configurar conexión.
RF2	Actualizar configuración de conexión.

RF3	Iniciar registro de variables fisiológicas.
RF4	Seleccionar solicitud de intervención.
RF5	Configurar escalas de la gráfica.
RF6	Guardar imagen de la gráfica.
RF7	Ver información del sistema.

Tabla 7: Requisitos funcionales

Requisitos no funcionales

Los requisitos no funcionales responden a las propiedades del sistema como la fiabilidad, el tiempo de respuesta, la capacidad de almacenamiento, la capacidad de los dispositivos de entrada/salida, y la representación de los datos que se utilizan en las interfaces del sistema.

Los requisitos no funcionales que responden a cada una de las condiciones que deberá cumplir la solución son:

Usabilidad

El sistema deberá estar diseñado de manera que los usuarios obtengan las habilidades necesarias para su utilización en el menor tiempo posible. Este debe indicar al usuario el hilo de acciones a seguir. Además presentará facilidad para acceder a las diferentes funcionalidades mediante íconos que representan la función a realizar. La interfaz será intuitiva y usable.

Seguridad

La aplicación deberá garantizar que si en la transmisión de datos el equipo de anestesia es apagado o desconectado del ordenador, los datos adquiridos hasta el momento deben ser almacenados. Estos deberán ser guardados en una base de datos protegida por contraseña por lo que esta información no deberá ser modificada. Para acceder al sistema los anestesiólogos deberán autenticarse, por lo que los datos no serán accedidos por personal ajeno.

Software

Se especifica las condiciones o capacidades que el sistema debe cumplir. El mismo debe ejecutarse en sistemas operativos Windows XP, Windows Vista, Windows 7 , Ubuntu 11.x, Ubuntu 12.x, Ubuntu 13.x, utiliza la plataforma JAVA 1.7 (Java Virtual Machine), y PostgreSQL 9.1 como gestor de base de datos. En las estaciones de trabajo en caso que se utilice como sistema operativo Windows, es necesario contar con la librería win32com.dll en el directorio C:\WINDOWS y si el sistema operativo es alguna de las distribuciones de Ubuntu antes mencionada las librerías serán libLinuxSerialParallel.so y libLinuxSerialParallel_g.so y se ubicarán en la dirección /usr/lib del sistema.

Requerimientos de hardware

Los requerimientos de hardware estarán dados por la plataforma específica que se utilice para la instalación del sistema, en cuanto a sistema operativo y gestor de bases de datos. Para un funcionamiento óptimo de la aplicación se necesitan estaciones de trabajo con un mínimo de 1GB de memoria RAM y un microprocesador Intel Core-2/Duo o Intel Dual-Core. Para los servidores la solución estará conformada fundamentalmente por alta capacidad de procesamiento y redundancia, que permitan garantizar movilidad, residencia de la información y aplicaciones bajo esquemas seguros y confiables.

Servidores de base de datos: Power Edge, Procesador Intel® Xeon® CPU E7-8837 @ 2.67, 16GB de memoria RAM, 1199GB de disco duro y sistema operativo Linux.

Restricciones de diseño

La capa de presentación contendrá la vista y la lógica de la presentación. La capa del negocio mantendrá los hilos iniciados activos y los procesos del negocio que concurrentemente pueden estar siendo ejecutados. La capa de acceso a datos contendrá las entidades y los objetos de acceso a datos correspondientes a las mismas.

Confiabilidad

Solo serán almacenados los valores de las variables del paciente a la base de datos si estos fueron previamente recibidos sin error en la transmisión y decodificación de los mismos. La aplicación no brindará la posibilidad de poder modificar los datos del paciente en cuestión.

Interfaz de usuario

La interfaz contendrá los datos claros y bien estructurados. Además permitirá la interpretación correcta de la información. Contará con teclas de función y menús desplegables que faciliten y aceleren su utilización. La entrada de datos incorrecta será detectada claramente e informada al usuario.

2.5 Modelo de casos de uso del sistema

El diagrama de casos de uso del sistema se utiliza para especificar la funcionalidad y el comportamiento de un sistema mediante su interacción con los usuarios. En el diagrama intervienen: actores, relaciones y casos de uso. Los actores son entidades externas que se modelan y que interactúan con el sistema. Los casos de uso van representar los requisitos o capacidades internas del sistema, y las relaciones son las encargadas de expresar la interacción entre caso de uso y actor o entre actores o entre casos de uso.

Los actores del sistema identificados son:

Actor	Descripción
Usuario	Usuario global que se encarga de ver la información que tiene el sistema.
Anestesiólogo	Usuario que selecciona la solicitud de intervención que corresponde, inicia o detiene el registro de las variables fisiológicas, configura las escalas de la gráfica de presión en función del tiempo si lo desea y almacena la imagen de la gráfica que el sistema muestra.
Administrador	Usuario encargado de establecer la conexión con la base de datos y con la máquina de anestesia.

Tabla 8: Actores del sistema

2.6 Definición de los actores del sistema

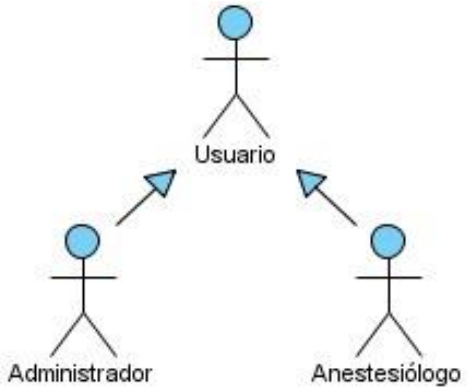

Figura 5: Diagrama de actores

2.7 Diagrama de casos de uso del sistema

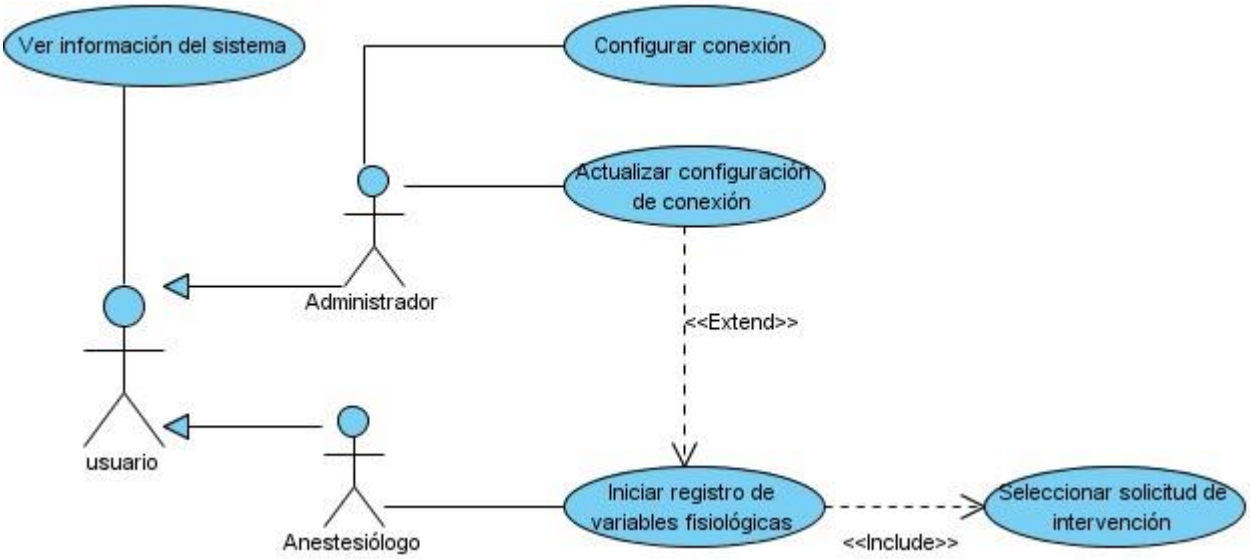

Figura 6: Diagrama de casos de uso del sistema

En el diagrama se encuentran tres actores, el administrador del sistema, encargado de configurar la conexión con la base de datos y con el puerto serie, además de actualizarlas en caso que estas varíen. Otro actor del sistema es el anestesiólogo, encargado de iniciar el registro o detenerlo cuando si es necesario, antes debe haber seleccionado la solicitud de intervención quirúrgica que le corresponde para ese día. Otras de las funcionalidades que este puede realizar son: configurar las escalas de la gráfica y hacer salvas como imágenes de la gráfica en cuestión para un posterior estudio de las mismas. También está presente un usuario global cuyas funciones en la aplicación son: ver información.

2.8 Descripción textual de los casos de uso

CU1	Configurar conexión.
Actor	Administrador
Descripción	El caso de uso inicia cuando el actor accede al sistema por primera vez y tiene que configurar la aplicación para la conexión con la base de datos y con la máquina de anestesia. El sistema muestra los campos necesarios para la conexión con la base de datos y los campos de la conexión con la

	máquina de anestesia se mantienen inhabilitados hasta que no se complete la conexión con la base de datos. El caso de uso termina cuando se muestra un mensaje informando que se ha establecido la conexión.
Referencia	RF1

Tabla 9: Descripción textual del caso de uso: Configurar conexión

CU2	**Actualizar configuración de conexión.**
Actor	Administrador, Anestesiólogo
Descripción	El caso de uso inicia cuando el actor accede al sistema luego de haber configurado los parámetros de la conexión y necesita actualizarlos. El sistema muestra todos los campos para realizar la modificación de estos. El caso de uso termina cuando se muestra un mensaje informando que se ha establecido la conexión.
Referencia	RF3

Tabla 10: Descripción textual del caso de uso: Actualizar configuración de conexión

CU3	**Iniciar registro de variables fisiológicas.**
Actor	Anestesiólogo
Descripción	El caso de uso inicia cuando el actor accede a la opción Iniciar, el sistema comienza a mostrar los datos adquiridos desde la máquina de anestesia, el actor puede configurar las escalas de la gráfica y guardar imagen de la misma y el caso de uso termina.
Referencia	RF2, RF4

Tabla 11: Descripción textual del caso de uso: Iniciar registro de variables fisiológicas

CU4	**Seleccionar solicitud de intervención**
Actor	Anestesiólogo

Descripción	El caso de uso inicia cuando el actor accede a la opción Seleccionar solicitud de intervención, el sistema muestra una interfaz para la búsqueda, el actor introduce el nombre del paciente, el sistema muestra los datos de este, del anestesiólogo y del cirujano que estará presente en la cirugía, brinda la posibilidad de seleccionarlo y el caso de uso termina.
Referencia	RF3

Tabla 12: Descripción textual del caso de uso: Seleccionar solicitud de intervención

CU7	**Ver información del sistema.**
Actor	Anestesiólogo, Administrador.
Descripción	El caso de uso inicia cuando el actor accede a la opción Información del sistema. Se muestra una ventana con toda la información de este y el caso de uso termina.
Referencia	RF7

Tabla 13: Descripción textual del caso de uso: Ver información del sistema

Conclusiones

Al finalizar el capítulo quedaron descritas las funcionalidades que realizará el componente de adquisición de datos desde máquinas de anestesia. Se hace una breve descripción de los principales conceptos que están presentes en el entendimiento del problema. Como resultado de su análisis se define un modelo de dominio donde se representa la interacción entre los conceptos encontrados, además de especificar cada uno de los requisitos funcionales y no funcionales con los que debe contar el sistema. Se define un diagrama de casos de uso del sistema con la ayuda de los requisitos funcionales definidos.

CAPÍTULO 3. Análisis y diseño del sistema de adquisición de datos

En el presente capítulo se realiza una descripción detallada del diseño del sistema. Se presenta el principal patrón de diseño que se ponen en práctica y los diagramas de clases y de secuencia del modelo de diseño.

3.1 Descripción de la arquitectura

La arquitectura de software es la estructuración del sistema que se debe crear en etapas tempranas de desarrollo del software. Esta consiste en un conjunto de patrones que serán capaces de aportar elementos idóneos para tomar buenas decisiones. Además, proporcionará conceptos y un lenguaje común, posibilitando la comunicación entre los equipos que participen en un proyecto. (26)

Teniendo en cuenta las tecnologías, herramientas y metodologías expuestas anteriormente, se define como parte de la línea base de la arquitectura la implementación del patrón de diseño Modelo Vista Controlador. Este permite realizar la programación multicapa, separando los datos de la aplicación, la interfaz de usuario y la lógica de control, en tres componentes distintos. (27)

Donde el modelo representa la información con la que trabaja la aplicación, o sea, los datos de esta. La vista le permite al usuario interactuar con la aplicación así como mostrar la información procesada al mismo y el controlador representa la lógica, el cual es el encargado de procesar las interacciones del usuario con el sistema.

La utilización del patrón de diseño Modelo Vista Controlador posibilita a la aplicación numerosos beneficios. Entre los que se encuentran agregar nuevas vistas sin necesidad de alterar la lógica del negocio, modificar los objetos, ya sea para cambiar la forma en que se encuentran o para migrar a otra tecnología. Este patrón brinda una mejor organización en el desarrollo de la aplicación, además de ahorrar tiempo en el mantenimiento del sistema.

La capa de presentación está compuesta por páginas java, donde estas están conformadas por formularios que mediante componentes UI Java Swing validan y obtienen los datos que el usuario facilita al sistema en cada función que este realiza. La página principal del sistema utiliza además la librería jFreeChart para el diseño de la gráfica que en ella se muestra. La capa de negocio está constituida por las clases controladoras, las cuales se encargan de definir las reglas y lógica del negocio. Estas utilizan a la librería javax.comm para la conexión con el puerto de comunicación RS

232. La capa de datos gestiona el acceso a los datos de la aplicación. En esta se hace uso de la persistencia de objetos, que permite vincular los objetos de la base de datos a objetos de lenguajes de programación Java, para aumentar el nivel de abstracción y facilitar el acceso a los datos desde la capa de negocio. (28)

3.2 Integración con el sistema alas HIS

El sistema de adquisición de datos transmitidos por máquinas de anestesia constituye un componente para el sistema alas HIS. Este se encarga de adquirir las variables fisiológicas que transmite el equipo de anestesia y registrarlas en la base de datos del HIS. Una vez almacenadas las variables, el módulo Bloque Quirúrgico tiene la posisblidad de leerlas y luego mostrarlas en las tablas que ya se encuentran diseñadas en el sistema para una posterior visualización del comportamiento del paciente bajo los efectos de la anestesia.

El sistema está diseñado para atender las solicitudes de intervenciones quirúrgicas planificadas en el sistema alas HIS, adquiriendo, a partir de estas, los datos asociados al paciente, anestesiólogo y cirujano que estarán presente en la cirugía.

3.3 Modelo de diseño

El modelo de diseño es una representación de la solución que se va a construir. Se realiza basándose en los requisitos del cliente, y al mismo tiempo la calidad se puede evaluar y comparar con el conjunto de criterios predefinidos para obtener un buen diseño. En el contexto de la ingeniería del software el diseño se centra en cuatro áreas importantes de interés: datos, arquitectura, interfaces y componentes. (29)

3.4 Patrones de diseño

Los patrones de diseño ofrecen una solución a problemas similares de desarrollo de software. Para su utilización debemos tener presente cuándo y dónde utilizarlos, estos logran ahorrar tiempo y mejorar el sistema, haciéndolo más eficaz, dinámico y seguro.

Para la realización del diseño de la aplicación otro de los patrones que se utilizaron fueron los llamados patrones *GRASP*, los cuales describen los principios fundamentales del diseño de objetos para la asignación de responsabilidades. A las clases del sistema se le asignaron tareas que podían

realizar según la información que poseían, así como la creación de objetos (instancias) de otras clases, haciéndose notar los patrones experto y creador. (30)

Otros patrones utilizados son: Bajo acoplamiento y Alta cohesión. Su aplicación se pone de manifiesto donde la asignación de responsabilidades se realiza permitiendo la colaboración entre las clases, sin afectar el proceso de reutilización. Otra muestra de su utilización es en las clases controladoras, donde las operaciones del sistema que ahí se implementan son factibles de manejar en otras capas.

Con el uso del gestor de base de datos *PostgreSQL,* versión 9.1, se realiza la recuperación y el almacenamiento físico de los datos. Se logra una abstracción del gestor de base de datos y la persistencia de las entidades del sistema obtenidas mediante el proceso de mapeo. (24)

3.5 Diagramas de clases del diseño

El diseño de software tiene una representación significativa en el desarrollo de aplicaciones informáticas. Este posibilita la producción de disímiles modelos del sistema o productos que se deseen diseñar. En este flujo de trabajo se modela el sistema, de manera que involucre todos los requisitos, tanto no funcionales como funcionales. Al mismo tiempo se va definiendo la arquitectura más factible a utilizar. El modelo obtenido unido a las clases del diseño y objetos que lo conforman dan lugar a los casos de uso, mediante los que se produce el diagrama de clases del diseño.

En los diagramas de clases de diseño se expone un conjunto de interfaces, colaboraciones y sus relaciones. Se utilizan para modelar la vista de diseño estática de un sistema. Estos son de gran importancia porque permiten visualizar, especificar y documentar modelos estructurales. Los diagramas de clases de diseño forman parte de las realizaciones de casos de usos.

Para la realización de los casos de uso los diagramas de interacción constituyen un elemento fundamental, estos muestran cómo interactúan los objetos y las relaciones entre ellos. Los diagramas de interacción están conformados por los diagramas de secuencia y los diagramas de colaboración, donde generalmente se recomienda utilizar los de colaboración en el análisis y el de secuencia para el diseño. Los de secuencia enfatizan el orden de tiempo de los mensajes.

Para la construcción del modelo de diseño, se establece una estructura de paquetes dividida en fragmentos manejables para su posterior implementación. Existe una relación entre los paquetes mediante los que se establecen dependencias entre las distintas clases que lo contienen. El proceso de empaquetado se realiza teniendo en cuenta un solo criterio y es el referente a los tipos de entidades que se pueden encontrar.

El diagrama de paquetes es una representación de cómo está estructurado el sistema y sus dependencias. Un paquete está conformado por subpaquetes que responden a las realizaciones de los casos de uso, donde cada uno de los subpaquetes contiene un diagrama de clases del diseño y un diagrama de secuencia. (31)

Los paquetes se grafican según la relación que guardan entre ellos y utilizan el paquete de repositorio de clases para su funcionamiento. Este está compuesto por las clases del diseño y posee dos paquetes: entidades y sesiones. El primero contiene el paquete de las entidades autogeneradas y el de las personalizadas. Donde las autogeneradas engloban a las que se generan desde la base de datos mediante el proceso de mapeo, y las personalizadas no se generan y permiten modificación, por lo que pueden heredar de las entidades autogeneradas. El segundo tiene el paquete de las controladoras personalizadas.

A continuación se representa el diagrama de paquetes del sistema:

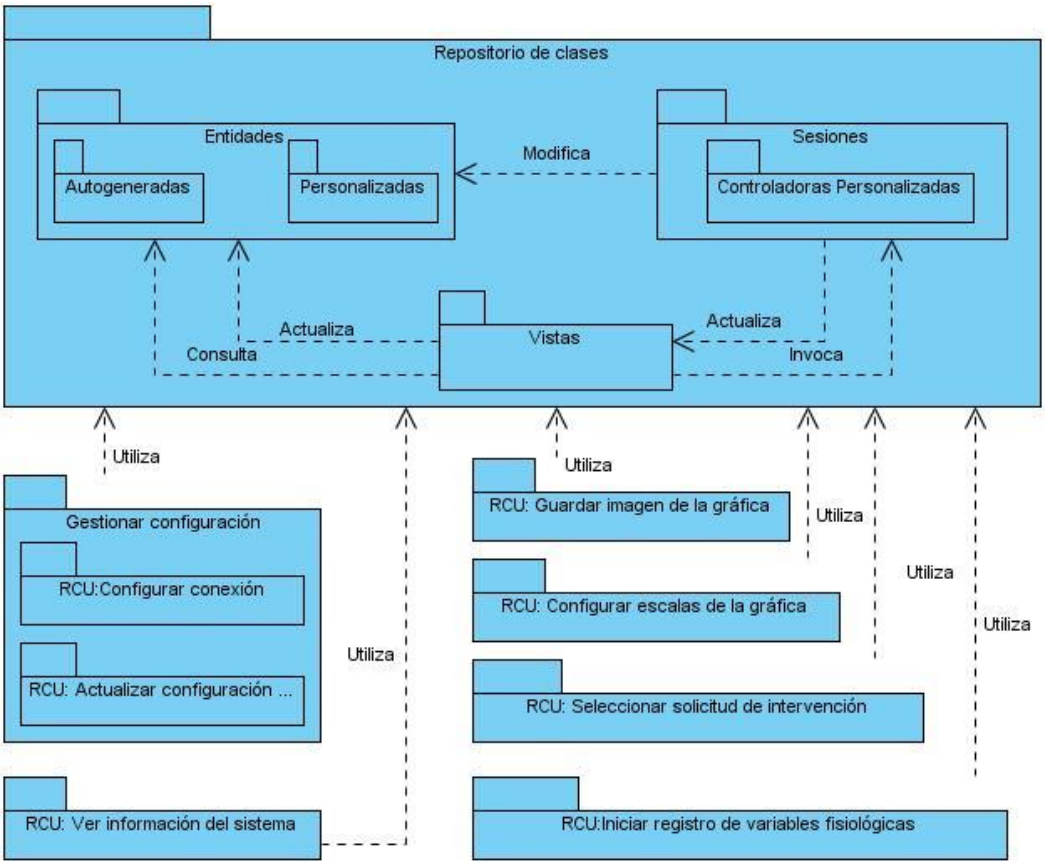

Figura 7: Diagrama de paquetes

En la representación del diagrama de paquetes se muestran los procesos Gestionar configuración y Configurar registro. Para cada uno de estos procesos se modela un diagrama de clases del diseño y por cada escenario del diagrama mencionado anteriormente se modela un diagrama de interacción. Donde para este último se seleccionó el diagrama de secuencia.

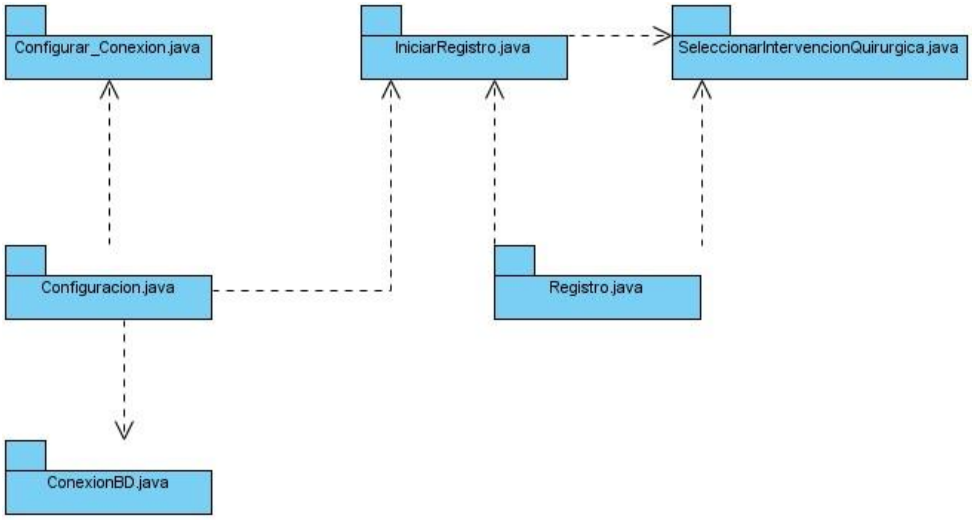

Figura 8: Diagrama de clases del diseño

El sistema está compuesto por dos clases controladoras Configuración y Registro. La clase configuración actualiza a las interfaces *Configurar_Conexion e IniciarRegistro*. Esta controladora utiliza a la clase de acceso a datos ConexionBD para establecer la conexión con la base de datos así como realizar las operaciones de búsqueda y actualización en la misma. La clase controladora Registro es la que gestiona todo el negocio en las interfaces *IniciarRegistro* y *SeleccionarIntervencionQuirurgica*.

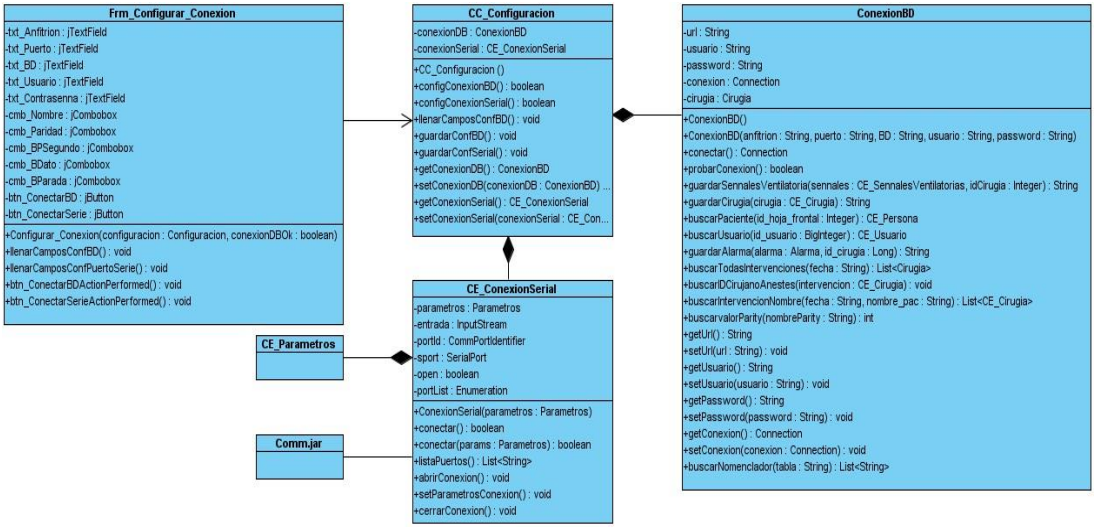

Figura 9: Diagrama de clases del diseño: Configurar conexión

En el **Anexo 1** se presentan los demás diagramas de clases del diseño del sistema.

3.6 Diagrama de secuencia

Los diagramas de secuencia como ya se mencionó anteriormente son parte de los diagramas de interacción. Estos muestran los objetos como líneas de vida a lo largo de la página con sus interacciones, estas representadas por mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida destino. Los diagramas de secuencia son buenos para mostrar qué objetos se comunican con otros objetos y qué mensajes emiten esas comunicaciones. (32)

A continuación se representa un diagrama de secuencia del sistema:

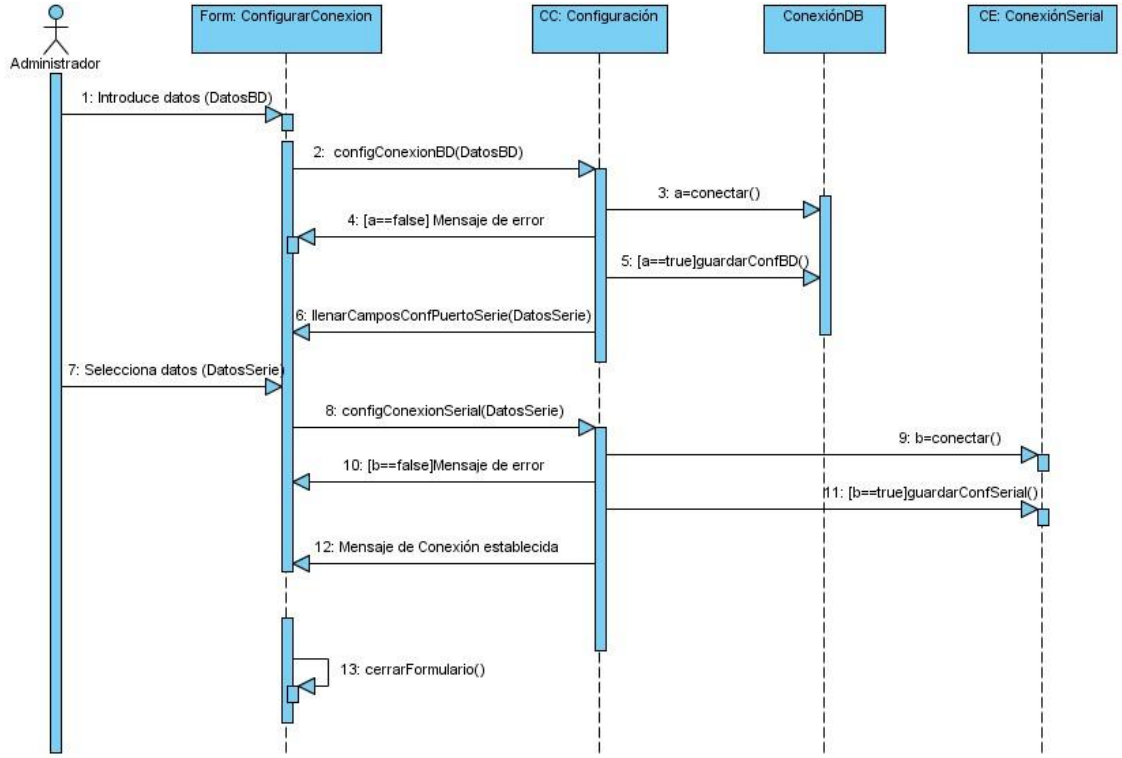

Figura 10: Diagrama de secuencia: Configurar conexión

Las clases de diseño están conformadas por:

Formularios: Son plantillas diseñadas con el propósito de que el usuario introduzca datos estructurados. Están formadas por componentes Swing los cuales pueden ser insertados desde la paleta de componentes o mediante código. El diseño se realiza con el fin de que el usuario introduzca o visualice datos (nombre, apellidos, sexo, etc.) para ser almacenadas y procesadas posteriormente.

Controladoras*:* Las clases controladoras implementan la lógica del negocio que se está informatizando, generalmente cada una de estas se encargan de la implementación de un caso o de varios casos de uso en dependencia de la complejidad de los mismos.

Entidades: Las clases entidades son todo lo relacionado con los datos que integra la aplicación.

Clase de acceso a datos: Es la clase encargada de realizar las consultas a la base de datos y de almacenarlos.

3.7 Descripción de las clases

La descripción de las clases del sistema ofrece información acerca de la interacción de las clases involucradas en la realización de los casos de uso. A continuación se muestra las descripciones de las clases controladoras del sistema.

Nombre: Configuracion	
Tipo de clase: Controladora	
Atributo	**Tipo**
conexionDB	ConexionDB
conexionSerial	ConexionSerial
puert	String
bd	String
user	String
pass	String
Para cada responsabilidad:	
Nombre:	Configuracion()
Descripción:	Constructor de la clase.
Nombre:	configConexionBD()
Descripción:	Lee el archivo de la configuración con la base de datos y verifica que se pueda establecer la conexión.
Nombre:	configConexionSerial()
Descripción:	Lee el archivo de la configuración con el puerto serie y verifica que se pueda establecer la conexión.

Nombre:	llenarCamposConfBD()
Descripción:	Muestra los datos en los componentes del visual para iniciar la configuración.
Nombre:	guardarConfBD()
Descripción:	Guarda la configuración con la base de datos.
Nombre:	guardarConfSerial()
Descripción:	Guarda la configuración con el puerto serie.

Tabla 14: Descripción de la clase controladora: Configuracion

Nombre: Registro	
Tipo de clase: Controladora	
Atributo	**Tipo**
datosCirugia	Cirugia
directorio	String
hiloprotocolo	Protocolo
grafico	Grafico
Para cada responsabilidad:	
Nombre:	Registro(Cirugia datosCirugia)
Descripción:	Constructor de la clase.
Nombre:	iniciarLectura()
Descripción:	Llama al método start() de la clase entidad protocolo el cual leerá los datos que envía la máquina de anestesia.

Tabla 15: Descripción de la clase controladora: Registro

Nombre: ConexionBD	
Tipo de clase: Acceso a datos	
Atributo	**Tipo**
url	String
usuario	String
password	String
conexion	Connection
cirugia	Cirugia
Para cada responsabilidad:	
Nombre:	ConexionBD()
Descripción:	Constructor de la clase.
Nombre:	ConexionBD(anfitrión: String, puerto:String, BD:String, usuario:String, password:String)
Descripción:	Constructor de la clase. Inicializa las variables “url”, “usuario” y “password” con los pasados por parámetros. La variable url estará conformada por el anfitrión, puerto y BD.
Nombre:	Conectar()
Descripción:	Establece la conexión con la base de datos a partir de los datos de “url, usuario y password”. Inicializa el objeto conexión.
Nombre:	probarConexion()
Descripción:	Método auxiliar utilizado por el anterior para verificar si es posible establecer la conexión. Retorna “true” si es posible y “false ” si no fue

	posible.
Nombre:	guardarSennalesVentilatorias(sennales:SennalesVentilatorias, idCirugia:Integer)
Descripción:	Guarda en la base de datos las Señales Ventilatorias pasadas por parámetro.
Nombre:	guardarCirugia(cirugía: Cirugia)
Descripción:	Guarda en la base de datos la Cirugia pasada por parámetro.
Nombre:	buscarPaciente(id_hoja_frontal:Integer)
Descripción:	Busca en la base de datos un paciente cuyo id sea el pasado por parámetro.
Nombre:	buscarUsuario(id_usuario : BigIntege)
Descripción:	Busca un usuario en la base de datos por el id pasado por parámetro.
Nombre:	guardarAlarma(alarma:Alarma, idcirugia:Long)
Descripción:	Guarda en la base de datos la Alarma pasada por parámetro.
Nombre:	buscarTodasIntervenciones(fecha:String)
Descripción:	Busca en la base de datos todas las solicitudes de intervención planificadas para la fecha pasada por parámetro.
Nombre:	buscarIDCirujanoAnest(intervención:Cirugia)
Descripción:	Busca en la base de datos los id del cirujano y el anestesiólogo que estarán presentes en la cirugía pasada por parámetro.
Nombre:	buscarIntervencionNombre(fecha:String, nombre_pac:string)
Descripción:	Busca en la base de datos de las solicitudes de intervención para el paciente y la fecha pasados por parámetro
Nombre:	buscarValorParity(nombreParity:String)

Descripción:	Busca en la base de datos el valor asociado al nombre de la paridad pasada por parámetro.

Tabla 16: Descripción de la clase de acceso a datos: ConexionBD

Conclusiones

En el capítulo se puntualizó la arquitectura a utilizar por todos los beneficios descritos anteriormente que le brinda al sistema. Además se logró dar una mejor organización a la aplicación con la construcción del diagrama de paquetes. Se realizaron todos los diagramas de diseño y de interacción de cada una de las funcionalidades del sistema para brindar al desarrollador una idea más clara de lo que se debe implementar.

CAPÍTULO 4: Características de la implementación del sistema de adquisición de datos

Este capítulo contiene las principales características de la implementación del sistema. Muestra los principales componentes y sus relaciones, haciendo uso del diagrama de componentes. Además se modela el diagrama de despliegue, artefacto correspondiente al flujo de trabajo implementación. Se exponen las estrategias de seguridad, los estándares de codificación así como los tratamientos de errores.

4.1 Modelo de datos

El modelo de datos aporta la base conceptual para el desarrollo de la aplicación y la decisión de cómo se almacenarán los datos y cómo se accederá a ellos. Este determina la estructura de la información con el propósito de mejorar la comunicación y la precisión en aplicaciones que usan e intercambian datos. (33)

En general, un modelo de datos representa la estructura física de las tablas de la base de datos y es esencial para el desarrollo de una aplicación. Pues a través de él se puede conseguir la compatibilidad necesaria para manejar grandes cantidades de datos.

El modelo de datos de la aplicación que a continuación se representa es un modelo E-R (entidad-relación). Las entidades representan objetos de los que el sistema necesita guardar información, y las relaciones son la forma en que dos objetos o entidades se relacionan. Otro de los elementos por lo que está compuesto el modelo de datos es por atributos. Estos últimos pueden tener diferentes clasificaciones y constituyen características asociadas a una entidad.

A continuación se muestra el modelo de datos de la aplicación:

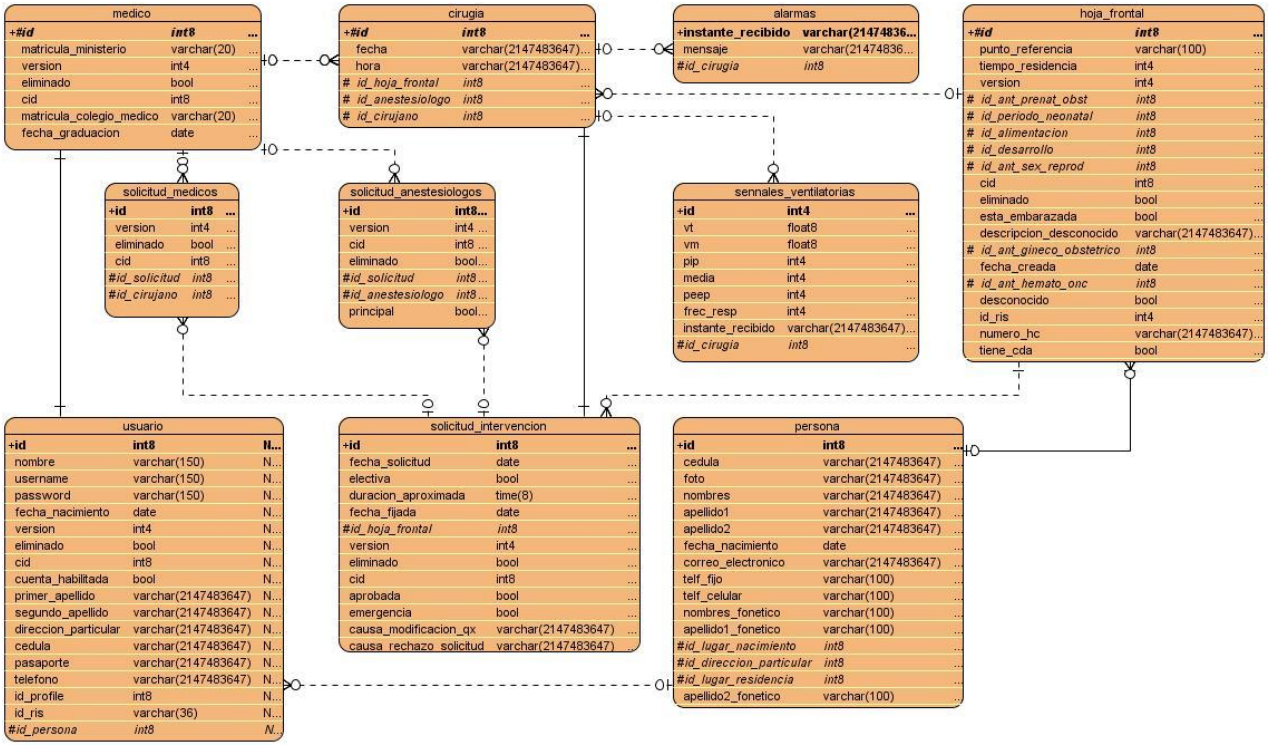

Figura 11: Modelo de datos

4.2 Descripción de las tablas de la base de datos

A continuación se describe el atributo común de las entidades. Esto se realizó para facilitar el trabajo de implementación de algunas funcionalidades.

Atributo	Tipo de Dato	Descripción
Id	integer	Identificador de las entidades para las referencias en las relaciones entre las tablas.

Tabla 17: Descripción del atributo común de todas las tablas

Cirugía		
Entidad que recoge los datos de la cirugía, contiene el id de la hoja frontal, del cirujano y del anestesiólogo encargado de la cirugía, además de la fecha y la hora en que se realizará la misma.		
Atributo	**Tipo**	**Descripción**

fecha	varchar	Fecha en la que se realiza la cirugía.
hora	varchar	Hora en la que se comienza la cirugía.
id_cirujano	integer	I Identificador de la entidad médico de la especialidad de cirugía.
id_hoja_frontal	integer	Identificador de la hoja frontal.
id_anestesiólogo	integer	Identificador de la entidad médico de la especialidad de anestesiología.

Tabla 18: Descripción de la tabla: cirugía

Alarmas		
Entidad que recoge el mensaje que envía la alarma de la máquina de anestesia, el instante en que fue enviado y en que cirugía se está efectuando el evento.		
Atributo	**Tipo**	**Descripción**
instante_recibido	varchar	Momento en que es recibida la alarma desde el equipo.
mensaje	varchar	Mensaje que envía la máquina de anestesia.
id_cirugia	integer	Identificador de la cirugía que se está llevando a cabo.

Tabla 19: Descripción de la tabla: alarma

sennales_ventilatorias		
Entidad que recoge las señales ventilatorias del paciente que se encuentra conectado a la máquina de anestesia.		
Atributo	**Tipo**	**Descripción**
VT	float	Volumen corriente.
VM	float	Volumen minuto.

PIP	integer	Presión pico
MEDIA	integer	Presión media.
PEEP	integer	Presión positiva al final de la espiración.
frec_resp	integer	Frecuencia respiratoria
instante_recibido	varchar	Momento en el que se comienzan a recibir las señales.
id_cirugía	integer	Identificador de la cirugía que se está llevando acabo.

Tabla 20: Descripción de la tabla: sennales_ventilatorias

Otra de las entidades presentes son las llamadas nomencladores, las cuales representan los valores posibles que puede tomar un campo. En la implementación estas entidades tienen los siguientes campos:

Nomenclador		
Atributo	**Tipo de Dato**	**Descripción**
valor	integer	Valor que puede tomar un campo. Ejemplo: la velocidad de transmisión.
nombre	varchar	Este solamente es utilizado en el nomenclador de paridad. Es definido para que el usuario comprenda los valores que puede tomar.

Tabla 21: Descripción de la tabla de nomencladores

4.3 Modelo de implementación

El modelo de implementación representa mediante descripciones cómo los elementos del modelo del diseño se implementan en términos de componentes y cómo estos se organizan de acuerdo a los nodos específicos en el modelo de despliegue. Este flujo de trabajo tiene como propósito definir la organización del código e implementar las clases y subsistemas encontrados durante el diseño. (34)

Diagrama de componentes

Este diagrama describe la estructura de los componentes del sistema agrupados por paquetes lógicos y muestra las dependencias entre estos componentes. El mimo es utilizado para modelar la vista estática y dinámica de un sistema, además de mostrar organización entre sus componentes. (35)

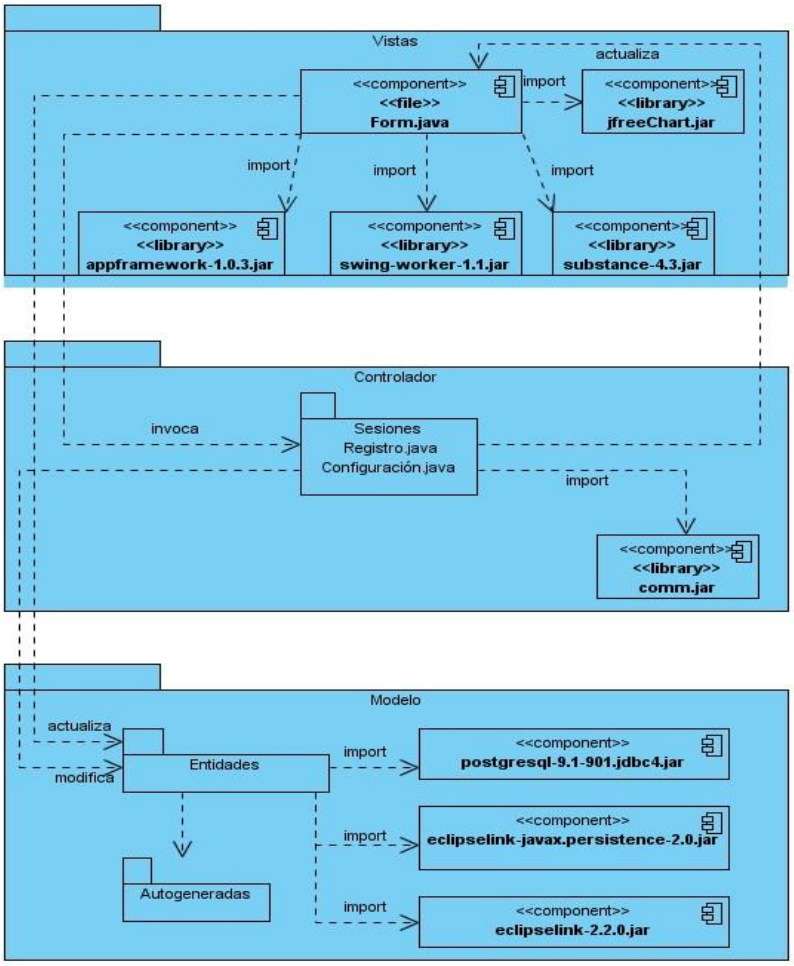

Figura 12: Diagrama de componentes

En el diagrama se muestra como está estructurado el sistema en componentes. Las vistas de la aplicación son formularios con extensión .java que importan las librerías mostradas para hacer uso de ellas. Estas invocan a las sesiones y actualizan las entidades. Las sesiones al igual que las vistas utilizan a la librería que se muestra, modifican las entidades y actualizan las vistas. En el modelo se encuentran las entidades personalizadas y las autogeneradas, estas importan a las librerías que se muestran para utilizarlas cuando estimen conveniente.

Diagrama de despliegue

Muestra las relaciones físicas de los distintos nodos que componen el sistema y el reparto de los componentes sobre dichos nodos. Este representa el complemento del diagrama de componentes y la unión de ellos provee la vista de implementación del sistema. El diagrama de despliegue representa a los nodos y sus relaciones. Los nodos son conectados por asociaciones de comunicación tales como enlaces de red y conexiones TCP/IP.

Con los nodos se modela la topología del hardware sobre el que se ejecuta el sistema. Estos pueden representar un elemento de hardware o de software. (36)

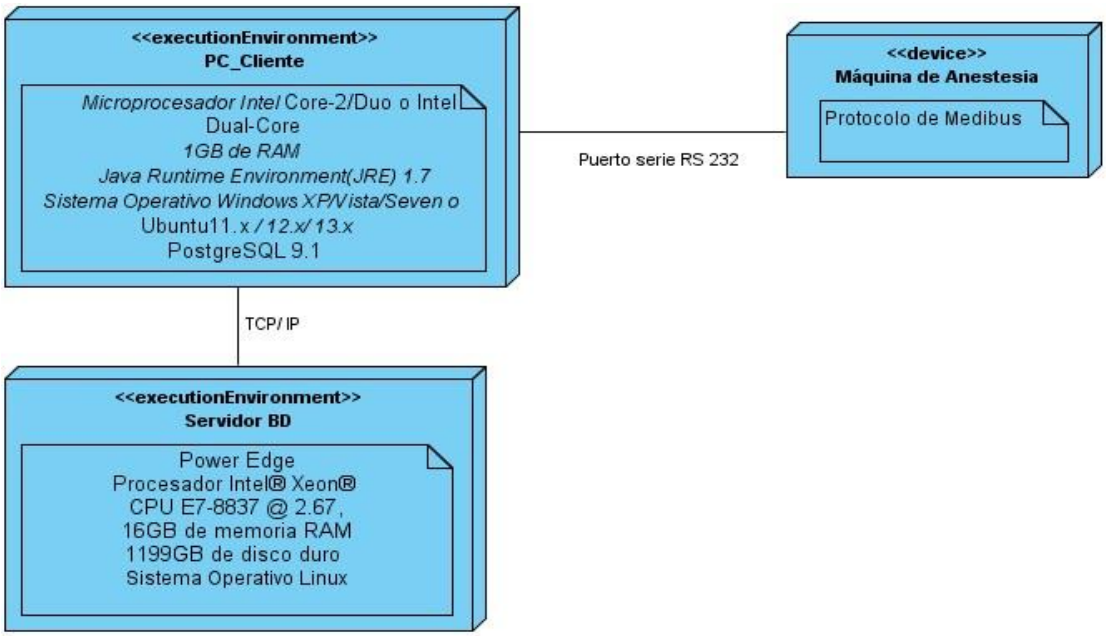

Figura 13: Diagrama de despliegue

4.4 Tratamiento de errores

Los sistemas informáticos durante su ejecución pueden producir fallas en las diferentes rutinas, a estas fallas en el sistema se les conoce por el nombre de excepciones o errores. Mediante el tratamiento de las excepciones se restaura un estado en que las rutinas pueden continuar su ejecución, lo que permite alcanzar un sistema fiable y seguro. (37)

El sistema propuesto para el tratamiento de excepciones propone un control en todas las regiones del código, especialmente donde los datos son insertados o modificados en la base de datos. También re requiere controlar las excepciones cuando los datos son insertados por los usuarios mediante los formularios y es necesario una validación de estos para su correcto almacenamiento y procesamiento por el sistema.

Para el tratamiento de las excepciones o errores se hará uso de bloques de código *try* y *catch*. Donde el bloque *try* es para detectar cuando ocurra algún fallo en el sistema y el bloque *catch* manejará dichas excepciones mediante mensajes que se muestran en las interfaces utilizando el método *showMessageDialog* de Java.

4.5 Seguridad

Todos los sistemas informáticos cuyo objetivo principal sea brindar servicios a pacientes deben garantizar seguridad y confidencialidad de la información que se gestiona. Para llevar a cabo este trabajo debe tenerse un control que garantice que los datos no serán accedidos o modificados por personas ajenas a la información que se procesa.

La aplicación garantizará en la transmisión de datos, en caso que el equipo de anestesia sea apagado o desconectado del ordenador, que los datos adquiridos hasta el momento sean almacenados. Estos serán guardados en una base de datos protegida por contraseña por lo que esta información no será modificada por personas ajenas si no se conoce la misma. El único rol con acceso a la información es el administrador del sistema.

4.6 Estratégias de codificación. Estándares y estilos a utilizar

Un estándar de codificación son reglas que se siguen para la escritura del código fuente. De tal manera que a otros programadores se les facilite entender dicho código. Estos estándares evitan los posibles errores que pueden existir para entender o de escribir algún código.

El uso de estilos de codificación ayuda a la calidad del software por lo que es de gran importancia su puesta en práctica en proyectos informáticos. Para el sistema propuesto se utilizó la notación Camello en la denotación de las variables, parámetros y métodos. (38)

4.7 Variables, parámetros y métodos

Estos elementos se definen escribiendo con minúscula la primera palabra y en caso de que la variable, parámetro o método contenga una segunda palabra, esta se escribirá con mayúscula la primera letra.

Ejemplo: string nombrePaciente; public void registrar()

Empleo de márgenes en el código

Los márgenes se utilizarán de dos espacios por bloque de código. El inicio "{"estará alineado a la declaración del método y el cierre "}" estará alineado debajo de la declaración a la que pertenece. Se dejará un espacio en blanco desde la instrucción anterior para el inicio y fin de bloque. Lo mismo sucede para el caso de las instrucciones *if, else, for, while, do while,.*

Comentarios, separadores, líneas, espacios en blanco y márgenes

Los comentarios se ubicarán al inicio de cada clase o método, resumiendo el objetivo de estos, así como el significado de los parámetros que utiliza.

Se dejará una línea en blanco antes y después de la declaración de cada método, de cada clase y luego de definir los atributos que conforman una clase.

Entre operadores habrá un espacio en blanco, esto se llevará a cabo para una mejor comprensión del código.

Conclusiones

En este capítulo se realizó la implementación del sistema en términos de componentes, cumpliendo con los requerimientos de seguridad, el tratamiento de errores expuesto anteriormente y las estratégias de codificación. Por lo que se proporciona una solución a los requisitos funcionales y no funcionales especificados en el Capítulo 2 del documento. Para ello se realizó la descripción de la estructura de tablas sobre las que se establece el modelo de datos del sistema, así como sus atributos y relaciones. Además se estructuran las clases de diseño en paquetes y subsistemas de implementación.

Conclusiones

Con el desarrollo del sistema se concluye lo siguiente:

1. La caracterización realizada sobre los sistemas de adquisición de datos estudiados evidenció limitantes para su integración con el sistema alas HIS.
2. La revisión bibliográfica de los sistemas de adquisición de datos existentes permitió identificar sus ventajas e incorporar buenas prácticas asociadas a sus características en la aplicación desarrollada.
3. El almacenamiento de las gráficas de tendencia obtenidas por la herramienta desarrollada a partir de los datos adquiridos, constituye una fuente de conocimientos para la investigación y la docencia, así como un soporte válido en la solución de casos médico legales.
4. La obtención de datos de manera automática reduce errores humanos en la introducción y lectura de los indicadores del paciente.
5. La implementación de las especificaciones correspondientes al protocolo de comunicación Medibus, constituye un componente que permite establecer la comunicación con máquinas de anestesia fabricadas para dicho protocolo.

Recomendaciones

Para una posterior versión se recomienda:

1. Implementar la decodificación de las alarmas que transmite la máquina de anestesia en otros idiomas como alemán, inglés, francés, entre otros.
2. Incorporar al sistema la decodificación de las variables fisiológicas que transmiten las máquinas de anestesia por otros protocolos de comunicación como el Vitalink y el Modbus.
3. Diseñar una base de datos en la cual el sistema almacene la información de forma tal que cualquier sistema de información hospitalaria pueda acceder a ella.

Bibliografía

1. **(UMMC), University of Maryland Medical Center.** Anestesia. *Anestesia.* [En línea] 2011. http://www.umm.edu/esp_ency/article/007280.htm.

2. **Pastrana, Jose Manuel.** Cirugia Estética Cancun. *Cirugia Estética Cancun.* [En línea] diciembre de 2012. [Citado el: 15 de enero de 2013.] http://cirugiaesteticacancun.com/?page_id=204.

3. **Montero, Barrante y Quirós.** *Introducción a los sistemas de control supervisor y de adquisición de datos (SCADA).* Costa Rica : s.n., 2004.

4. *Lectura y decodificación para el sistema alas HIS de datos transmitidos por máquinas de anestesia.* **González Martínez, Yoandy.** La Habana, Cuba : s.n., 2013.

5. **Fisher, Tim.** Driver. *Driver.* [En línea] 2012. [Citado el: 10 de 12 de 2012.] http://pcsupport.about.com/od/termsag/g/term_driver.htm.

6. **CLASA, Comite de neuroanestesiología de.** Neuroanestesiología. *Neuroanestesiología.* [En línea] 2005. http://neuroanestesiologia.mx/softw/softw.htm.

7. *REGANE: adquisición normalizada de señales de una máquina.* **Thevenet D, Simini F.** Argentina : s.n., 2012.

8. **Sterle, Adam .** Roche Diagnostics. *Roche Diagnostics.* [En línea] 6 de 2011. [Citado el: 12 de enero de 2013.] http://www.captodayonline.com/.

9. **Drager.** Drager. *Drager.* [En línea] Drägerwerk AG & Co. KGaA, 2012. http://www.draeger.com/ES/es/products/it_software/mon_innovian_solution_suite.jsp.

10. *Driver EPP (Encriptado Pin Pad).* **Medina, Gastón Nicolás.** Córdoba : s.n., 2003.

11. **Ourense, Campus .** *"SOFTWARE DE COMUNICACIÓN ENTRE 2 PC's A.* Vigo : s.n., 2007.

12. **Stallings, William.** Comunicaciones y redes de datos. *Comunicaciones y redes de datos.* [En línea] 2003. http://es.scribd.com/doc/30097309/Comunicaciones-y-Redes-de-Datos-William-Stallings.

13. **Bartolome, Jordi.** El protocolo MODBUS. *El protocolo MODBUS.* [En línea] enero de 2011. [Citado el: 12 de enero de 2013.] http://www.tolaemon.com/site/protocolo_modbus.

14. **Drager.** DragerVitalink. *DragerVitalink.* [En línea] mayo de 2006. http://www.capsuletech.com/Docs/DDIhelps/DragerVitalink.htm.

15. **GS, Fabius.** DagerMedical. [En línea] abril de 2006. http://www.draeger.com.

16. **Oracle.** Java. *Java.* [En línea] oracle. [Citado el: 15 de 11 de 2012.] http://www.java.com/es/download/faq/whatis_java.xml.

17. **Acuña, Karenny Brito.** *SELECCIÓN DE METODOLOGÍAS DE DESARROLLO PARA APLICACIONES WEB EN LA FACULTAD DE INFORMÁTICA DE LA UNIVERSIDAD DE CIENFUEGOS.* 2011.

18. **Rodriguez, Mylena.** *LENGUAJE UNIFICADO DE MODELAMIENTO.* ecuador : s.n.

19. **Pedro, David Cuesta y Peris, Amores Salva.** Netbeans. *Netbeans.* [En línea] 2009. http://www.slideshare.net/e1da4/netbeans-2164338.

20. **Cediel, Giovanny Rey.** *Giovynet.* 2012.

21. **Oracle.** Oracle Technology Network. *Oracle Technology Network.* [En línea] Oracle Support. [Citado el: 15 de 12 de 2012.] http://www.oracle.com/technetwork/java/index-jsp-141752.html.

22. **Management:, Simba.** cnet. *cnet.* [En línea] 2012. [Citado el: 9 de 5 de 2013.] http://es.download.cnet.com/JFreeChart/3000-2213_4-10103763.html%5D.

23. **Pressman, Roger S.** *Ingeniería de Software: un enfoque práctico, 5ta Edición.* Nueva York, E.U.A : McGraw-Hill, 2005.

24. PostgreSQL. *PostgreSQL.* [En línea] The PostgreSQL Global Development Group, 2013. [Citado el: 7 de 5 de 2013.] http://www.postgresql.org/.

25. **PgAdmin III.** Guia de Ubuntu. *Guia de Ubuntu.* [En línea] 10 de 3 de 2008. [Citado el: 7 de 5 de 2013.] http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.

26. **Sifuentes, Miguel Almeyda.** Arquitectura de Software: ¿Qué es, y cómo funciona? *Arquitectura de Software: ¿Qué es, y cómo funciona?* [En línea] Universidad Continental, 10 de 1 de 2013. [Citado el: 9 de 5 de 2013.] http://www.ucci.edu.pe/blog/ingenieria_sistemas/?p=34.

27. *Modelo-Vista-Controlador.* **González, YD.** Cuba : revistatelematica, 2012.

28. **Cuello, Ing. Ronald.** *Persistencia de objetos.* España : s.n., 2010.

29. **Mendoza Navarro, Javier.** Diseño del Sistema de Tarjeta de Crédito. *Diseño del Sistema de Tarjeta de Crédito.* [En línea] 2012. [Citado el: 7 de 5 de 2013.] http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/basic/mendoza_nj/cap5.pdf.

30. **Cortez, Casallas, Gloria y Rubby.** *Introduccion a los patrones de software.* Andes : universidad de los Andes, 2010.

31. **Quiceno, Felipe.** Diagrama de Paquetes. *Diagrama de Paquetes .* [En línea] 9 de 5 de 2011. [Citado el: 5 de 5 de 2013.] http://analisisydiseoiii.blogspot.com/2011/05/diagrama-de-paquetes.html.

32. **sparxsystems.** Diagrama de Secuencia UML 2. *Diagrama de Secuencia UML 2.* [En línea] 2007. [Citado el: 9 de 5 de 2013.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_sequencediagram.html.

33. **Salazar, Cristian.** *Modelo de datos.* Chile : Académico at Universidad Austral de Chile, 2012.

34. **JeyssonPoquioma.** *Implementación.* España : s.n., 2013.

35. **Cruz Quispe, Víctor Fabio, Gutiérrez Mamani, Ever Dino y Mendivil Torrico, Luís Briam.** *Diagrama de componentes.* 2012.

36. **Michael, Marca Huallpara Hugo.** *Diagrama de despliegue.* Argentina : s.n., 2012.

37. **Carrero.** Manejo de Errores Usando Excepciones Java. *Manejo de Errores Usando Excepciones Java.* [En línea] 2013. [Citado el: 10 de 5 de 2013.] http://www.programacion.com/articulo/manejo_de_errores_usando_excepciones_java_98/2.

38. **Calleja, Manuel Arias.** *Estándares de codificación.* 2010.

39. Junta de Andalucia. *Junta de Andalucia.* [En línea] [Citado el: 7 de 4 de 2013.] http://www.juntadeandalucia.es/servicios/madeja/contenido/subsistemas/arquitectura/capa-acceso-datos.

40. **Grupo CUYS.** *Metodología del Desarrollo del Software.* Argentina : s.n., 2011.

41. msdn. *msdn.* [En línea] Microsoft, 2013. [Citado el: 9 de 4 de 2013.] http://msdn.microsoft.com/es-es/library/bb972240.aspx.

42. SG Software Guru. *SG Software Guru.* [En línea] Creative Commons Attribution-ShareAlike 3.0 Unported, 2000. [Citado el: 4 de 4 de 2013.] http://sg.com.mx/content/view/922.

43. **DrWael.** *Historia de la Anestesia.* 1989.

44. **Fisher, Tim.** PC Support. *PC Support.* [En línea] http://pcsupport.about.com/od/termsag/g/term_driver.htm.

45. **Klumpe, Ulrich.** Nortis. *Nortis.* [En línea] http://www.medep.de/medibus_e/index.html.

46. **Martínez, Jairo Chapela.** *Introducci´on al entorno de desarrollo Eclipse.* 2007.

47. **Mirabal, Ing. Lisseth Agüero.** *Requerimientos funcionales y no funcionales.* Barcelona : s.n., 2013.

48. **SECURED.** *El puerto serie RS 232.*

49. **Vega, Miguel.** *Casos de uso.* Granada : LSI - UGR, 2010.

50. **WordPress.** Definicion de. [En línea] 2008. http://definicion.de/computadora/.

Referencias bibliográficas

1. **(UMMC), University of Maryland Medical Center.** Anestesia. *Anestesia.* [En línea] 2011. http://www.umm.edu/esp_ency/article/007280.htm.

2. **Pastrana, Jose Manuel.** Cirugia Estética Cancun. *Cirugia Estética Cancun.* [En línea] diciembre de 2012. [Citado el: 15 de enero de 2013.] http://cirugiaesteticacancun.com/?page_id=204.

3. **Montero, Barrante y Quirós.** *Introducción a los sistemas de control supervisor y de adquisición de datos (SCADA).* Costa Rica : s.n., 2004.

4. *Lectura y decodificación para el sistema alas HIS de datos transmitidos por máquinas de anestesia.* **González Martínez, Yoandy.** La Habana, Cuba : s.n., 2013.

5. **Fisher, Tim.** Driver. *Driver.* [En línea] 2012. [Citado el: 10 de 12 de 2012.] http://pcsupport.about.com/od/termsag/g/term_driver.htm.

6. **CLASA, Comite de neuroanestesiología de.** Neuroanestesiología. *Neuroanestesiología.* [En línea] 2005. http://neuroanestesiologia.mx/softw/softw.htm.

7. *REGANE: adquisición normalizada de señales de una máquina.* **Thevenet D, Simini F.** Argentina : s.n., 2012.

8. **Sterle, Adam .** Roche Diagnostics. *Roche Diagnostics.* [En línea] 6 de 2011. [Citado el: 12 de enero de 2013.] http://www.captodayonline.com/.

9. **Drager.** Drager. *Drager.* [En línea] Drägerwerk AG & Co. KGaA, 2012. http://www.draeger.com/ES/es/products/it_software/mon_innovian_solution_suite.jsp.

10. *Driver EPP (Encriptado Pin Pad).* **Medina, Gastón Nicolás.** Córdoba : s.n., 2003.

11. **Ourense, Campus .** *“SOFTWARE DE COMUNICACIÓN ENTRE 2 PC's A.* Vigo : s.n., 2007.

12. **Stallings, William.** Comunicaciones y redes de datos. *Comunicaciones y redes de datos.* [En línea] 2003. http://es.scribd.com/doc/30097309/Comunicaciones-y-Redes-de-Datos-William-Stallings.

13. **Bartolome, Jordi.** El protocolo MODBUS. *El protocolo MODBUS.* [En línea] enero de 2011. [Citado el: 12 de enero de 2013.] http://www.tolaemon.com/site/protocolo_modbus.

14. **Drager.** DragerVitalink. *DragerVitalink.* [En línea] mayo de 2006. http://www.capsuletech.com/Docs/DDIhelps/DragerVitalink.htm.

15. **GS, Fabius.** DagerMedical. [En línea] abril de 2006. http://www.draeger.com.

16. **Oracle.** Java. *Java.* [En línea] oracle. [Citado el: 15 de 11 de 2012.] http://www.java.com/es/download/faq/whatis_java.xml.

17. **Acuña, Karenny Brito.** *SELECCIÓN DE METODOLOGÍAS DE DESARROLLO PARA APLICACIONES WEB EN LA FACULTAD DE INFORMÁTICA DE LA UNIVERSIDAD DE CIENFUEGOS.* 2011.

18. **Rodriguez, Mylena.** *LENGUAJE UNIFICADO DE MODELAMIENTO.* ecuador : s.n.

19. **Pedro, David Cuesta y Peris, Amores Salva.** Netbeans. *Netbeans.* [En línea] 2009. http://www.slideshare.net/e1da4/netbeans-2164338.

20. **Cediel, Giovanny Rey.** *Giovynet.* 2012.

21. **Oracle.** Oracle Technology Network. *Oracle Technology Network.* [En línea] Oracle Support. [Citado el: 15 de 12 de 2012.] http://www.oracle.com/technetwork/java/index-jsp-141752.html.

22. **Management:, Simba.** cnet. *cnet.* [En línea] 2012. [Citado el: 9 de 5 de 2013.] http://es.download.cnet.com/JFreeChart/3000-2213_4-10103763.html%5D.

23. **Pressman, Roger S.** *Ingeniería de Software: un enfoque práctico, 5ta Edición.* Nueva York, E.U.A : McGraw-Hill, 2005.

24. PostgreSQL. *PostgreSQL.* [En línea] The PostgreSQL Global Development Group, 2013. [Citado el: 7 de 5 de 2013.] http://www.postgresql.org/.

25. **PgAdmin III.** Guia de Ubuntu. *Guia de Ubuntu.* [En línea] 10 de 3 de 2008. [Citado el: 7 de 5 de 2013.] http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.

26. **Sifuentes, Miguel Almeyda.** Arquitectura de Software: ¿Qué es, y cómo funciona? *Arquitectura de Software: ¿Qué es, y cómo funciona?* [En línea] Universidad Continental, 10 de 1 de 2013. [Citado el: 9 de 5 de 2013.] http://www.ucci.edu.pe/blog/ingenieria_sistemas/?p=34.

27. *Modelo-Vista-Controlador.* **González, YD.** Cuba : revistatelematica, 2012.

28. **Cuello, Ing. Ronald.** *Persistencia de objetos.* España : s.n., 2010.

29. **Mendoza Navarro, Javier.** Diseño del Sistema de Tarjeta de Crédito. *Diseño del Sistema de Tarjeta de Crédito.* [En línea] 2012. [Citado el: 7 de 5 de 2013.] http://sisbib.unmsm.edu.pe/bibvirtualdata/tesis/basic/mendoza_nj/cap5.pdf.

30. **Cortez, Casallas, Gloria y Rubby.** *Introduccion a los patrones de software.* Andes : universidad de los Andes, 2010.

31. **Quiceno, Felipe.** Diagrama de Paquetes. *Diagrama de Paquetes .* [En línea] 9 de 5 de 2011. [Citado el: 5 de 5 de 2013.] http://analisisydiseoiii.blogspot.com/2011/05/diagrama-de-paquetes.html.

32. **sparxsystems.** Diagrama de Secuencia UML 2. *Diagrama de Secuencia UML 2.* [En línea] 2007. [Citado el: 9 de 5 de 2013.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_sequencediagram.html.

33. **Salazar, Cristian.** *Modelo de datos.* Chile : Académico at Universidad Austral de Chile, 2012.

34. **JeyssonPoquioma.** *Implementación.* España : s.n., 2013.

35. **Cruz Quispe, Víctor Fabio, Gutiérrez Mamani, Ever Dino y Mendivil Torrico, Luís Briam.** *Diagrama de componentes.* 2012.

36. **Michael, Marca Huallpara Hugo.** *Diagrama de despliegue.* Argentina : s.n., 2012.

37. **Carrero.** Manejo de Errores Usando Excepciones Java. *Manejo de Errores Usando Excepciones Java.* [En línea] 2013. [Citado el: 10 de 5 de 2013.] http://www.programacion.com/articulo/manejo_de_errores_usando_excepciones_java_98/2.

38. **Calleja, Manuel Arias.** *Estándares de codificación.* 2010.

Glosario de términos

Adquisición: Consiste, en tomar un conjunto de señales físicas, convertirlas en tensiones eléctricas y digitalizarlas de manera que se puedan procesar en una computadora.

Anestesia: Acto médico controlado en el que se usan fármacos para bloquear la sensibilidad táctil y dolorosa de un paciente, sea en todo o parte de su cuerpo.

Anestesiología: Especialidad médica dedicada a la atención y cuidados especiales de los pacientes durante las intervenciones quirúrgicas y otros procesos que puedan resultar molestos o dolorosos.

ASCII: (acrónimo inglés de ***A****merican* ***S****tandard* ***C****ode for* ***I****nformation* ***I****nterchange — Código Estándar Estadounidense para el Intercambio de Información*), es un código de caracteres basado en el alfabeto latino, tal como se usa en inglés moderno y en otras lenguas occidentales.

Driver: Un driver es un software o programa que sirve de intermediario entre un dispositivo de hardware y el sistema operativo. Su finalidad es la de permitir extraer el máximo de las funcionalidades del dispositivo para el cual ha sido diseñado. (5)

***Framework*:** Estructura predefinida para la creación de aplicaciones. Puede estar formado por un conjunto de librerías y clases o por una arquitectura que facilita el desarrollo de software.

Protocolo de comunicación: Define las reglas para la transmisión y recepción de la información entre el ordenador y la máquina de anestesia, de modo que para que los nodos se puedan comunicar entre si es necesario que ambos empleen la misma configuración de protocolos.

Puerto serie: Es una interfaz de comunicaciones entre ordenadores y periféricos el cual envía y recibe información bit por bit. Un puerto serial posee un conector estándar y trabaja con un protocolo que permite la conexión de dispositivos al computador. Se denomina “serial” porque el puerto serie “serializa” los datos. Esto quiere decir que toma un byte de datos y transmite los 8 bits del byte de uno en uno.

UML: Lenguaje Unificado de Modelado (*Unified Modeling Language*). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software.

Anexos

Anexo 1. Diagramas de clases del diseño

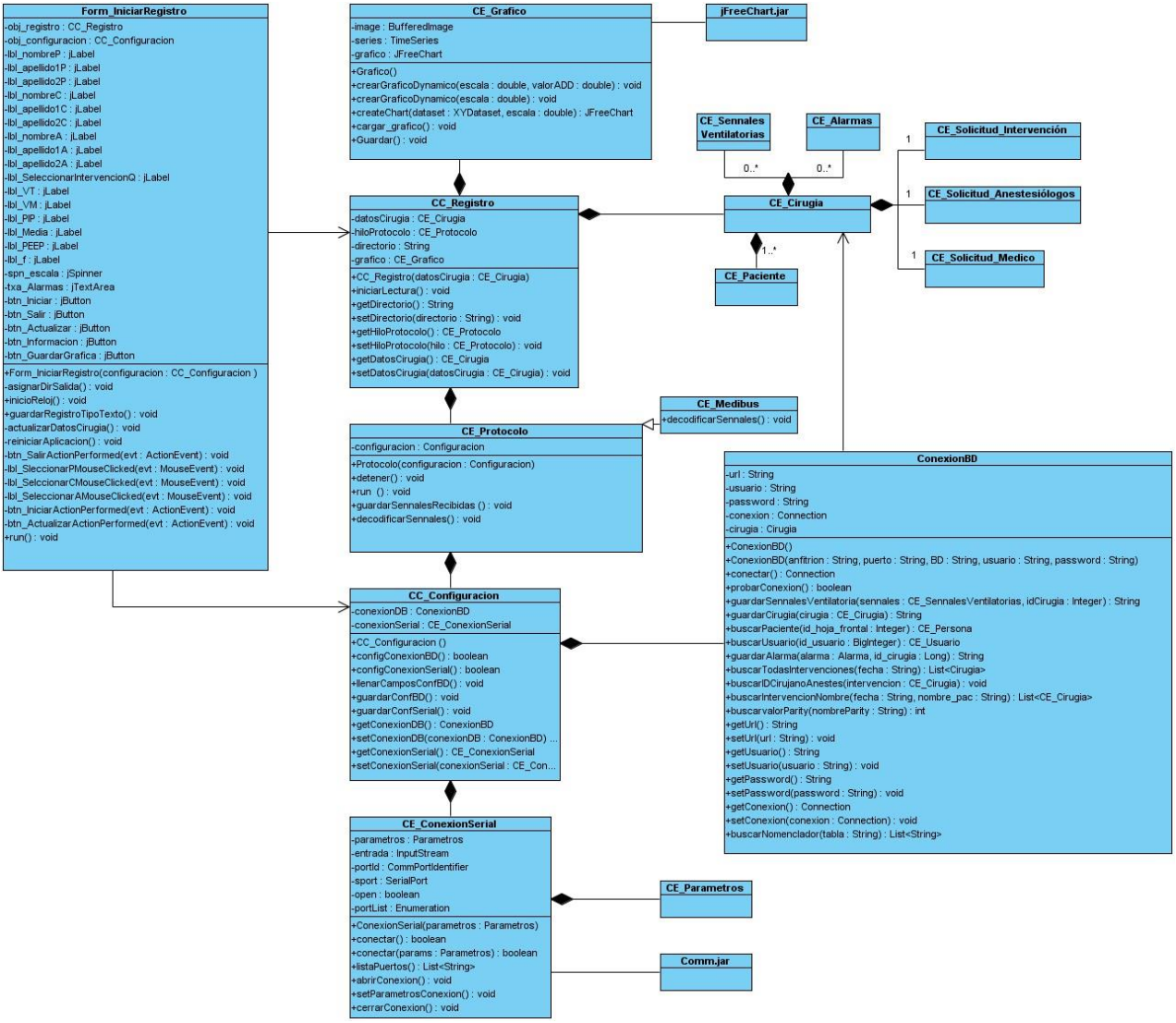

Figura 14: Diagrama de clases del diseño: Iniciar registro de variables fisiológicas

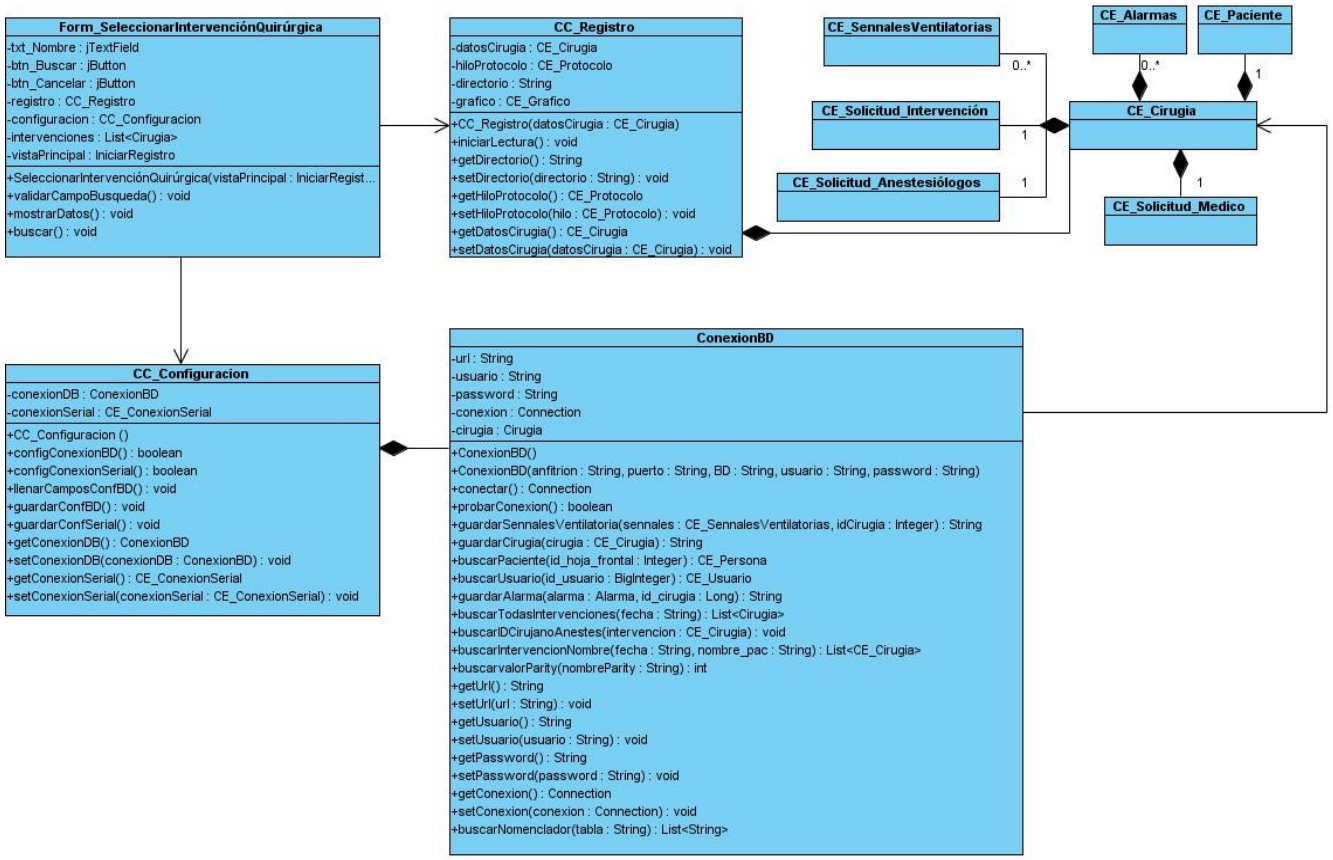

Figura 15: Diagrama de clases del diseño: Seleccionar solicitud de intervención

Imágenes de la aplicación

Figura 16: Interfaz de configurar conexiones

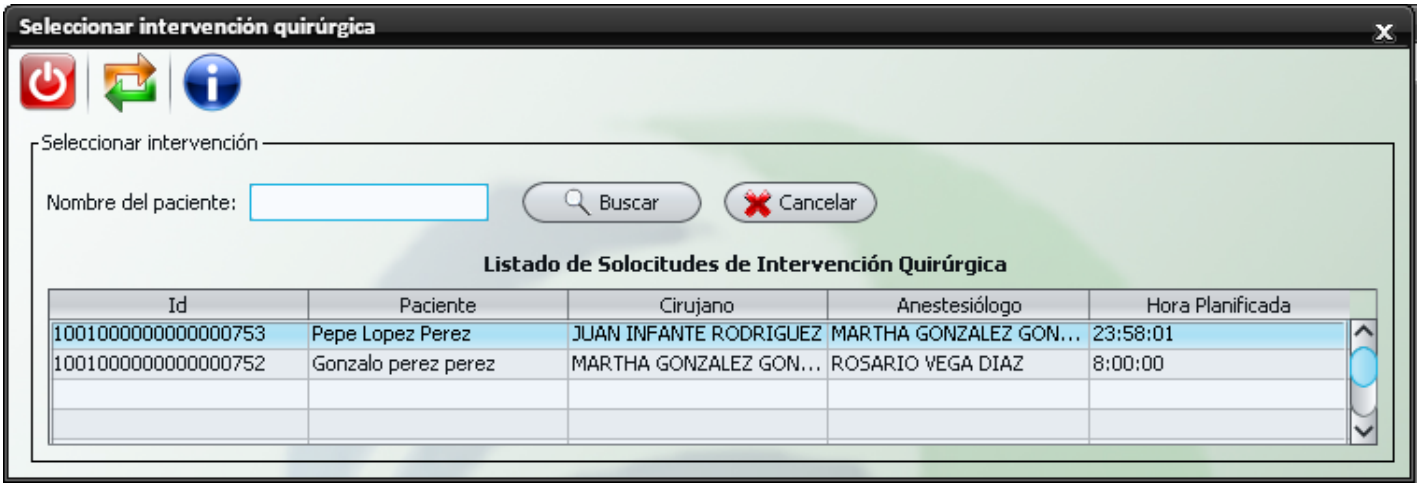

Figura 17: Interfaz de seleccionar intervención

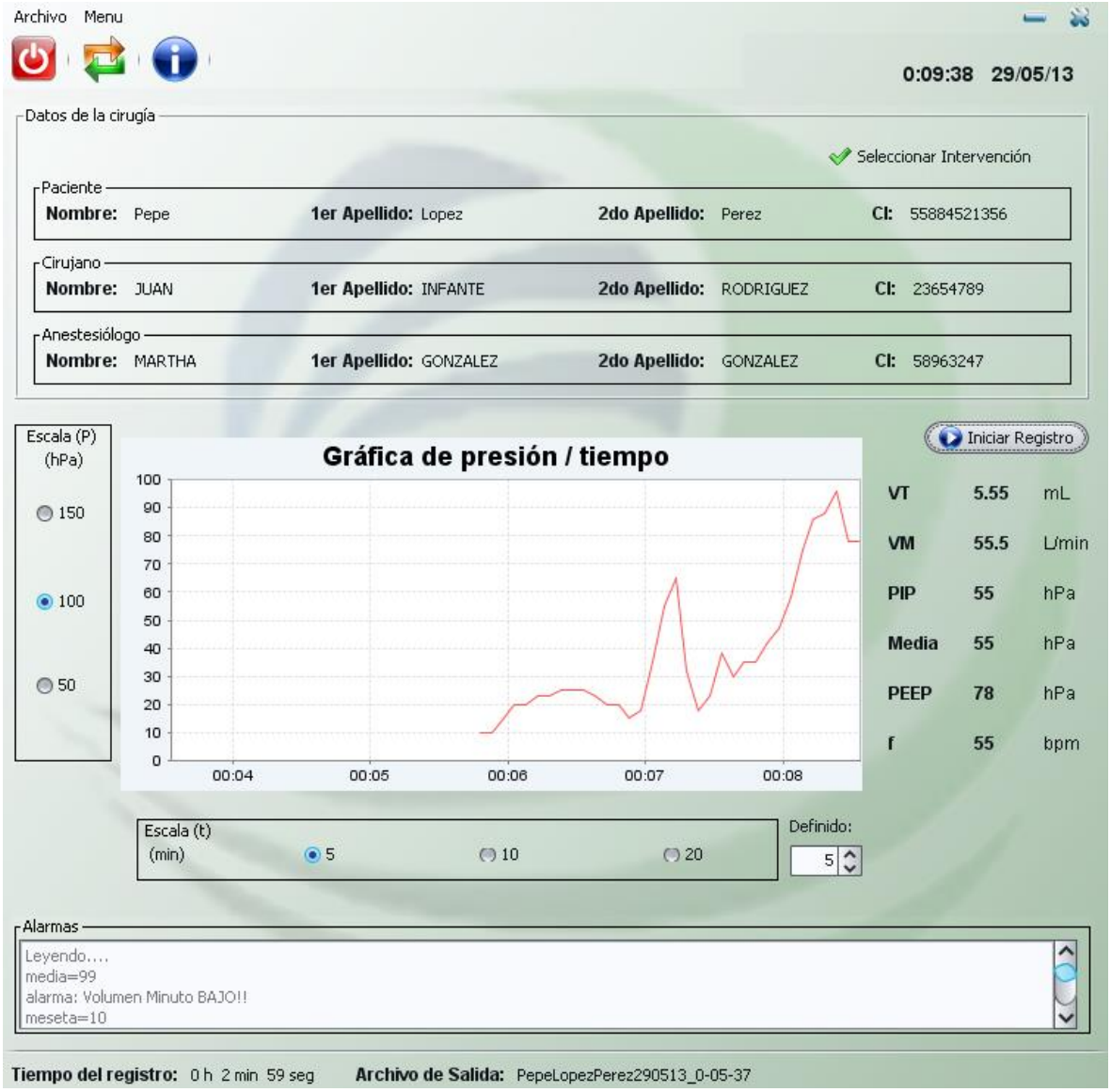

Figura 18: Interfaz de iniciar registro