Universidad de las Ciencias Informáticas

UCID

Título: Desarrollo del módulo de base de datos para un SCADA.

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

Autor: Igniris Valdivia Báez

Tutor: Ing. Adanlay Rivero Montero.

Cotutor: Ing. Osmany Jorge Riverón.

Ciudad de La Habana, 2010.

"Año 52 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____ _____

Igniris Valdivia Baez

Autor

Ing. Adanlay Rivero Montero

Tutor

Quien tenga una computadora dispone de todos los conocimientos publicados. La privilegiada memoria de la máquina le pertenece también a él.

Las ideas nacen de los conocimientos y de los valores éticos. Una parte importante del problema estaría resuelta tecnológicamente, la otra hay que cultivarla sin descanso o de lo contrario se impondrán los instintos más primarios.

La tarea que los graduados de la UCI tienen por delante es grandiosa. Espero que la cumplan, y la cumplirán.

Fidel Castro.

DATOS DE CONTACTO

Tutor:

Ing. Adanlay Rivero Montero

Graduado de Ingeniería en Ciencias Informáticas.

Correo electrónico: amontero@uci.cu

Graduado en el 2009

AGRADECIMIENTOS

A Mami, por ser mi inspiración, la persona en quien pienso antes de cumplir una tarea, por darme la seguridad de que siempre podré contar con ella.

A Papi, por ser mi guía, el ejemplo a seguir. Por tener siempre una solución para cada problema, un consejo oportuno en momentos de duda.

A nana y kiki, por quererme y cuidar a nuestros padres mientras no estoy.

A Hectico por su apoyo, por el amor que me ha profesado todos estos años, sin ti no hubiera sido posible lograr esta meta.

A Adanlay, mi tutor, por su apoyo y ayuda incondicional. Y a Tony por asesorarme cuando lo necesité.

A todos mis amigos, en especial a Joan, Yoansy, Yunet, Dayana e Irelys que han estado presentes en los momentos en que más los he necesitado.

A los profesores que han contribuido a mi formación a lo largo de los años.

Gracias a todos.

DEDICATORIA

Dedico este Trabajo de Diploma de manera especial a nuestro querido ***Comandante en Jefe Fidel Castro Ruz****, por hacer realidad el gran sueño que es la Universidad de las Ciencias Informáticas, por lograr mantener esta Revolución por tantos años, por brindarnos a los jóvenes su luz y su guía.*

A mi mamá, mi papá, a mis hermanos y mi novio.

Este triunfo es también de Ustedes.

RESUMEN

El almacenamiento de los datos y la forma en que los mismos se organizan y pueden ser consultados, constituye un elemento de gran importancia para el desarrollo de sistemas SCADA. En las Fuerzas Armadas Revolucionarias se necesitan sistemas de supervisión y control de equipamiento a distancia, que permitan tanto monitorizar como controlar, a cualquier nivel sobre los distintos despliegues. Estos sistemas necesitan de un módulo de base de datos que realice la recolección, organización y almacenamiento de los datos que recogen los sensores. El propósito de la presente investigación consistió en la realización del Diseño e Implementación de un módulo de base de datos para un sistema SCADA. Para ello fue necesario proponer un SGBD para desarrollar el módulo, luego la obtención de los requisitos de la base de datos y el desarrollo del módulo de base de datos, finalmente, se realizó la validación del diseño propuesto. El presente estudio se justificó por cuanto posee valor teórico, utilidad práctica, relevancia social, por su conveniencia y sobre la base de los beneficios netos que genera. Se obtuvo como resultado un modelo de datos normalizado y genérico que responde a las necesidades planteadas, el cual ha sido validado teórica y funcionalmente teniendo en cuenta un conjunto aspectos importantes para el área de las bases de datos relacionales.

PALABRAS CLAVES: Base de datos, SCADA, SGBD, Modelo de Datos.

Índice de Contenidos

Índice de Ilustraciones

Índice de Tablas

INTRODUCCIÓN

Los sistemas de información existen desde las primeras civilizaciones. Los datos se recopilaban, estructuraban, centralizaban y almacenaban convenientemente. El objetivo inmediato de este proceso era poder recuperar estos mismos datos u otros derivados de ellos en cualquier momento, sin necesidad de volverlos a recopilar, lo que solía ser el paso más costoso. Desde la antigüedad ha sido necesario almacenar grandes cantidades de información, es por ello que surge lo que en la actualidad se conoce como base de datos, que no es más que una colección de datos recopilados y estructurados que existe durante un período de tiempo.

SCADA proviene de las siglas de "*Supervisory Control And Data Acquisition*", es decir: adquisición de datos y control de supervisión. Se trata de una aplicación *software* especialmente diseñada para funcionar sobre ordenadores en el control de producción, proporcionando comunicación con los dispositivos de campo (controladores autónomos, autómatas programables, etc.) y controlando el proceso de forma automática desde la pantalla del ordenador. Además, provee de toda la información que se genera en el proceso productivo a diversos usuarios, tanto del mismo nivel como de otros supervisores dentro de la empresa: control de calidad, supervisión, mantenimiento, etc.

Las Fuerzas Armadas Revolucionarias a raíz del proceso de modernización de la técnica y el armamento que viene desarrollando, necesita de sistemas de supervisión y control de equipamiento a distancia que permita tanto monitorizar como controlar a cualquier nivel sobre los distintos despliegues. Estos sistemas necesitan de un módulo de base de datos que realice la recolección, organización y almacenamiento de los datos que recogen los sensores, que pueden ser usados posteriormente para conocer datos estadísticos entre otros.

Para dar respuesta a la situación problemática expuesta anteriormente se considera el siguiente **Problema Científico:** Necesidad de un módulo de base de datos que permita tener un histórico de los valores obtenidos de los sensores implantados en el sistema. Para dar respuesta al problema existente es necesario plantearse el siguiente **Objetivo General:** Diseñar e implementar un módulo de base de datos para un sistema SCADA.

Del objetivo general de la investigación se derivaron los siguientes **Objetivos Específicos:**

1. Obtener la propuesta de un SGBD para un SCADA.
2. Obtener los requisitos de la base de datos.
3. Desarrollar un módulo de base de datos.
4. Validar el diseño propuesto.

Del problema científico anterior y el objetivo general planteado, se decide centrar la investigación en: El diseño de base de datos como **Objeto de Estudio** y como **Campo de Acción:** El diseño de base de datos para sistemas SCADA.

Como **Idea a defender** se tiene que:
Con el desarrollo de un módulo de base de datos para un sistema SCADA se logrará que todos los datos estén guardados y ordenados de forma tal que permita un acceso rápido a los mismos y que estén disponibles en todo momento.

Métodos Científicos: La investigación está sustentada en los métodos teóricos y los métodos empíricos, que facilitarán la investigación y servirán de guía para organizar mejor el trabajo y de esta forma posibilitar entender el problema, estudiarlo, analizarlo y llegar a conclusiones para la solución. A continuación se explica la utilización de los que han sido seleccionados:

Métodos teóricos:

- Método Analítico – Sintético: Este método será utilizado para realizar un análisis de las tendencias actuales en cuanto a base de datos**,** para identificar cual SGBD escoger para el sistema.
- Método Análisis Histórico – Lógico: Este método será utilizado en el estudio y análisis de las principales herramientas utilizadas para el manejo de datos.
- Inductivo – Deductivo: Este método será utilizado en el momento de diseñar la base de datos, pues se debe definir un nuevo modelo de datos que cumpla con los requisitos necesarios.

Este trabajo está estructurado de la siguiente manera: Introducción, Desarrollo dividido en tres capítulos, Conclusiones, Recomendaciones, Bibliografía, Referencias bibliográficas, Anexos y Glosario de términos.

Capítulo #1: Fundamentación teórica

- Estudio de varios sistemas de gestión de información tanto en el ámbito nacional como internacional. Tratamiento de los datos históricos dentro de estos sistemas.
- Definir herramientas y metodologías que se utilizarán en la propuesta de solución.

Capítulo #2: Análisis y descripción de la solución propuesta

- Arquitectura de la propuesta de solución.
- Requisitos que se tienen en cuenta en el desarrollo de la propuesta de solución.
- Diagramas que muestran el diseño de la solución, además de la descripción de cada una de las tablas que los componen.

Capítulo #3: Validación del diseño realizado

- Se valida el diseño realizado de forma teórica y teniendo en cuenta aspectos como la integridad, la normalización, la seguridad y como responde el sistema a las pruebas.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA.

1.1. Introducción

En el presente capítulo se abordan los conceptos clave de esta investigación relacionados con los sistemas SCADA: su descripción, funcionalidades y componentes; además se tratan los temas correspondientes a las bases de datos y los sistemas gestores de bases de datos que se utilizan para trabajar en el área de los sistemas de supervisión y control. Se realiza también, un estudio del estado del arte de los sistemas SCADA, las herramientas y metodologías utilizadas en el mundo vinculadas al área de base de datos y relacionados con los mismos.

1.2. Sistemas SCADA

Los sistemas SCADA originalmente se diseñaron para cubrir las necesidades de un sistema de control centralizado, sobre procesos o complejos industriales distribuidos sobre áreas geográficas muy extensas. Tanto es así que en la definición clásica de un sistema SCADA se hace referencia a esta característica. Hoy en día, con el desarrollo de las redes digitales, la definición se tiene que modificar para incluir esta nueva forma de conectividad.

SCADA proviene de las siglas: **"Supervisory Control And Data Acquisition"**; es decir: hace referencia a un sistema de adquisición de datos y control supervisor. Tradicionalmente se define a un SCADA como un sistema que permite supervisar una planta o proceso por medio de una estación central que hace de Máster (llamada también estación maestra o unidad terminal maestra, MTU) y una o varias unidades remotas (generalmente RTU) por medio de las cuales se hace el control / adquisición de datos hacia / desde el campo. (Corrales, 2007)

Los sistemas SCADA *("Supervisory Control And Data Acquisition")* son aplicaciones de *software*, diseñadas con la finalidad de controlar y supervisar procesos a distancia. Se basan en la adquisición de datos de los procesos remotos. (Mendiburu Díaz, 2008)

Los sistemas SCADA, comprenden todas aquellas soluciones de aplicación que requieren de la captura de información de un proceso o planta industrial, la cual es utilizada para realizar una serie de análisis o estudios con los que se pueden obtener valiosos indicadores que permitan una realimentación sobre un operador o sobre el propio proceso. (Herrera Vázquez, 2008.)

Se considera que esta es la definición más completa de un SCADA:

Desde el punto de vista de *software*, un SCADA, se puede definir como una aplicación diseñada para funcionar sobre ordenadores en el control de producción, proporcionando comunicación con los dispositivos de campo (controladores autónomos, autónomas programables, etc.) y controlando el proceso de forma automática desde la pantalla del ordenador. Además provee de toda la información que se genera en el proceso productivo a diversos usuarios tanto del mismo nivel como de otros supervisores dentro de la empresa: control de calidad, supervisión, mantenimiento, etc. (Ferrari, 2005)

Estos sistemas incluyen la transferencia de datos entre el SCADA una computadora central y un número de Unidades Terminales Remotas (RTU) y / o Controladores Lógicos Programables (PLC), y el servidor central y los terminales de operador. Un sistema SCADA recoge información (por ejemplo, dónde se ha producido una fuga en una tubería), transfiere la información a un sitio central, luego alerta a la estación principal de que la fuga ha ocurrido, llevando consigo la realización de análisis y de control necesario, tales como determinar si la fuga es crítica, y mostrando la información de una manera lógica y organizada. Estos sistemas pueden ser relativamente simples, tales como uno que monitorea las condiciones ambientales de un pequeño edificio de oficinas, o muy complejas, como un sistema que monitorea toda la actividad en una planta de energía nuclear o la actividad de un sistema municipal de agua. (National Comunications System, 2004)

1.2.1. Descripción general de un SCADA

En la actualidad los sistemas SCADA son usados para un sin número de aplicaciones de las que se pueden mencionar: el control de oleoductos, plantas químicas, sistemas de transmisión de energía eléctrica, yacimientos de gas y petróleo y control de todos los procesos de esta rama, redes de distribución de gas natural, subterráneos, generación energética, sistemas de distribución de agua, entre otros.

Cada uno de los elementos del SCADA involucran muchos subsistemas, por ejemplo, la adquisición de los datos puede estar a cargo de un Controlador Lógico Programable (PLC) el cual toma las señales y las envía a las estaciones remotas usando un protocolo determinado, otra forma podría ser que una computadora realice la adquisición vía un hardware especializado y luego esa información la transmita hacia un equipo de radio vía su puerto serial, y así existen muchas otras alternativas.

Las tareas de Supervisión y Control generalmente están más relacionadas con el *software* SCADA, en él, el operador puede visualizar en la pantalla del computador de cada una de las estaciones remotas que conforman el sistema, los estados de esta, las situaciones de alarma y tomar acciones físicas sobre algún equipo lejano, la comunicación se realiza mediante buses especiales o redes LAN. Todo esto se ejecuta normalmente en tiempo real, y están diseñados para dar al operador de planta la posibilidad de supervisar y controlar dichos procesos. (Mendiburu Díaz, 2008)

Hay ciertas capacidades que no se puede obviar que debe tener un *software* SCADA según (Mendiburu Díaz, 2008), el mismo debe ser capaz de ofrecer al sistema:

1. Posibilidad de crear paneles de alarma, que exigen la presencia del operador para reconocer una parada o situación de alarma, con registro de incidencias.
2. Generación de datos históricos de las señales de planta, que pueden ser procesados con posterioridad.
3. Ejecución de programas, que modifican la ley de control, o incluso anular o modificar las tareas asociadas al autómata, bajo ciertas condiciones.
4. Posibilidad de programación numérica, que permite realizar cálculos aritméticos de elevada resolución sobre el CPU del ordenador.

Algunos requisitos básicos son:

- Todo sistema debe tener arquitectura abierta, es decir, debe permitir su crecimiento y expansión, así como deben poder adecuarse a las necesidades futuras del proceso y de la planta.
- La programación e instalación no debe presentar mayor dificultad, debe contar con interfaces gráficas que muestren un esquema básico y real del proceso.
- Deben permitir la adquisición de datos de todo equipo, así como la comunicación a nivel interno y externo (redes locales y de gestión).
- Deben ser programas sencillos de instalar, sin excesivas exigencias de hardware, y fáciles de utilizar, con interfaces amigables para el usuario.

1.2.2. Funcionalidades principales de un SCADA

Los sistemas SCADA presentan funcionalidades que son estándar para cualquiera de estos sistemas las más importantes en este sentido son las que se relacionan a continuación. (Mendiburu Díaz, 2008)

- Supervisión remota de instalaciones y equipos: Permite al operador conocer el estado de desempeño de las instalaciones y los equipos alojados en la planta, lo que permite dirigir las tareas de mantenimiento y estadística de fallas.
- Control remoto de instalaciones y equipos: Mediante el sistema se puede activar o desactivar los equipos remotamente (por ejemplo abrir válvulas, activar interruptores, prender motores, etc.), de manera automática y también manual. Además, es posible ajustar parámetros, valores de referencia, algoritmos de control, etc.
- Procesamiento de datos: El conjunto de datos adquiridos conforma la información que alimenta el sistema, esta información es procesada, analizada, y comparada con datos anteriores, y con datos de otros puntos de referencia, dando como resultado una información confiable y veraz.
- Visualización de gráfica dinámica: El sistema es capaz de brindar imágenes en movimiento que representen el comportamiento del proceso, dándole al operador la impresión de estar presente dentro de una planta real. Estos gráficos también pueden corresponder a curvas de las señales analizadas en el tiempo.
- Generación de reportes: El sistema permite generar informes con datos estadísticos del proceso en un tiempo determinado por el operador.
- Representación de señales de alarma: A través de las señales de alarma se logra alertar al operador de que está frente a una falla o de la presencia de una condición perjudicial o fuera de lo aceptable. Estas señales pueden ser tanto visuales como sonoras.
- Almacenamiento de información histórica: Se cuenta con la opción de almacenar los datos adquiridos, esta información puede analizarse posteriormente, el tiempo de almacenamiento dependerá del operador o de los requisitos del sistema.
- Programación de eventos: Está referido a la posibilidad de programar subprogramas que brinden automáticamente reportes, estadísticas, gráfica de curvas, activación de tareas automáticas, etc.

En este esquema se puede apreciar las **Funciones Básicas** de un SCADA:

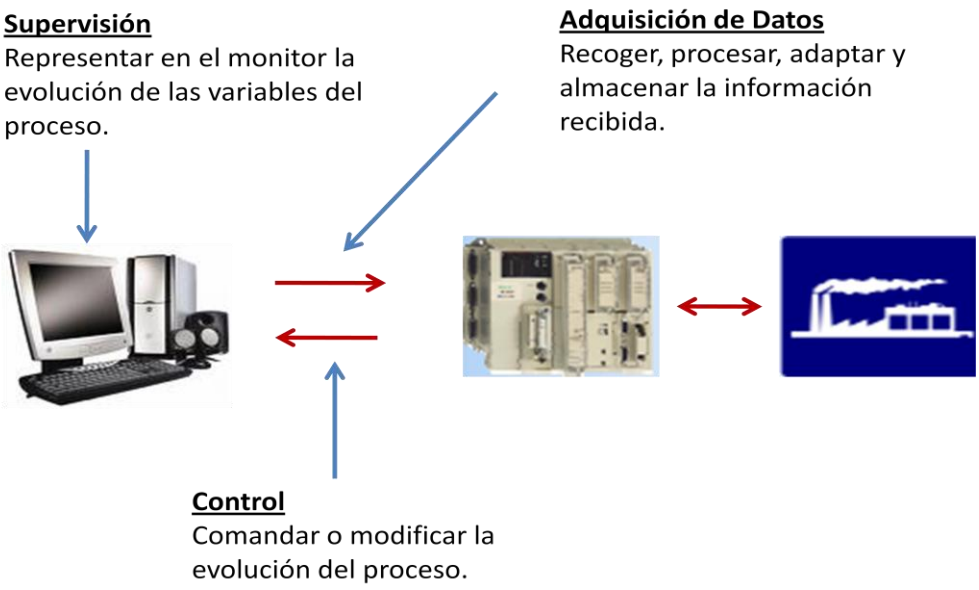

Ilustración 1: Funciones Básicas de un SCADA.

También tiene algunas **Funciones avanzadas** como:

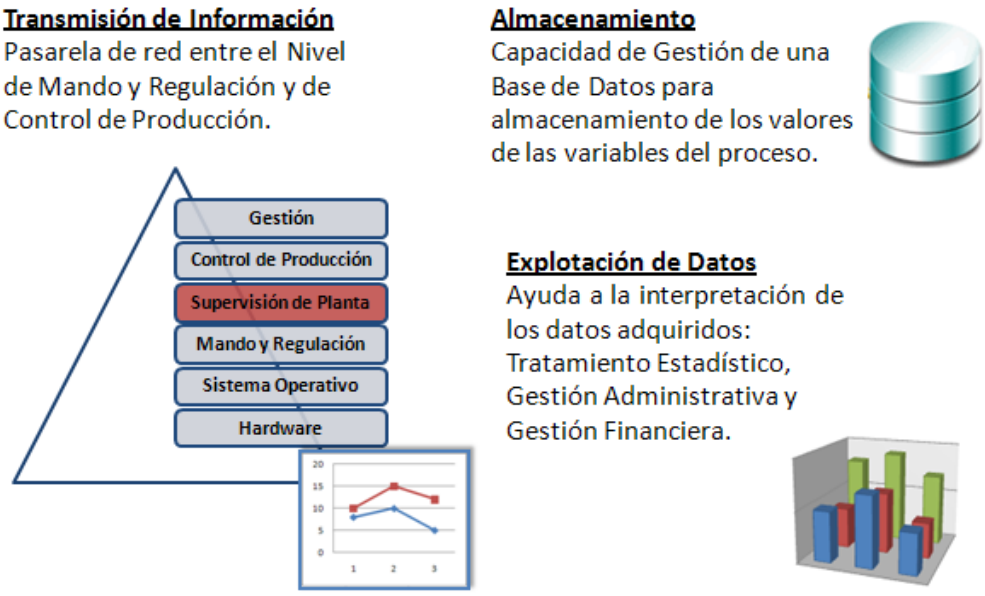

Ilustración 2: Funciones Avanzadas de un SCADA.

Un sistema SCADA está conformado por: (Mendiburu Díaz, 2008)

- Interfaz Operador Máquinas: Es el entorno visual que brinda el sistema para que el operador se adapte al proceso desarrollado por la planta. Permite la interacción del ser humano con los medios tecnológicos implementados.
- Unidad Central (MTU): Conocido como Unidad Maestra. Ejecuta las acciones de mando (programadas) sobre la base de los valores actuales de las variables medidas. La programación se realiza por medio de bloques de programa en lenguaje de alto nivel (como C, Basic, etc.). También se encarga del almacenamiento y procesado ordenado de los datos, de forma tal que otra aplicación o dispositivo pueda tener acceso a ellos.
- Unidad Remota (RTU): Lo constituye todo elemento que envía algún tipo de información a la unidad central. Es parte del proceso productivo y necesariamente se encuentra ubicada en la planta.
- Sistema de Comunicaciones: Se encarga de la transferencia de información del punto donde se realizan las operaciones, hasta el punto donde se supervisa y controla el proceso. Lo conforman los transmisores, receptores y medios de comunicación.
- Transductores: Son los elementos que permiten la conversión de una señal física en una señal eléctrica (y viceversa). Su calibración es muy importante para que no haya problema con la confusión de valores de los datos.

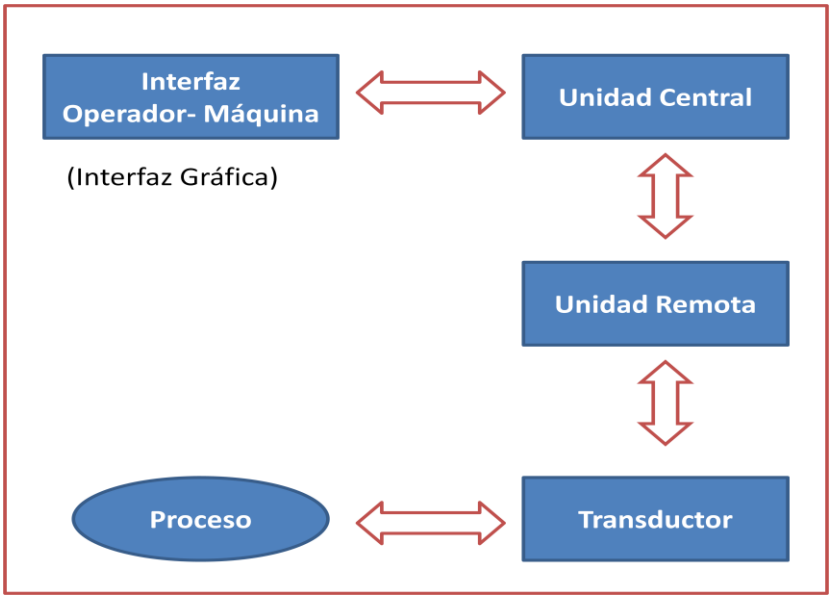

Ilustración 3: Esquema de los elementos de un sistema SCADA.

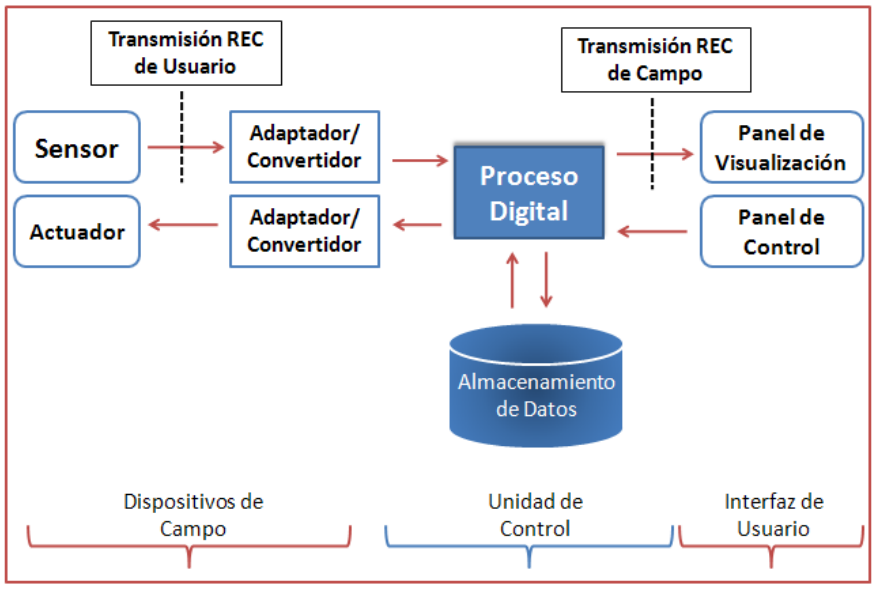

Ilustración 4: Esquema básico de un Sistema SCADA.

1.2.3. Arquitectura de los sistemas SCADA

Históricamente los sistemas SCADA presentan un equipo que, conectado físicamente a los dispositivos de adquisición de datos, actúa como servidor para sus clientes interconectados a través de una red de comunicaciones.

El flujo de la información es como sigue: Un fenómeno físico (presión, temperatura, flujo, el disparo de un breaker, exceso de presión en una tubería, nivel de un tanque, etc.), es captado por un transductor**,** el cual alimenta una señal eléctrica a un transmisor para que éste entregue una señal análoga también eléctrica pero normalizada hacia un PLC o RTU. Dependiendo del caso, el transmisor además proveerá aislación eléctrica y filtraje con el objeto de reducir posibles transitorios y ruido originado en el campo. (Corrales, 2007)

La señal que entregan los transmisores se envía hacia un cuarto de control donde se reúne la información de toda la planta industrial. Una vez que los datos llegan al centro de acopio, generalmente una computadora, se los almacena para su análisis, generación de históricos y para la toma de decisiones. Simultáneamente, por medio de una HMI se muestra la información al operador del sistema, para la supervisión.

Basado en la información, el operador puede tomar decisiones que pueden modificar el trabajo del proceso supervisado. El operador comanda a la computadora y ésta obedece enviando un dato que al final llegará a un actuador que al mover un elemento final logran controlar la variable bajo supervisión o control en el campo.

1.2.4. Ciclo de Vida de los sistemas SCADA

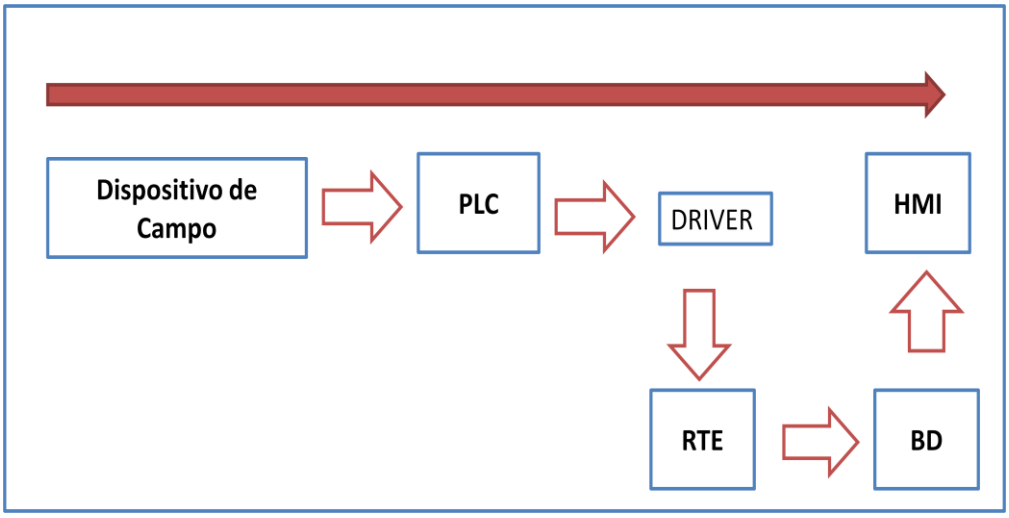

Ilustración 5: Ciclo de Vida del SCADA en sentido Directo.

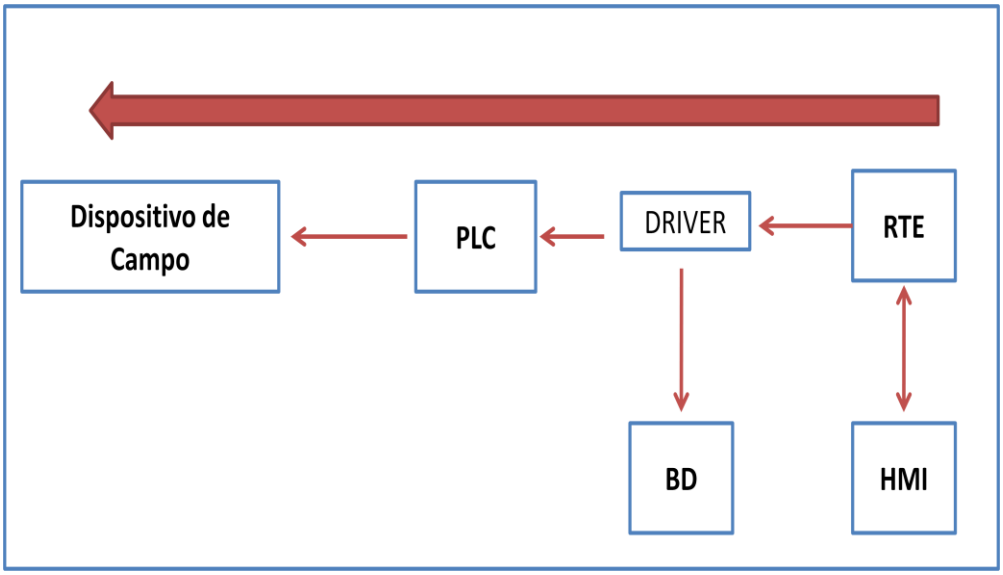

Ilustración 6: Ciclo de Vida del SCADA en sentido Inverso.

1.3. Herramientas utilizadas en el manejo de datos para sistemas SCADA

1.3.1. Base de Datos y Modelo de datos: Definición.

Un elemento importante en esta investigación lo constituyen las Bases de Datos (BD), puesto que la propuesta de nuestro trabajo es el módulo de BD para un Sistema SCADA.

Las bases de datos, proporcionan unos datos organizados, en un entorno estático, según determinados criterios, y facilitan su exploración y consulta selectiva. Se pueden emplear en múltiples actividades como por ejemplo: seleccionar datos relevantes para resolver problemas, analizar y relacionar datos, extraer conclusiones, comprobar hipótesis. Una base de datos es una recopilación de datos o información relacionados entre sí, la misma está compuesta por filas que son los registros y por columnas que son los campos. (Mesquida Peña & Basulto Uribe, 2009)

El término base de datos ha sido definido de varias maneras. Algunas de ellas son:

- Colección de información organizada y relacionada entre ella. (Navarrete Navarrete, 2004)
- Colección organizada de datos, relativa a un problema concreto, que puede ser compartida por un conjunto de usuarios/aplicaciones. (Zorrilla Pantaleón, 2003)
- Un conjunto exhaustivo de datos estructurados, fiables y homogéneos, organizados independientemente de su utilización e implementación en una computadora, accesibles en tiempo real, que pueden compartir varios usuarios con necesidades de información diferentes y no predecibles en el tiempo. (Núñez Camallea, 2004)

Modelo de datos.

Los modelos de datos aportan la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos, así como la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de sistemas de información. Algunas de las definiciones son:

- Colección de herramientas conceptuales que se emplean para especificar datos, las relaciones entre ellos, su semántica asociada y las restricciones de integridad. (Zorrilla Pantaleón, 2009)

- Es un lenguaje utilizado para la descripción de una base de datos**.** Por lo general, un modelo de datos permite describir las estructuras de datos de la base (el tipo de los datos que incluye la base y la forma en que se relacionan), las restricciones de integridad (las condiciones que los datos deben cumplir para reflejar correctamente la realidad deseada) y las operaciones de manipulación de los datos (agregado, borrado, modificación y recuperación de los datos de la base). (Definición.de, 2008)

- Con respecto al diseño de bases de datos, el *modelado de datos* puede ser descrito así: "dados los requerimientos de información y proceso de una aplicación de uso intensivo de datos (por ejemplo,

un sistema de información), construir una representación de la aplicación que capture las propiedades estáticas y dinámicas requeridas para dar soporte a los procesos deseados (por ejemplo, transacciones y consultas). Además de capturar las necesidades dadas en el momento de la etapa de diseño, la representación debe ser capaz de dar cabida a eventuales futuros requerimientos".

1.3.2. Revisión histórica de los sistemas de Base de Datos.

Las bases de datos tienen sus orígenes en la década del 50, se relacionan algunos de los elementos más importantes en el surgimiento y desarrollo de las mismas: (Zorrilla Pantaleón, 2003)

- De 1950 a 1960, se desarrollaron las cintas magnéticas para el almacenamiento de datos. Lectura secuencial.
- En la década de los 70, aparición de los discos magnéticos lo que permitió el acceso directo a los datos. Surgen las BD jerárquicas y en red.
    - Tratamiento de información requería conocer detalles de implementación (bajo nivel).
    - Codificación de consultas de forma procedimental.
    - Edgar Frank Codd definió el modelo relacional en 1970.
- Pero hasta 1980 no aparecieron gestores relacionales comerciales (Oracle, IBM DB2, Ingres,…) con buen rendimiento y más fáciles de diseñar y mantener (independencia física y lógica)
    - Estudios BD distribuidas y paralelas.
    - Inicios en BD orientadas a objetos.
- Desde 1990:
    - BD relacionales orientadas a objetos.
    - BD dimensionales y OLAP.
    - XML.
    - Minería de datos.

Sus características pueden ser ventajosas o no: pueden ayudarnos para almacenar, organizar, recuperar, comunicar y manejar información en formas que serían imposibles sin los computadores, pero también nos afecta de alguna manera ya que existen enormes cantidades de información en bases de datos de las que no se tiene control del acceso.

Las bases de datos tienen muchos usos: nos facilitan el almacenamiento de grandes cantidades de información; permiten la recuperación rápida y flexible de información, con ellas se puede organizar y reorganizar la información, así como imprimirla o distribuirla en formas diversas.

1.3.3. Tipos de base de datos

Los tipos de BD son:

1. Jerárquicas
2. En red
3. Relacionales
4. Orientadas a objetos
5. Objeto relacionales

Se centrará el estudio en las **Base de Datos Relacionales** pues es la que se seleccionó para desarrollar el módulo de BD, este tipo de BD tiene las siguientes características**:**

- Los datos se muestran en forma de tablas y relaciones.
- Edgar Frank Codd definió las bases del modelo relacional a finales de los 70. En las bases de Codd se definían los objetivos de este modelo:
  - **Independencia física**. La forma de almacenar los datos, no debe influir en su manipulación lógica.
  - **Independencia lógica**. Las aplicaciones que utilizan la base de datos no deben ser modificadas porque se modifiquen elementos de la base de datos.
  - **Flexibilidad**. La base de datos ofrece fácilmente distintas vistas en función de los usuarios y aplicaciones.
  - **Uniformidad**. Las estructuras lógicas siempre tienen una única forma conceptual (las tablas).
  - **Sencillez**.

Principios sobre Bases de Datos Relacionales:

Una base de datos relacional es una base de datos que cumple con el modelo relacional, el cual es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente.

Tiene como principios:

1. Una base de datos relacional se compone de varias tablas o relaciones.
2. No pueden existir dos tablas con el mismo nombre.
3. Cada tabla es a su vez un conjunto de registros, filas o tuplas.
4. Cada registro representa un objeto del mundo real.
5. Cada uno de estos registros consta de varias columnas, campos o atributos.
6. No pueden existir dos columnas con el mismo nombre en una misma tabla.
7. Los valores almacenados en una columna deben ser del mismo tipo de dato.
8. Todas las filas de una misma tabla poseen el mismo número de columnas.
9. No se considera el orden en que se almacenan los registros en las tablas.
10. No se considera el orden en que se almacenan las tablas en la base de datos.
11. La información puede ser recuperada o almacenada por medio de sentencias llamadas consultas.

1.3.4. Base de datos aplicado a sistemas SCADA

Las BD usadas para los sistemas SCADA son de dos tipos: Base de Datos en Tiempo Real y Base de Datos Histórica. La mayoría de los sistemas cuentan con estos dos tipos de BD, aunque algunos sólo implementan una de estas BD.

La Base de Datos en tiempo real (real-time database -RTDB) aplicado a sistemas SCADA, es el área de memoria compartida del kernel del programa/equipo usada para almacenar el valor actual de los datos o variables monitoreadas. Es una memoria que guarda datos mientras el programa que la origina está en ejecución por lo cual, no guarda valores históricos una vez que este es detenido. (Interactive Programmers Community, 2007)

Los sistemas SCADA por ser multitareas (realizan operaciones de lectura/escritura I/O, registro de datos, alarmas, tendencias y cálculos al mismo tiempo) poseen una base de datos en tiempo real basada en su mayoría, en el valor de *tags*/variables que el sistema tiene configurado y que pueden cambiar de forma externa (lectura desde un PLC) o interna (ingresada por el usuario). Estos datos cambiantes viven en esa base de datos de tiempo real para ser usada por cualquier otra aplicación o variable dentro del SCADA que pueda necesitarla.

Por otra parte, la **base de datos histórica** es: Una base de datos que proporciona una perspectiva histórica de los datos. Esta permite obtener comportamientos, tendencias, reportes de eventos ocurridos anteriormente entre otras funcionalidades.

1.3.5. Diseño de Bases de datos

Según (Zorrilla Pantaleón, 2009) **las fases del diseño de Bases de Datos:**

- Fase inicial: análisis de requisitos. Descripción de la información a gestionar y sus procesos. Entrevistas con usuarios y expertos.
  - Captación y Análisis de requisitos. Especificación funcional
- Diseño conceptual: traducción del análisis de requisitos al esquema conceptual. Representación generalmente gráfica de las entidades y sus relaciones.
  - Modelo Entidad Relación, modelo UML.
  - DFD, diagrama de casos, diagramas de colaboración, de secuencia, etc.
- Implantación en el gestor:
  - Diseño lógico: traducción del modelo conceptual al LDD del gestor correspondiente. Modelo relacional.
  - Diseño físico: determina la organización de archivos y las estructuras de almacenamiento interno.

A continuación, se amplía algunos de los procesos más importantes por los que está compuesto el diseño de Bases de Datos:

1. **Captación y análisis de requerimientos:**
   - Caracterizar de forma completa las necesidades de los usuarios de la BD, tanto en los datos como en las operaciones a realizar con los datos.
   - Entrevistar los futuros usuarios de la BD para captar las necesidades.

Resultado:

- **Requisitos de datos:** Necesidades de datos. Especificación de la información que se quiere guardar.
- **Requisitos funcionales:** Necesidades de manipulación de datos. Especificación de las operaciones a realizar con los datos.

2. **Diseño conceptual de la BD:**
   - Una vez encontrado el modelo abstracto que se quiere utilizar, el diseñador aplica los conceptos de este modelo para traducir los requisitos de datos del usuario al modelo abstracto, formando el esquema conceptual de la BD.
   - Validar el esquema conceptual sobre las transacciones especificadas en los requisitos funcionales (consultas, actualizaciones, borrados, etc.).

Además, se debe tener en cuenta que el diseño de una base de datos requiere que se realicen determinadas tareas, entre las que se encuentran:

- Determinar la finalidad de la base de datos.
- Determinar las tablas que se necesitan en la base de datos.
- Determinar los campos que se necesitan en las tablas.
- Perfeccionar los campos y las tablas.
- Identificar los campos con valores exclusivos en cada registro.
- Determinar las relaciones entre las tablas.
- Perfeccionar el diseño.
- Introducir datos y crear otros objetos de la base de datos.

1.3.6. Gestores de BD

Entre la base de datos física, es decir, almacenamiento real de los datos y los usuarios de ésta, existe una interfaz de *software*, llamada Sistema de Gestión de Bases de Datos (SGBD), que es el responsable de tratar todas las peticiones de información de dichos usuarios.

Este sistema es un conjunto de programas de propósito general, que permite a los usuarios controlar el acceso y la utilización de la base de datos, para incluir, modificar o recuperar información, incluyendo prestaciones con el fin de conseguir la independencia, integridad y seguridad de los datos, y la concurrencia de los usuarios. (Núñez Camallea, 2004)

Otra definición está dada por:

- Programa o conjunto de programas que sirve para mantener bases de datos y responder consultas sobre ellas. (Zorrilla Pantaleón, 2003)

Según (Castillo, 2008) **los SGBD deben permitir:**

- Crear bases de datos y esquemas.
  - Data Definition Language.
- Preguntar sobre los datos.
  - Query Language.
- Almacenar los datos de forma persistente.
- Controlar el acceso de múltiples usuarios.
- Garantizarla disponibilidad de los datos, independientemente de los fallos del sistema.

1.4. Soluciones Similares

1.4.1. Ámbito Internacional

En el mundo existen diversos sistemas SCADA con varios propósitos, se presentan algunos de los más sobresalientes.

SW SCADA	Fabricante:
MIRAGE	CONTROLES S.A
Likindoy	Axaragua
FactorySuite 2000	Wonderware
Génesis 32	Iconics
SCADA Eléctrico	Kyber
WinCC	Siemens
SCADA In Touch	LOGITEK.
RS-VIEW32	Rockwell
Factory Link 7	USDATA

Tabla 1: Sistemas SCADA en el mundo.

1.4.1.1. Mirage

El sistema MIRAGE es un *software* SCADA y su función principal es supervisar el funcionamiento de un sistema de telecontrol. Está basado en una arquitectura modular sobre plataforma Windows XP, el mismo utiliza en todo momento servicios estándar del sistema operativo ofreciendo de este modo una arquitectura abierta y fácilmente expandible.

Características principales:

- Arquitectura modular cliente/servidor con comunicación TCP/IP y RPC entre módulos.
- Capacidad de crecimiento y configuración según los requerimientos de confiabilidad y de la aplicación.
- Puede ser instalado en una única PC o en múltiples PC en red con los diversos módulos duplicados.
- Diseños de alta confiabilidad con posibilidad de respaldo en caliente (Hot Stand by) de módulos principales.
- Seguridad de acceso por usuario y por estación de trabajo completamente configurable.
- Almacenamiento de históricos y eventos de base de datos estándar a través de ODBC.
- Consultas a la base de datos con OBDC.

El sistema consta de varios módulos entre los que se encuentra el MBDH (módulo de base de datos histórica), además cuenta con el MSSC (Módulo servidor SCADA) donde se encuentra incluida la Base de datos de tiempo real y la Lista de eventos recientes.

Módulo servidor SCADA, MSSC

Es el núcleo del Sistema MIRAGE y su función es mantener actualizada la base de datos de tiempo real con el estado del sistema tele-controlado y comunicar ese estado a los módulos clientes.

Funciones:

- Recibir los datos de las unidades remotas a través de los módulos de adquisición de datos, procesarlos, almacenarlos en la base de datos de tiempo real y comunicarlos al resto de los módulos.
- Aceptar solicitudes de los clientes para realizar comandos sobre las unidades remotas o modificaciones sobre la base de datos de tiempo real, marcas, *tags*, valores manuales, los cuales una vez procesados son comunicados al resto de los módulos del sistema.
- Mantener una lista de eventos en memoria permitiendo de esta forma un manejo unificado de la misma entre las distintas consolas de operación MMI.
- Administrar la seguridad y el control de acceso de los usuarios permitiendo o denegando la apertura de consolas.

Módulo de base de datos histórica:

El módulo de base de datos histórica es cliente del servidor SCADA, recibe los cambios sobre el estado del sistema y los almacena, a través de ODBC en las bases de datos históricas de variables y de eventos.

Funciones básicas:

- Base de datos de variables: almacena los valores de las variables con la marca de tiempo absoluta junto con los valores máximo, mínimo, promedio y marcas de estado.
- Base de datos de eventos: almacena los eventos de las RTU, Centro de Control y las operaciones con la marca de tiempo y la identificación de la consola en que se originó.
- Permite administración local o remota.

Duplicación:

Cada módulo de base de datos puede disponer de enlaces con dos servidores SCADA diferentes, obteniendo sus datos del servidor activo; a su vez, cada módulo de base de datos puede almacenar en dos bases de datos diferentes.

Cualidades del formato de almacenamiento de los datos:

Como la comunicación con la base de datos se realiza a través de ODBC es posible utilizar cualquier programa de base de datos que soporte este protocolo, por ejemplo, SQL Server, Oracle, Access, entre otras.

1.4.1.2. Likindoy

LIKINDOY es un SCADA realizado con tecnologías libres, que acerca la automatización y el control de los sistemas industriales a los expertos en informática.

Está realizado íntegramente con tecnologías Open Source:

- El sistema operativo es Linux.
- El lenguaje de programación es Python.
- El gestor de bases de datos es MySQL.
- Para la generación de gráficas se usa el lenguaje de programación R ideado para uso estadístico y representación visual de datos.

LIKINDOY-HTR

El módulo para la gestión de históricos dispone de 4 niveles de procesamiento de los datos.

1. **Recolección de los datos:** Donde el programa usa diferentes módulos para conseguir la información de fuentes externas.
2. **Carga de los datos:** Recoge los datos descargados y los transfiere de modo homogéneo a una base de datos SQL.
3. **Generación de gráficas:** Usa la información almacenada en la base de datos para generar gráficas.
4. **Envío de los datos:** Envía los gráficos generalmente por email, pero soporta también FTP, SFTP y SOCKET.

1.4.1.3. FactorySuite 2000

Consiste en un conjunto de aplicaciones de *software* industrial orientado hacia las aplicaciones de control y MMI. Los principales componentes de la suite son:

- Intouch.
- **IndustrialSQL Server.**
- InControl.
- In Track.
- InBatch.

IndustrialSQL Server: Es una base de datos en tiempo real de alto rendimiento que tiene el propósito de hospedar todos los datos de proceso que se generan en la planta. Combina la fortaleza y la flexibilidad de una base relacional convencional con las particularidades de un sistema de tiempo real. Así, la información correspondiente a los procesos de la planta y la de negocios se integran con facilidad.

1.4.2. Ámbito Nacional

En el ámbito nacional se pueden encontrar sistemas SCADA como por ejemplo:

SW SCADA	Fabricante:
Guardián del ALBA	UCI

Tabla 2: Sistemas SCADA a nivel Nacional.

1.4.2.1. Guardián del ALBA

Fabricante: UCI

Guardián del Alba: Sistema de Supervisión y Control de procesos industriales para los pueblos del ALBA. Es un sistema que permite la supervisión y control automático de los procesos industriales en el sector petrolero, aumentando la eficiencia y seguridad de esta industria. Esta aplicación, desarrollada por especialistas cubanos y venezolanos para PDVSA, utiliza software libre, con lo cual contribuye no sólo a la disminución de los costos sino también a elevar la soberanía tecnológica de esta importante rama de la economía venezolana.

Es un sistema que brinda alta integridad y confiabilidad, garantizando su funcionamiento ininterrumpido, de manera que se garantice la continuidad operacional y el flujo normal de los procesos de exploración, producción y transporte de crudo hasta las instalaciones aguas abajo.

Ventajas

- Es configurable.
- Fácilmente extensible para su operación con nuevos dispositivos.
- Cuenta con una interfaz amigable que apoya la toma de decisión eficaz y un seguimiento efectivo de los procesos en la industria.
- Es muy flexible permitiendo la integración con otros sistemas, su adaptación y extensión.

El sistema tiene un módulo de Base de Datos Histórica (BDH):

El gráfico que se muestra a continuación esquematiza al subsistema historiador, compuesto por un grupo de sub-módulos que permiten la interacción con los demás componentes del SCADA.

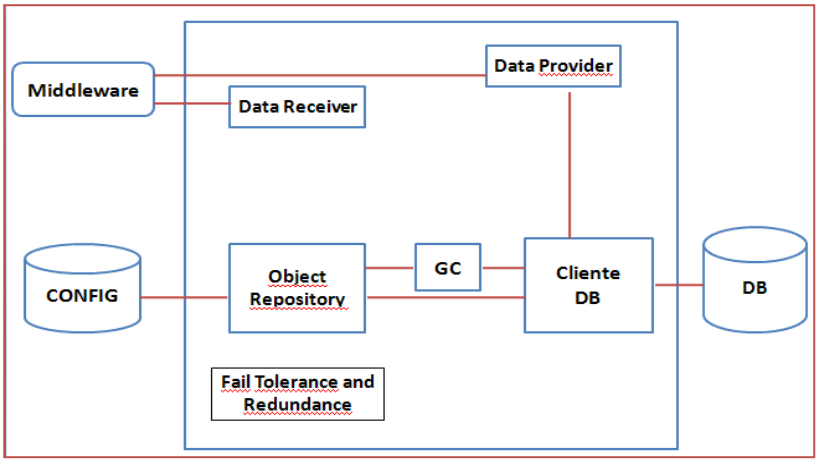

Ilustración 7: Subsistema de BDH.

Servicios brindados por el módulo BDH:

El módulo de BDH es el encargado de almacenar la información del sistema con el objetivo de que esta pueda ser empleada luego en la generación de reportes, tendencias, gestión de producción, etc. Él fue concebido para que brindara 4 servicios fundamentales, que son los que más se encuentran en los SCADA actuales:

- Servicio histórico de variables (puntos).
- Servicio histórico de alarmas.
- Servicio histórico de eventos.
- Servicio histórico de Bitácora

El sistema se encuentra ya en funcionamiento en algunas instalaciones de PDVSA en Venezuela con alentadores resultados.

1.5. Tecnologías a utilizar para obtener la propuesta de solución

1.5.1. Sistema Operativo: GNU/Linux

El Sistema Operativo GNU/Linux es una implementación de libre distribución UNIX para computadoras personales (PC), servidores, y estaciones de trabajo. Este sistema operativo es muy eficiente y tiene un excelente diseño. Es multitarea, multiusuario, multiplataforma y multiprocesador. En las plataformas Intel corre en modo protegido; protege la memoria para que un programa no pueda hacer caer al resto del sistema. Carga sólo las partes de un programa que se usan; comparte la memoria entre programas aumentando la velocidad y disminuyendo el uso de memoria. Usa un sistema de memoria virtual por páginas; utiliza toda la memoria libre para cache. Permite usar bibliotecas enlazadas tanto estática como dinámicamente. Se distribuye con código fuente y soporta redes tanto en TCP/IP como en otros protocolos.

1.5.2. Distribución: UBUNTU

Ubuntu es un sistema operativo construido por un equipo internacional de desarrolladores expertos. Contiene todas las aplicaciones que se necesita para trabajar: un navegador web, suite de oficina, aplicaciones multimedia, la mensajería instantánea. Ubuntu es una alternativa de código abierto para Windows y Office. Es una distribución basada en Debian, con lo que esto conlleva y centrada en el usuario

final y facilidad de uso. Muy popular y con mucho soporte en la comunidad. Ubuntu es y será siempre gratuito. No hay que pagar la licencia para su uso.

Ventajas:

- Es **Gratuito**.
- Tiene muchos programas muy útiles y que lo puedes encontrar muy fácilmente en internet.
- Un punto muy importante es la seguridad, los **Hackers** y/o creadores de virus rara vez atacan a Software de Linux.
- Existe infinidad de Información técnica que servirá de ayuda.
- Carga y realiza tareas con mayor eficiencia que Windows.
- La constante actualización es asombrosa. Existen infinidades de **Distribuciones de Linux**.

1.5.3. Herramienta CASE: Visual Paradigm

Herramientas CASE (Computer Aided/ Assisted Software/ System Engineering): Conjunto de programas que asisten a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del Ciclo de Vida de desarrollo de un Software. (Zorrilla Pantaleón, 2009)

Las herramientas CASE proporcionan al ingeniero la posibilidad de automatizar actividades manuales y de mejorar su visión general de la ingeniería. Al igual que las herramientas de ingeniería y de diseño asistidos por computadora que utilizan los ingenieros de otras disciplinas. Las herramientas CASE ayudan a asegurar la calidad de un producto desde su diseño antes de construirlo.

Herramientas CASE en el mercado actual:
Actualmente existen muchas herramientas CASE en el mundo, algunas de estas son: CoolStuf, Informix, Opal, SystemArchitect, OracleDesigner, Erwin y Visual Paradigm.

Se propone el uso de **Visual Paradigm** el cual tiene las siguientes características:

Visual Paradigm es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software. Ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Permite la Generación de bases de datos, la transformación de diagramas de Entidad Relación en tablas de base de datos, la ingeniería inversa de bases de datos desde

SGBD existentes a diagramas de Entidad-Relación. También proporciona una mayor documentación de la Base de Datos y diagramas de mapeo de relación de objetos. Esta herramienta funciona sobre múltiples plataformas, es un producto de muy buena calidad, así como las imágenes y reportes generados por la misma. Además soporta aplicaciones web, es fácil de instalar y actualizar y es compatible con otras ediciones. Se puede descargar una versión gratuita desde su sitio oficial

1.5.4. SGBD: PostgreSQL

Es un servidor de base de datos relacional libre, liberado bajo la licencia BSD. Es una alternativa a otros sistemas de bases de datos de código abierto (como MySql, Firebird y MaxDB), así como sistemas propietarios como Oracle o DB2.

Algunas de sus principales características son:

- Soporte total del Modelo Relacional de Bases de datos.
- Extensiones propias a SQL para realizar consultas sobre la base de datos.
- Dependencias entre objetos, integridad referencial.
- Acceso concurrente multi-versión (no se bloquean las tablas, ni siquiera las filas, cuando un proceso escribe).
- Estructuras de datos de tipo Array para el dominio de un atributo.
- Definición de Tipos de datos y dominios propios del usuario.
- Definición de esquemas dentro de una base de datos.
- Herencia de Tablas.
- Soporte para funciones y operadores definidos por el usuario.
- Asignación de privilegios para los diferentes objetos (base de datos, tabla, vista, función, esquema)
- Definición de disparadores y reglas para un objeto de la base de datos.
- Existencia de funciones y operadores incorporados al sistema para trabajar con los diferentes tipos de datos soportados.
- Expresiones condicionales dentro de sentencias SELECT.
- Varias interfaces para la programación de clientes.

Ventajas:

- Es uno de los SGBD de código abierto más potente.
- Fue diseñado y creado para tener un mantenimiento y un ajuste menor que el de otros productos.

- Es multiplataforma y extensible.
- Diseñado además para ambientes de alto volumen de información con satisfactorios resultados.
- Disponible totalmente sin coste alguno.

1.5.5. Metodología de desarrollo: Proceso de Desarrollo de Software

La metodología de desarrollo a utilizar será Proceso de Desarrollo de Software desarrollada por el UCID, se logra con la combinación entre los modelos basado en Componentes, el Iterativo y el Incremental. Se emplearán las técnicas para crear prototipos, si son requeridas, para los requerimientos del usuario de los que no existe una visión clara por parte de estos, con el objetivo de desarrollar una definición mejorada de los requisitos del usuario para el sistema.

CAPÍTULO 2: DISEÑO DE LA SOLUCIÓN.

2.1. Introducción

En el presente capítulo se describen actividades importantes para la construcción de la base de datos, entre ellas se encuentran: la selección y argumentación de los requisitos del sistema, la descripción de la arquitectura propuesta. Se construye, además, la propuesta de diseño de la base de datos, para lograr este objetivo se desarrolla el modelo lógico y físico de los datos y se describen las clases obtenidas en ambos modelos. Se mencionan las aplicaciones y tecnologías empleadas como herramientas para resolver el problema científico. Se describen las metodologías empleadas para dar cumplimiento a los objetivos propuestos.

2.2. Descripción de la arquitectura propuesta y su fundamentación

El modelo de Base Datos está basado en uno de sus principales requisitos, o sea, que debe ser genérico y fácilmente adaptable a cualquier Sistema SCADA que se desee implementar. En función de esto y respetando la política de soberanía tecnológica del Centro se utilizan para el desarrollo de la BD el Visual Paradigm como herramienta de modelado, el PostgreSQL como SGBD y el PgAdmin III como el cliente del SGBD.

A continuación se puntualizan de forma teórica algunos aspectos, de la base de datos, que permitirán que su confección sea entendible por todos siguiendo un orden lógico y así, facilitar el trabajo del diseñador de la misma.

Nomenclatura utilizada en la Base de Datos: El nombre de la Base de Datos comienza con la primera letra en mayúscula y el resto en minúscula, en caso de que sea un nombre compuesto se empleará notación *PascalCasing**. Con sólo leerlo se reconoce el propósito de la misma.

Ejemplo: ContMaterial

Nombres de las tablas: El nombre empleado, debe escribirse con todas las letras en minúscula para evitar problemas con el *Case Sensitive* del gestor y con solo leerlo se reconoce el propósito de la misma.

Ejemplo: create table ‘nom_producto’;

Prefijos a utilizar en la creación de tablas

Los prefijos a utilizar en la creación de tablas serán los siguientes:

- **dat**_ Prefijo utilizado en tablas que almacenan la mayor cantidad de características de una entidad.
- **nom**_ Prefijo utilizado en tablas nomencladoras.
- **seg**_ Prefijo utilizado en tablas que almacenan control de acceso, usuarios y opciones de acceso de uno o varios sistemas. (Tablas de Seguridad)
- **conf**_ Prefijo utilizado en tablas que almacenan parámetros de configuración del sistema**.** (Tablas de Configuración)
- **his**_ Prefijo utilizado para tablas que almacenan datos por largos períodos de tiempo y que solo son utilizados para análisis esporádicos. (Tablas Históricos)

Ejemplos:

- nom_producto (Nomeclador).
- seg_usuarios (de seguridad).
- conf_almacen (de configuración).

Campos de las Tablas: El nombre a emplear para los campos se escribe con todas las letras en minúscula y con la mayor claridad posible de manera tal que se entienda su propósito, debe incluir un comentario con su descripción. Si el campo es un identificador debe empezar con id, así como también si el campo es parte de una llave foránea se nombrará con el mismo nombre de donde proviene.

Ejemplo:

- add field ‘idproducto’;
- cantidadembalajes: cantidad de embalajes.
- idcuenta (fk viene de nom_cuenta).

El tratamiento para los distintos tipos de atributos será el siguiente:

- Identificadores (Id): tendrán tipo de dato integer de longitud 10 y autoincrementado.
- Descripciones: tendrán tipo de dato varchar de longitud 80.
- Indicadores: tendrán tipo de dato integer de longitud 1.
- Texto: tendrán tipo de dato varchar de longitud 255.

Nombre de las llaves primarias

El nombre de las restricciones se escribe con minúscula. Para el caso de las claves primarias se utiliza el prefijo pk seguido del símbolo "_", y luego el nombre de la tabla sin especificar el prefijo de la misma.

Ejemplo: pk_cuenta

Nombre de las llaves foráneas

El nombre de las llaves foráneas se escribe con minúscula, comenzará con el prefijo fk seguido del símbolo "_", luego de la tabla de donde viene, seguido del símbolo "_" y por último el nombre de la tabla que la recibe, con solo leerlo se entiende la idea.

Ejemplo: fk_cuenta_comprobante

De esta manera, se entiende que la tabla dat_comprobante contiene una llave foránea que viene desde la tabla nom_cuenta, es importante observar que no se incluyen los prefijos de las mismas.

Nombres de las funciones, triggers, tipos de datos y vistas

Se utilizan los prefijos siguientes para la denominación de funciones, triggers, tipos de datos y vistas, seguidos del símbolo "_".

- **f_** Funciones (Procedimientos Almacenados.)
- **ft_** Funciones de triggers
- **t_** Triggers
- **td_** Tipo de datos
- **v_** Vistas

2.3. Selección y argumentación de los requisitos funcionales y no funcionales del sistema propuesto

2.3.1. Requisitos Funcionales

Según la Metodología Proceso de Desarrollo de Software el propósito de la definición de requisitos es especificar las condiciones o capacidades que el sistema debe cumplir y las restricciones bajo las cuales

debe operar, logrando un entendimiento entre el equipo de desarrollo y el cliente, y especificando las necesidades reales de forma que satisfaga sus expectativas.

La captación y análisis de requerimientos permite caracterizar de forma completa las necesidades de los usuarios de la BD, tanto en los datos como en las operaciones a realizar con los datos. (Caballero, 2009)

Al analizar la problemática que se necesita resolver y porque la BD debe tener como característica principal que sea genérica y adaptable a cada uno de los sistemas SCADA que se necesite construir se ha determinado que la misma no cuenta con Requisitos Funcionales.

2.3.2. Requisitos no funcionales

Los requerimientos no funcionales forman una parte significativa de la especificación de requisitos de un proyecto pues, en muchos casos, son fundamentales en el éxito del producto. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable, casi siempre son estas propiedades las que pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación. A continuación se describen los principales requerimientos relacionados con la base de datos.

1. **Confiabilidad (CON)**
   - 1.1. Deben establecerse los mecanismos necesarios para el restablecimiento de la base de datos ante fallos de comunicación u otros, los tiempos mínimos para ello no deben exceder las 6 hrs.
   - 1.2. Deben garantizarse la realización de copias de respaldo de todos los datos periódicamente.
2. **Rendimiento (REN)**
   - 2.1. Los tiempos de respuesta y velocidad de procesamiento de la información serán rápidos, no mayores de 5 segundos para las actualizaciones.
   - 2.2. **Escalabilidad:** El sistema debe ser capaz de mantener un rendimiento y una estabilidad adecuada al gestionar amplios volúmenes de datos.
3. **Soporte (SOP)**
   - 3.1. Se necesita un servidor de bases de datos que soporte grandes volúmenes de datos.
4. **Restricciones de diseño (RED)**
   - 4.1. Emplear como servidor de bases de datos PostgreSQL v8.2.
   - 4.2. Diseñar la BD de forma tal que sea Genérica.
5. **Políticos Culturales (CUL)**

5.1. La base de datos solo podrá ser utilizado en territorio cubano y por las entidades autorizadas por el Ministerio de las FAR.

5.2. El producto debe respetar los términos empleados normalmente por los especialistas en el tema de la esfera que se automatiza.

6. **Seguridad (SEG)**

6.1. La información que se maneje en el sistema estará protegida de acceso no autorizado y divulgación, a partir de los diferentes roles de los usuarios que empleen en la BD.

6.2. El Sistema Gestor de Base de Datos escogido debe presentar facilidades de administración de roles y usuarios restringiendo el acceso a los datos.

6.3. Políticas de seguridad por usuarios y roles

6.4. **Disponibilidad:** Se necesita una base de datos con alta disponibilidad.

7. **Software (SFT)**

7.1. Un servidor de base de datos PostgreSQL – 8.2.

7.2. El PgAdmin III como cliente para la BD.

8. **Hardware (HDW)**

8.1. Por el lado cliente:

- Procesador: 1.40 GHZ
- RAM: 512 Mb
- Tarjeta de Red: 1

8.2. Del lado del servidor de Base de datos:

- Tarjeta de Red: 1
- Procesador: 3.00 GHZ
- RAM: 1GB
- Disco duro: 500 GB
- UPS: 1
- Lector de CD: 1

2.4. Diseño de la Base de Datos

2.4.1. Patrones utilizados

Durante la realización del diagrama de clases persistentes y el diagrama entidad – relación de la base de datos fueron utilizados varios patrones de diseño dentro de los que se encuentran los patrones de Brown, a continuación se describe la utilización de los mismos.

Nombre: Representación de objetos como tablas

Problema: ¿Cómo representar un objeto en un esquema de base de datos relacional?

Solución: Definir una tabla para cada clase de objetos persistentes. Los atributos de la clase que son tipos primitivos serán las columnas de las tablas.

Nombre: Representación de relaciones como tablas

Problema: ¿Cómo representar una relación en un esquema de base de datos relacional?

Solución: Al diseñar una base de datos se debe tener en cuenta cómo representar una relación en un esquema de base de datos relacional, este patrón de diseño propone soluciones para los distintos tipos de relaciones que posee este modelo de datos.

Para las relaciones de uno a uno o uno a muchos:

- Colocar una clave ajena en la tabla de cardinalidad uno, para representar la relación de los objetos.
- O crear una tabla asociativa para registrar los identificadores de cada uno de los objetos de la relación.

Para las relaciones muchos a muchos:

- Crear una tabla asociativa para registrar los identificadores de cada uno de los objetos de la relación.

Nombre: **Identificador de objetos**

Problema: ¿Cómo mantener la identidad de un objeto en una base de datos relacional? Cada identidad de objetos individuales debe ser presentada en la base de datos.

Solución: Asignar un identificador independiente (OID) a cada objeto persistente. Se recomienda el uso de un generador de secuencias si hay alguno disponible en la base de datos ya que los identificadores son generalmente enteros largos que garantizan que sean únicos para una clase de objetos en particular.

Nombre: **Referencia de llaves foráneas**

Problema: ¿Cómo representar objetos que referencian otros objetos que no son de tipos de datos base?

Solución: Asignar a cada objeto un identificador único. Luego añadir una columna por cada variable de instancia que no tenga un tipo de dato base o sea una colección. En esa columna almacenar el identificador del objeto referenciado y declarar la columna como llave foránea.

2.4.2. Descripción de la Solución

Para dar respuesta a los objetivos de esta investigación se propone el diseño de una BD genérica que permita adaptarse a cualquier sistema SCADA que se necesite, y que la misma sea lo suficientemente escalable para que si se necesitan hacer cambios los mismos sean los mínimos posibles y que no afecten la relación con el resto de las tablas del modelo. Para lo anterior se realizaron tareas como las siguientes:

- Se crearon todas las entidades apoyadas en los conceptos genéricos que representan ellas en la vida real.
- Se crearon entidades de tipo Nomenclador, que garantizan que una vez que se necesite agregar nuevos tipos esto sea posible.
- La BD fue creada de forma tal que si se necesita realizar cambios para adaptarla a algún SCADA en específico sea solamente agregar las nuevas tablas y no entren en conflicto con el resto pues existe un bajo acoplamiento entre las tablas del modelo.

Por otra parte, el modelo se realizo con la herramienta CASE Visual Paradigm, a partir de ahí se generó el Diagrama Entidad-Relación Lógico y luego Físico, además del diagrama de Clases Persistentes. Posteriormente se generó el script de la BD que permitió crear la misma en el Servidor de PostgreSQL con la ayuda del cliente PgAdmin III.

2.4.3. Diagrama de clases persistentes

Para el diseño de la base de datos del sistema, se parte de un diagrama de clases persistentes, definiendo la persistencia como la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Una vez conformado dicho diagrama se puede obtener el modelo de datos. A continuación se muestra el diagrama de clases persistentes.

Ilustración 8: Diagrama Clases Persistente Parte 1

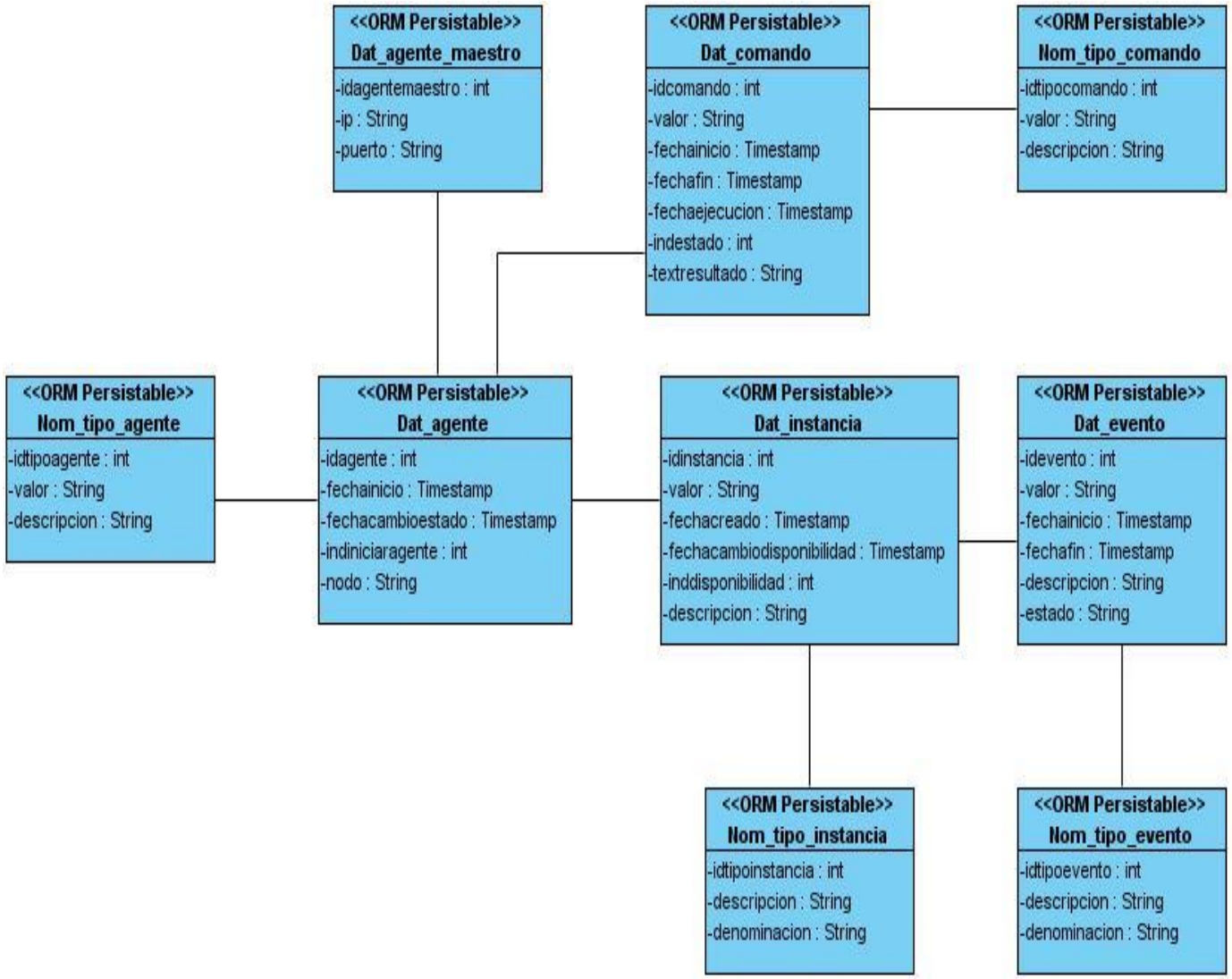

Ilustración 9: Diagrama Clases Persistente Parte 2

2.4.4. Descripción de las clases persistentes

A continuación se representa la descripción de las clases persistentes modeladas anteriormente, de cada una de ellas se especifica el nombre, el tipo de clase, atributos y el tipo de cada uno de estos.

Nombre: Dat_agente_maestro	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idagentemaestro	Int
ip	String
puerto	String

Tabla 3: Descripción de la Clase Dat_agente_maestro

Nombre: Nom_tipo_agente	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idtipoagente	Int
valor	String
descripcion	String

Tabla 4: Descripción de la Clase Nom_tipo_agente

Nombre: Dat_agente	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idagente	Int
fechainicio	Timestamp
fechacambioestado	Timestamp
indiniciaragente	Int
nodo	String

Tabla 5: Descripción de la Clase Dat_agente

Nombre: Dat_comando	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idcomando	Int
valor	String
fechainicio	Timestamp
fechafin	Timestamp
fechaejecucion	Timestamp
indestado	Int
textresultado	String

Tabla 6: Descripción de la Clase Dat_comando

Nombre: Nom_tipo_comando	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idtipocomando	Int
valor	String
descripcion	String

Tabla 7: Descripción de la Clase Nom_tipo_comando

Nombre: Dat_instancia	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idinstancia	Int
valor	String
fechacreado	Timestamp
fechacambiodisponibilidad	Timestamp
inddisponibilidad	Int
descinstancia	String

Tabla 8: Descripción de la Clase Dat_instancia

Nombre: Nom_tipo_instancia	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idtipoinstancia	Int
denominacion	String
descripcion	String

Tabla 9: Descripción de la Clase Nom_tipo_instancia

Nombre: Nom_tipo_evento	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idtipoevento	Int
denominacion	String
descripcion	String

Tabla 10: Descripción de la Clase Nom_tipo_evento

Nombre: Dat_evento	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idevento	Int
valor	String
fechainicio	Timestamp
fechafin	Timestamp
descripcion	String
estado	String

Tabla 11: Descripción de la Clase Dat_evento

Nombre: His_evento	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idhisevento	Int
idevento	Int
idinstancia	Int
idtipoevento	Int
valor	String
fechainicio	Timestamp
fechafin	Timestamp
descripcion	String
estado	String

Tabla 12: Descripción de la Clase His_evento

Nombre: His_instancia	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idhisinstancia	Int
idinstancia	Int
idtipoinstancia	Int
valor	String
fechacreado	Timestamp
fechacambiodisponibilidad	Timestamp
inddisponibilidad	Int
descripcion	String

Tabla 13: Descripción de la Clase Hist_Instancia

Nombre: His_agente	
Tipo de Clase: Entidad	
Atributos:	**Tipo:**
idhisagente	Int
idagente	Int
idagentemaestro	Int
idtipoagente	Int
valor	String
fechainicio	Timestamp
fechacambioestado	Timestamp
nodo	String

Tabla 14: Descripción de la Clase His_agente

2.4.5. Diagrama Entidad – Relación de la Base de Datos

El diagrama entidad-relación representa el estado final de las tablas de la base de datos.

Un modelo de datos es una serie de conceptos que pueden utilizarse para describir un conjunto de datos y las operaciones para manipularlos. (Marqués, 2001)

El modelo entidad-relación es el modelo conceptual más utilizado para el diseño conceptual de bases de datos, está formado por un conjunto de conceptos que permiten describir la realidad mediante un conjunto de representaciones gráficas y lingüísticas. (Marqués, 2001)

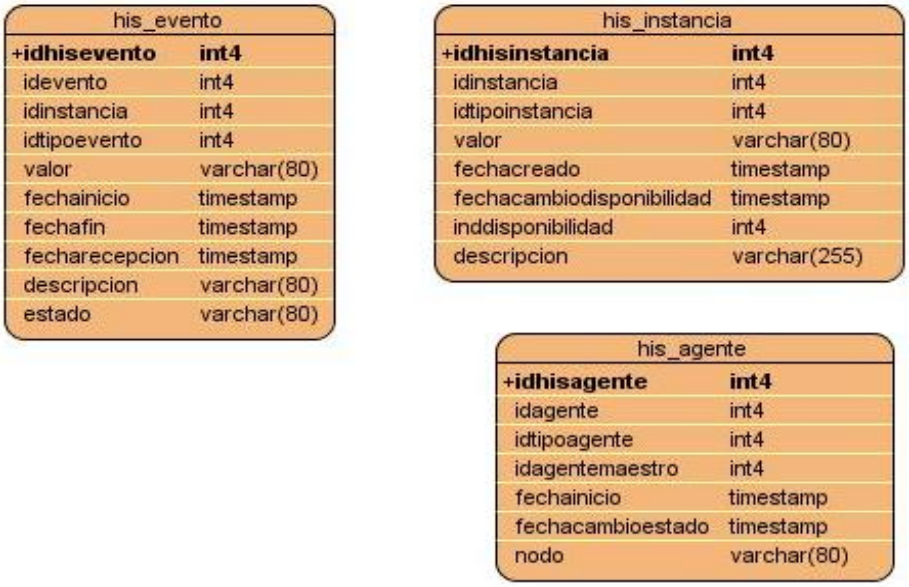

Ilustración 10: Diagrama Entidad-Relación Parte 1

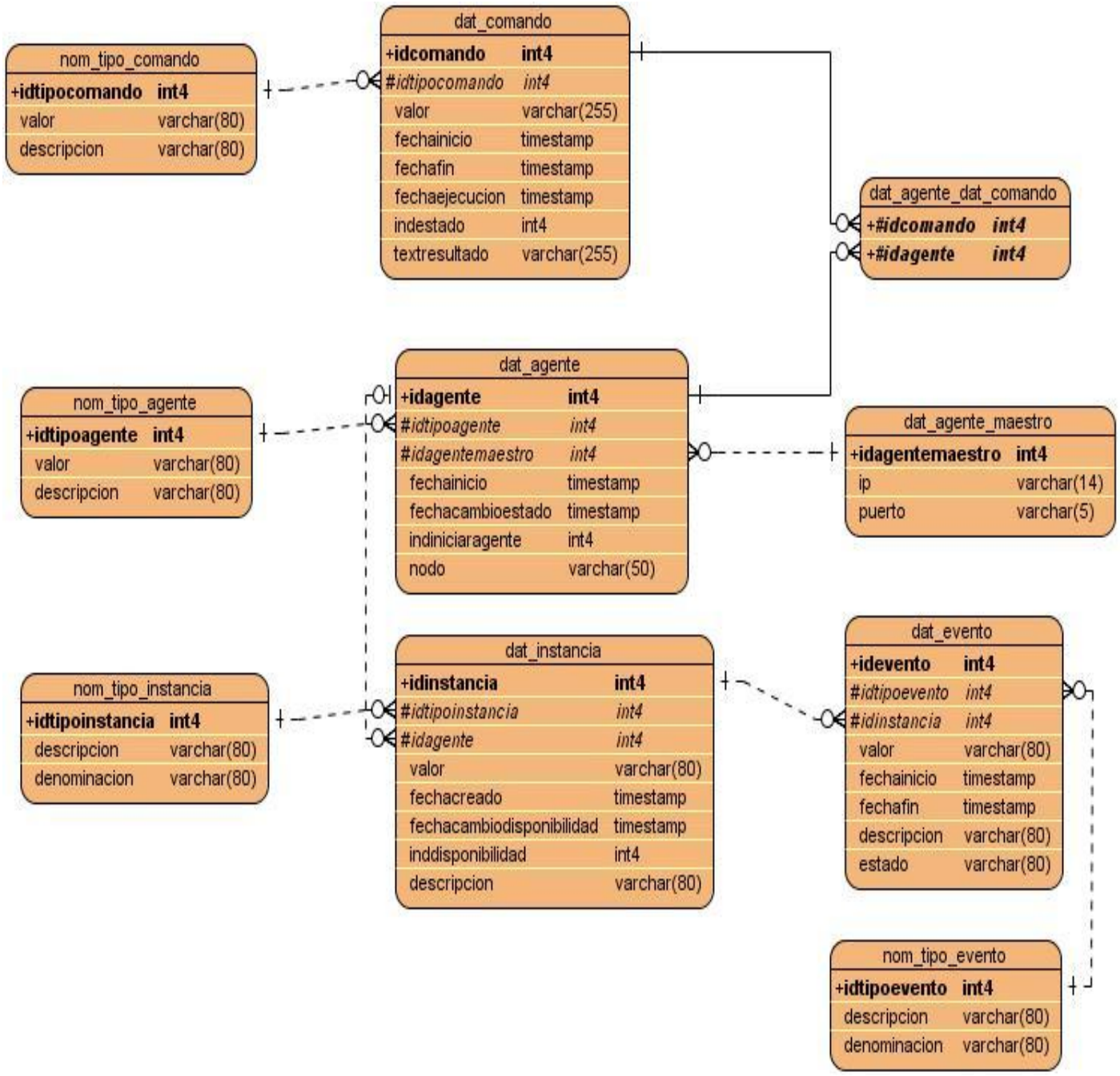

Ilustración 11: Diagrama Entidad-Relación Parte 2

2.4.6. Descripción de las tablas del diagrama entidad – relación

Nombre: **dat_agente_maestro**		
Descripción: Agentes maestros instalados en el sistema.		
Atributo	**Tipo**	**Descripción**
idagentemaestro	Integer (10)	Identificador del Agente Maestro
ip	Varchar (14)	Dirección IP del Agente Maestro
puerto	Varchar (5)	Puerto del Agente Maestro

Tabla 15: Descripción de la Tabla dat_agente_maestro

Nombre: **nom_tipo_agente**		
Descripción: Tipo de agentes que puedan supervisar el sistema.		
Atributo	**Tipo**	**Descripción**
idtipoagente	Integer (10)	Identificador del tipo de agente
valor	Varchar (80)	Valor del Tipo de Agente
descripcion	Varchar (80)	Descripción

Tabla 16: Descripción de la Tabla nom_tipo_agente

Nombre: **dat_agente**		
Descripción: Agentes instalados en el sistema para el control de las instancias. (ej.: Autómatas (VL))		
Atributo	**Tipo**	**Descripción**
idagente	Integer (10)	Identificador del agente
idtipoagente	Integer (10)	Identificador del tipo de agente
idagentemaestro	Integer (10)	Identificador del Agente Maestro
fechainicio	Timestamp	Indica la fecha en que se inició el agente
fechacambioestado	Timestamp	Indica la fecha en que el agente cambia de estado o de configuración
indiniciaragente	Integer (1)	Iniciar este agente cuando el Agente Maestro comience a trabajar 0-No 1-Si
nodo	Varchar (80)	Indica el nodo que pertenece

Tabla 17: Descripción de la Tabla dat_agente

Nombre: **dat_comando**		
Descripción: Comandos o tareas a ser procesadas por el agente.		
Atributo	**Tipo**	**Descripción**
idcomando	Integer (10)	Identificador de la Comando
idtipocomando	Integer (10)	Identificador del Tipo de Comando
valor	Varchar (80)	Valor que indica el comando que debe ejecutarse
fechainicio	Timestamp	Fecha de inicio del Comando
fechafin	Timestamp	Fecha de fin del Comando
fechaejecucion	Timestamp	Fecha de ejecución del Comando
indestado	Integer (1)	Estado: 0:pendiente 1:en proceso 2:OK 3:error
textresultado	Varchar (255)	Texto con el resultado de ejecutar la tarea

Tabla 18: Descripción de la Tabla dat_comando

Nombre: **nom_tipo_comando**		
Descripción: Tipo de Tareas.		
Atributo	**Tipo**	**Descripción**
idtipocomando	Integer (10)	Identificador del Tipo de Tarea
valor	Varchar (80)	Valor que indica el tipo de comando
descripcion	Varchar (80)	Descripción

Tabla 19: Descripción de la Tabla nom_tipo_comando

Nombre: **dat_instancia**		
Descripción: Instancias en el sistema susceptibles de ser controlados. (pueden ser sensores, actuadores, entre otros)		
Atributo	**Tipo**	**Descripción**
idinstancia	Integer (10)	Identificador de la instancia
idtipoinstancia	Integer (10)	Identificador del Tipo de instancia
idagente	Integer (10)	Identificador del agente
valor	Varchar (80)	Valor de la Instancia (ej.: si es de temperatura 10 grados)
fechacreado	Timestamp	Fecha de creación de la Instancia
fechacambiodisponibilidad	Timestamp	Fecha de cambio de disponibilidad de la Instancia
inddisponibilidad	Integer (1)	Indicador de disponibilidad(0:disponible, 1: no disponible)
descripcion	Varchar (80)	Descripción

Tabla 20: Descripción de la Tabla dat_instancia

Nombre: **nom_tipo_instancia**		
Descripción: Tipo de Instancias. (ej.: sensor de temperatura, de humedad relativa, etc.)		
Atributo	**Tipo**	**Descripción**
idtipoinstancia	Integer (10)	Identificador del Tipo de instancia
denominacion	Varchar (80)	Nombre del sensor (ej.: temperatura)
descripcion	Varchar (80)	Descripción

Tabla 21: Descripción de la Tabla nom_tipo_instancia

Nombre: **nom_tipo_evento**		
Descripción: Tipo de eventos.		
Atributo	**Tipo**	**Descripción**
idtipoevento	Integer (10)	Identificador del Tipo de evento
denominacion	Varchar (80)	Nombre del Evento (Urgente, No urgente)
descripcion	Varchar (80)	Descripción

Tabla 22: Descripción de la Tabla nom_tipo_evento

Nombre: **dat_evento**		
Descripción: Eventos enviados por los agentes maestro al servidor central.		
Atributo	**Tipo**	**Descripción**
idevento	Integer (10)	Identificador del Evento
idtipoevento	Integer (10)	Identificador del Tipo de evento
idinstancia	Integer (10)	Identificador de la Instancia
valor	Varchar (80)	Valor del Evento (ej.: la temperatura alcanzo su valor máximo)
fechainicio	Timestamp	Fecha de inicio del Evento
fechafin	Timestamp	Fecha de fin del Evento
descripcion	Varchar (80)	Descripción
estado	Varchar (80)	Estado del evento (ej.: recibido, solucionado, etc.)

Tabla 23: Descripción de la Tabla dat_evento

Nombre: **his_evento**		
Descripción: Histórico de los Eventos ocurridos.		
Atributo	**Tipo**	**Descripción**
idhisevento	Integer (10)	Identificador del Evento Histórico
idevento	Integer (10)	Identificador del Evento
idinstancia	Integer (10)	Identificador del Tipo de evento
idtipoevento	Integer (10)	Identificador de la Instancia
valor	Varchar (80)	Valor del Evento (ej.: la temperatura alcanzo su valor máximo)
fechainicio	Timestamp	Fecha de inicio del Evento
fechafin	Timestamp	Fecha de fin del Evento
descripcion	Varchar (80)	Descripción
estado	Varchar (80)	Estado del evento (ej.: recibido, solucionado, etc.)

Tabla 24: Descripción de la Tabla his_evento

Nombre: **his_instancia**		
Descripción: Histórico de las Instancias.		
Atributo	**Tipo**	**Descripción**
idhisinstancia	Integer (10)	Identificador de la Instancia Histórica
idinstancia	Integer (10)	Identificador del Instancia
idtipoinstancia	Integer (10)	Identificador del Tipo de Instancia
valor	Varchar (80)	Valor de la Instancia (ej.: si es de temperatura 10 grados)
fechacreado	Timestamp	Fecha de creación de la Instancia
fechacambiodisponibilidad	Timestamp	Fecha de cambio de disponibilidad de la Instancia
inddisponibilidad	Int	Indicador de disponibilidad(0:disponible, 1: no disponible)
descripcion	Varchar (80)	Descripción

Tabla 25: Descripción de la Tabla his_instancia

Nombre: **his_agente**		
Descripción: Histórico de los Agentes instalados.		
Atributo	**Tipo**	**Descripción**
idhisagente	Integer (10)	Identificador del agente histórico
idagente	Integer (10)	Identificador del agente
idtipoagente	Integer (10)	Identificador del tipo de agente
idagentemaestro	Integer (10)	Identificador del Agente Maestro
fechainicio	Timestamp	Indica la fecha en que se inició el agente
fechacambioestado	Timestamp	Indica la fecha en que el agente cambia de estado o de configuración
nodo	Varchar (80)	Indica el nodo que pertenece

Tabla 26: Descripción de la Tabla his_agente

CAPÍTULO 3: VALIDACIÓN DEL DISEÑO REALIZADO.

3.1. Introducción

En el presente capítulo se realiza una validación de la propuesta de solución para el diseño de la Base de Datos Genérica para los Sistemas SCADA, que se obtuvo en el capítulo anterior. Además, se describe el tratamiento dado a importantes aspectos para el área de la base de datos como son la normalización, redundancia, integridad y seguridad de la información en el diseño propuesto, elementos donde se han centrado los esfuerzos de los diseñadores. El objetivo fundamental de este capítulo es validar el diseño realizado con aspectos teóricos y funcionales.

3.2. Validación teórica del diseño

Para realizar un buen diseño de una BD hay que tener en cuenta varios aspectos importantes como: la integridad de los datos, la normalización del diseño, la redundancia de información, la trazabilidad de las acciones que se realizan y sobre todo la seguridad en la BD, la cual es la base principal de los aspectos anteriormente mencionados, permitiendo controlar el acceso a los datos y que los mismos no se corrompan como resultado de acciones no controladas. No basta solamente con realizar el diseño de una BD, se tienen que tener en cuenta estos aspectos para garantizar la consistencia, integridad y seguridad de la misma.

3.2.1. Integridad

La integridad de los datos se refiere a la corrección y completitud de los datos en una BD. Al modificar el contenido de la BD con sentencias INSERT, DELETE o UPDATE, la integridad de los datos puede verse afectada añadiendo datos no válidos, modificando datos existentes tomando un valor incorrecto o eliminando datos que al hacerlo violan alguna regla.

La integridad de los datos se contempla en diferentes niveles. Las restricciones de dominio, transacciones y entidades definen las reglas para el mantenimiento de la integridad de las relaciones individuales. Las relaciones de integridad referencial aseguran que se mantienen las asociaciones necesarias entre las relaciones. Las restricciones de integridad de la BD gobiernan la misma como un todo y las restricciones de integridad de transacciones controlan la forma en que se manipulan los datos, dentro de una o entre múltiples BD. (Riordan, 2000)

3.2.1.1. Integridad de dominio

Una restricción de integridad de dominio es una regla que define valores válidos para los atributos de las diferentes tablas de una BD. Puede ser necesario definir más de una restricción de dominio para describir por completo un dominio. Cuanto más sean las reglas que se definan mejor será el funcionamiento de la base de datos.

Existe integridad de dominio básica, como no poder introducir letras en campos donde se va a almacenar números. Estas normas o reglas de integridad de dominio pueden indicar que campos son necesarios tener obligatoriamente con valores, para que la base de datos no tenga datos sin conectar en el caso de tener relaciones o dependencias entre tablas.

Algunas herramientas que asegurar la integridad de dominio son: definir los tipos de datos correctamente para cada campo, definir campos no nulos (NOT NULL) para evitar resultados inconsistentes, definir reglas (CHECK), que pueden ser aplicadas a nivel de tabla y columna y que restringen los valores durante las inserciones y modificaciones.

Para realizar estas validaciones PostgreSQL presenta los obligadores (CONSTRAINS), que se encargan de chequear (CHECK) el cumplimiento de una o varias condiciones que deben cumplir el o los campos de una tabla y disparan excepciones, en caso de ser violadas. También existen los disparadores (TRIGGERS), los cuales, pueden validar campos de más de una tabla a la vez; como su nombre lo indica disparan una excepción, en caso de no cumplirse alguna(s) de las condiciones que valida. Estos hacen uso de funciones especiales, las cuales validan las condiciones que debe cumplir una tupla para ser aceptada como correcta e insertarla en la BD o retornan valor nulo (NULL), en caso de violarse alguna condición.

Para garantizar la integridad de dominio de los datos de la BD, no se definió como nulos (NULL) ninguno de los campos de la BD. Durante el proceso de confección del modelo se ha tenido en cuenta la restricción del tipo de datos, el formato a través de las restricciones CHECK.

3.2.1.2. Integridad Referencial

La integridad referencial se puede decir que es la más importante de las integridades dentro del área de la base de datos, la misma da la medida de qué tan exacta o correcta es la información que se maneja. Implica que en todo momento los datos sean correctos y sin repeticiones innecesarias evitando la pérdida de datos y relaciones incorrectas. Establece que una tupla en una relación que haga referencia a otra,

deberá referirse a un valor existente en esa relación. Se define para asegurarse que las filas relacionadas entre tablas, no desaparezca ni varíe, cuando se lleve a cabo una modificación en los datos. Con esta integridad se limita la actividad que puede realizar un usuario sobre la base de datos.

Para mantener esta integridad en la BD diseñada se utilizó las llaves foráneas, las cuales obligan a que los valores introducidos en las columnas marcadas por esta restricción correspondan a valores en la tabla referenciadas, así como permite realizar acciones en caso de actualización o eliminación de los valores a los que se hacen referencia, para esto, se definió que al realizar la actualización o eliminación de datos, las mismas se realizarían en cascada.

El SGBD utilizado, PostgreSQL, maneja de manera muy positiva la integridad referencial, el mismo asegura en todo momento que, en un diseño elaborado correctamente, las referencias por llaves foráneas sean las correctas.

3.2.1.3. Integridad de Entidades

Las restricciones de entidades aseguran la integridad de las entidades que son modeladas por el sistema. Un caso de este tipo de restricción es el de las llaves primarias para cada tabla. En el caso de la BD que se está analizando cada entidad tiene definida su llave primaria, que no puede ser nula ni se puede repetir. Definir correctamente la llave primaria, es de vital importancia, ya que esto representa la principal herramienta que utiliza el servidor de BD para seleccionar la información que se necesite, evitando la duplicidad de registros, y el lanzamiento de excepciones por errores en los registros. Para garantizar la integridad de entidad de los datos de la BD, se definió una llave primaria para cada tabla, las mismas son de tipo entero y generador incremental y no permiten valores nulos (NOT NULL).

3.2.1.4. Integridad de Transacciones

Define los estados por los que una tupla puede pasar válidamente, como son: introducido, pendiente, seleccionado, enviado, cancelado y terminado. Está encargada de asegurar que el estado de una determinada tupla, no pase de un estado inicial a uno final, sin haber pasado por los estados intermedios. En el caso de la BD genérica para sistemas SCADA no se encuentran restricciones de este tipo.

De manera general estas son las consideraciones que se han tenido en cuenta durante el diseño de la propuesta, asegurando que el mismo cumpla con la integridad de la información requerida, debido fundamentalmente a la importancia que se le atribuye a este aspecto dentro de la base de datos.

3.2.2. Normalización de la base de datos

La normalización de la BD es el proceso mediante el cual se transforman datos complejos a un conjunto de estructuras de datos más pequeñas, que además de ser más simples y más estables, son más fáciles de mantener. (Domínguez Vaillant & Miranda Gutiérrez, 2007)

La normalización es un proceso mediante el cual se descompone una relación que no cumple con una serie de requisitos en relaciones más pequeñas que si las cumplan. Tiene como objetivo eliminar redundancias y anomalías de inserción, eliminación y actualización. (Hernández Castellanos & Pérez García, 2007)

Para resumir se puede decir que la normalización es el proceso que permite eliminar la redundancia de los datos y evitar los problemas al insertar, eliminar y actualizar los datos, es decir, optimizar el trabajo con la información almacenada. Es un proceso en el cual se va comprobando el cumplimiento de una serie de reglas o restricciones por parte de un esquema de relación, donde cada regla que se cumple, aumenta el grado de normalización del esquema de relación. Si una regla no se cumple, el esquema de relación se debe descomponer en varios esquemas de relación que sí la cumplan por separado.

En este proceso existen varios niveles de normalización, pero los tres primeros son los más usados. Las reglas de normalización se relacionan entre ellas de forma dependiente, lo que indica que para que una base de datos se encuentre en una determinada forma normal, primeramente tiene que cumplir con las reglas de los niveles inferiores de normalización.

Los niveles de normalización existentes son (Eduardo, 2003):

- Primera Forma Normal (1NF).
- Segunda Forma Normal (2NF).
- Tercera Forma Normal (3NF).
- Forma Normal Boyce-Codd.
- Cuarta Forma Normal.
- Quinta Forma Normal o Forma Normal de Proyección-Unión.
- Forma Normal de Proyección-Unión Fuerte.
- Forma Normal de Proyección-Unión Extra Fuerte.
- Forma Normal de Clave de Dominio.

Una base de datos se considera en un nivel de normalización **N** si todas sus tablas están normalizadas a la forma **N.**

Cada nuevo nivel acerca más el diseño a una BD verdaderamente relacional. Cada una tiene sus propias reglas. Durante el proceso de la normalización se fue comprobando que cada relación (tabla) cumpliera con un conjunto de reglas basadas en la clave primaria y las dependencias funcionales. Se llevaron a cabo una serie de pasos donde cada paso responde a una forma normal.

Primera forma normal (1FN):

La primera formal normal se definió para prohibir los atributos multivaluados, los atributos compuestos y sus combinaciones, es decir, establece que las columnas repetidas deben eliminarse y colocarse en tablas separadas. Lo anterior indica que todos los atributos deben tener valores atómicos. Esta forma normal constituye el primer nivel en el proceso de normalización y es en ella donde se eliminarán los atributos multivaluados, los atributos compuestos y sus combinaciones.

La BD genérica para sistemas SCADA cumple con la primera forma normal, ya que los datos de las relaciones involucradas son atómicos, o sea, no son ni multivaluados, ni compuestos, ni ninguna combinación de estos.

Segunda forma normal (2FN):

La segunda forma normal plantea que todos los atributos que no forman parte de la llave primaria, deben depender de manera total de los atributos que forman la llave primaria. Se dice que una entidad está en 2FN si está en 1FN y si sus atributos no llaves (ni primarias ni candidatas) son funcional y completamente dependientes de la llave primaria. Esta forma normal asegura que todas las columnas que no son llave sean completamente dependientes de la llave primaria, es decir, establece que todas las dependencias parciales se deben eliminar y separar dentro de sus propias tablas.

Se analizaron todas las tablas de la BD genérica para sistemas SCADA se comprobó que todas cumplen con la segunda forma normal, excepto la tabla Tb_Comando donde se encontró que el atributo valor depende del id_tipo_comando, se determinó eliminar el atributo valor y obtener ese valor de la tabla N_Tipo_Comando a través del identificador de esta tabla.

Luego de esta modificación toda la BD está en 2FN pues ya está en primera forma normal y cada atributo no primo de la relación es completamente dependiente de la clave primaria. Con la 2FN se evitan anomalías a la hora de realizar actualizaciones y se controlan la mayoría de los problemas de lógica.

Tercera forma normal (3FN):

La tercera forma normal plantea que la relación debe estar en segunda forma normal y que cada atributo de la relación que no está contenido dentro de la llave primaria dependa solo de la llave primaria y no de ningún otro atributo. Una relación está en tercera forma normal sí, y sólo sí, está en 2FN y además, cada atributo que no está en la clave primaria no depende transitivamente de la clave primaria. En esta forma normal se elimina cualquier dependencia transitiva. Una dependencia transitiva es aquella en la cual las columnas que no son llave son dependientes de otras columnas que tampoco son llave. Una tabla está normalizada en esta forma si todas las columnas que no son llave son funcionalmente dependientes por completo de la llave primaria y no hay dependencias transitivas.

Una vez que se tiene la BD en 2FN se pasó a eliminar las dependencias transitivas para obtener la 3FN con el objetivo de eliminar las redundancias que aún quedaban de la 2FN para evitar sufrir anomalías frente a las actualizaciones.

Se analizaron las tablas de la BD, las mismas ya se encontraban en 2FN y se detectó que una de ellas, **Tb_Instancia**, no estaba en 3FN pues el atributo fecha_cambio_disponibilidad dependía del atributo ind_disponibilidad, y el mismo no forma parte de la llave primaria, por lo que se procedió a eliminar los dos atributos de la tabla **Tb_Instancia**, se creó una nueva tabla, **N_Disponibilidad**, donde se agregó el id_disponibilidad y el atributo valor que seria 1 si está disponible y 0 en caso contrario. Luego se creó una relación entre estas dos tablas creándose una nueva tabla con los id de las mismas y además la fecha del cambio de disponibilidad de la instancia.

Después de este proceso, la BD genérica para sistemas SCADA, ha quedado normalizada hasta la 3FN. La BD está en 1FN puesto que se puede asegurar que cada tupla contiene exactamente un valor para cada atributo de las tablas de la BD. También se cumple que los esquemas de relación están en 2FN, porque se encuentran en 1FN, y además todos los atributos que no son claves en las tablas, dependen totalmente de la clave primaria. Y finalmente se puede plantear que se encuentra en 3FN, porque está en 2FN, y además no existen dependencias transitivas entre llaves candidatas y atributos no primos.

Después del proceso de Normalización el Diagrama Entidad-Relación sufrió los cambios explicados anteriormente quedando de la siguiente manera:

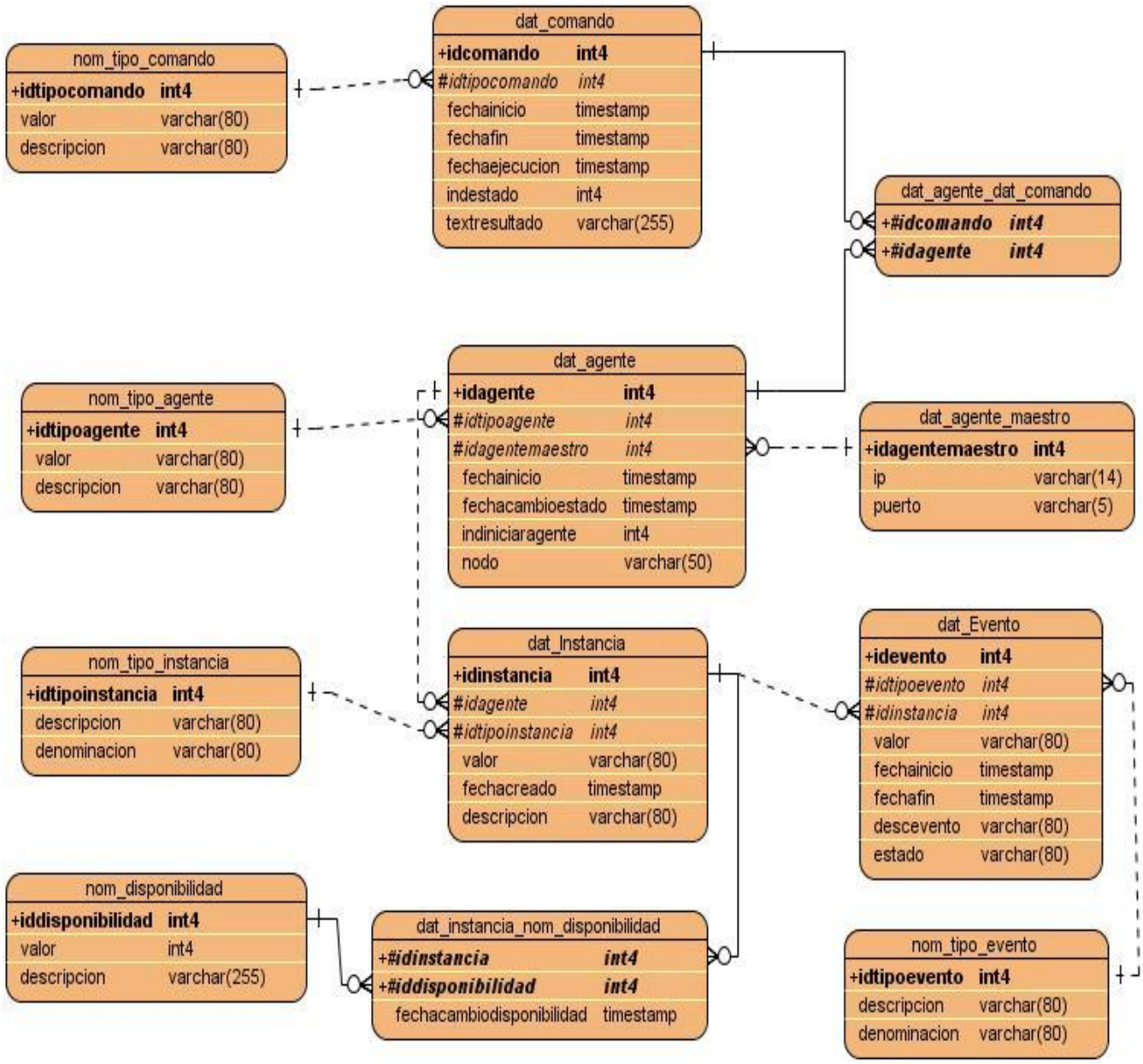

Ilustración 12: Diagrama Entidad-Relación Normalizado Parte 1

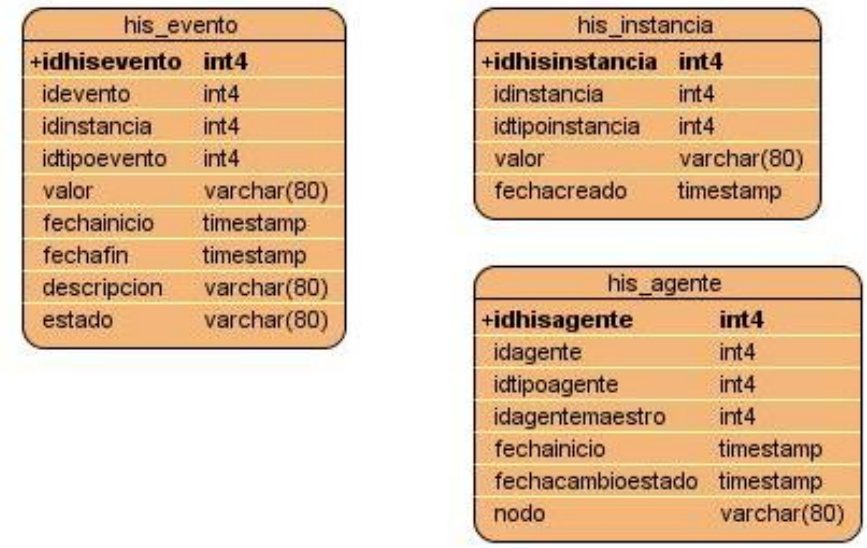

Ilustración 13: Diagrama Entidad-Relación Normalizado Parte 2

3.2.3. Análisis de redundancia de la información

La redundancia de datos es aquella información duplicada o almacenada varias veces en la misma base de datos. Esto dificulta la tarea de modificación de datos y es el motivo más frecuente de inconsistencia de los mismos. Además, requiere un mayor espacio de almacenamiento, que influye en un mayor coste y tiempo de acceso a los datos. La redundancia siempre debe evitarse, aunque en proyectos grandes es imposible evitarla al 100%, lo que a veces es deseable por cuestiones de rendimiento. Con un buen diseño de una base de datos se logrará evitar la aparición de información redundante.

Después de llevar la BD genérica para sistemas SCADA a 3FN, es decir, al normalizarla, queda libre de redundancias, ya que se elimina completamente la presencia de datos repetidos innecesariamente. La normalización es una técnica que elimina en gran medida esta dificultad, aunque en bases de datos grandes es imposible eliminarla al ciento por ciento. Además de la normalización se aplicó el uso de nomencladores para todos aquellos datos comunes dentro de la base de datos.

3.2.4. Análisis de la seguridad de la base de datos

La seguridad es un punto esencial en las bases de datos para evitar ataques e impedir cualquier acceso no autorizado, con la intensión de modificar, usar y/o difundir información almacenada en las bases de datos, ya sea por error del usuario o por mala intención. La información que almacena una BD, está en

constante riesgo de sufrir ataques que puedan provocar su modificación o pérdida, por ello es de vital importancia velar la seguridad de la misma, protegiéndola, en un primer momento, contra accesos no autorizados o cualquier acción que puedan violar la integridad de los datos o la confidencialidad de los mismos. En un segundo momento es necesario proteger los datos que se almacenan en la BD, para lo cual se deben realizar salvas.

Las técnicas de copia de seguridad y restauración, compatibles con los sistemas de bases de datos deben garantizar la preservación de los datos ante cualquier imprevisto o falla del sistema. Aún así, el administrador de base de datos debe definir procedimientos para recuperar la información perdida. La Base de Datos genérica para sistemas SCADA cuenta con políticas de gestión de copias de seguridad que establece que se debe realizar las mismas una vez al día.

El administrador de la base de datos desempeña un papel importante en la seguridad de la misma porque es quien otorga privilegios a los usuarios y los clasifica, así como a los datos. Las órdenes de los administradores de bases de datos incluyen las siguientes acciones: creación de cuentas, concesión de privilegios y revocación de privilegios. La creación de cuentas sirve para controlar el acceso al SGBD en general, la concesión y revocación de privilegios para controlar autorizaciones discrecionales. Estas autorizaciones constituyen un importante mecanismo de seguridad.

El SGBD a utilizar para diseñar la base de datos genérica para sistemas SCADA, PostgreSQL, incluye formas de restringir el acceso al sistema. Esta función se denomina control de acceso y se pone en práctica creando cuentas de usuarios y contraseñas para que el SGBD controle el proceso de entrada al sistema.

3.2.5. Trazabilidad de las acciones

La trazabilidad es la capacidad que posee un sistema para rastrear, reconstruir o establecer relaciones entre objetos monitoreados, para identificar y analizar situaciones específicas o generales en los mismos.

Al referirse a la trazabilidad se puede decir entonces que existe la posibilidad de conocer o establecer, entre la información que se maneja de un sistema, quién ha realizado qué, y poder llevar el sistema a un estado anterior. La trazabilidad puede realizarse sobre tablas o sobre procesos. Incluso pueden crearse trazabilidades sobre errores o conflictos de la aplicación, (log).

Los logs son un excelente mecanismo que tienen los SGBD para seguir las trazas en bases de datos. Estos son usados para registrar datos o información sobre (quién, qué, cuándo, dónde) convirtiéndose en un registro oficial de eventos durante un período de tiempo en particular. De esta manera, el administrador de BD tendrá un efectivo archivo de evidencias que podrá utilizar en cualquier momento que necesite.

El SGBD que se utilizó para implementar la BD genérica para sistemas SCADA, el PostgreSQL, almacena en la carpeta pg_log, los registros, en archivos de texto, con la información del día, hora, usuario, máquina de origen, consulta, datos y tiempo que invirtió en la consulta, lo que permite llevar un control estricto de que se está haciendo en la BD.

3.3. Validación Funcional

Para comprobar el correcto funcionamiento de la BD genérica, es necesario realizarle pruebas antes de dar por terminado el proceso de diseño y creación de la misma. El propósito de estas pruebas es simular una carga de producción real y observar cómo se comporta la base de datos bajo cargas intensivas. Esto permite solucionar los problemas de rendimiento, antes de poner la BD en marcha. El objetivo general es ejecutar las consultas que se supone que sean las más utilizadas y percibir cómo responde el sistema a ellas.

3.3.1. Llenado voluminoso e inteligente de la base de datos.

Las pruebas de volumen centran su trabajo en poblar a la BD de grandes cantidades de información para determinar si existe algún momento donde la misma alcance sus límites de almacenamiento y cause fallas en el sistema, identificándose así la carga máxima que puede manejar la base de datos en un período dado. En el mundo existen algunas herramientas para el llenado voluminoso de BD.

Para la realización de pruebas de llenado voluminoso e inteligente a la BD se utilizó la herramienta EMS Data Generator 2005 para PostgreSQL, la cual permite la generación de datos para una o varias tablas a la vez, definiendo para cada campo el rango de valores admisibles, dependiendo del tipo, además de la cantidad de tuplas que se desean generar, validando de forma automática la integridad referencial, ya que genera los datos que provienen de otras tablas para evitar errores. La aplicación permite llenar la BD de forma sencilla y rápida. La generación de los datos es de forma aleatoria e incremental.

Para realizar el llenado voluminoso se ha configurado la herramienta utilizada para generar 10 000 tuplas en todas las tablas concurrentemente como se muestra en la siguiente figura:

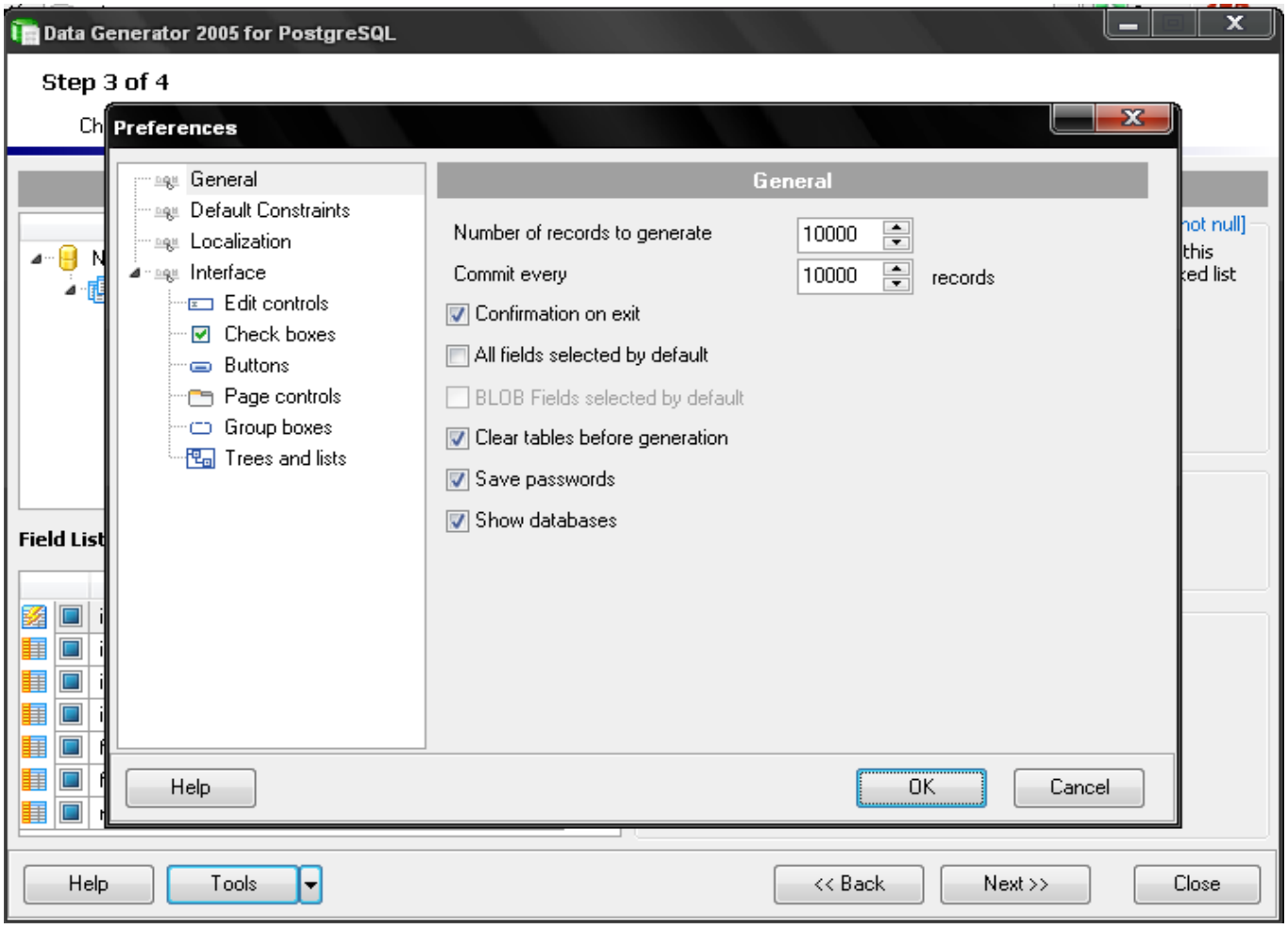

Ilustración 14: Configuración del Data Generator.

Además, se configuraron los diferentes tipos de datos para que estuvieran dentro de los valores admisibles. Luego se pasó a generar los datos, siendo generados 150 000 tuplas en un tiempo de 7 minutos con 21 segundos.

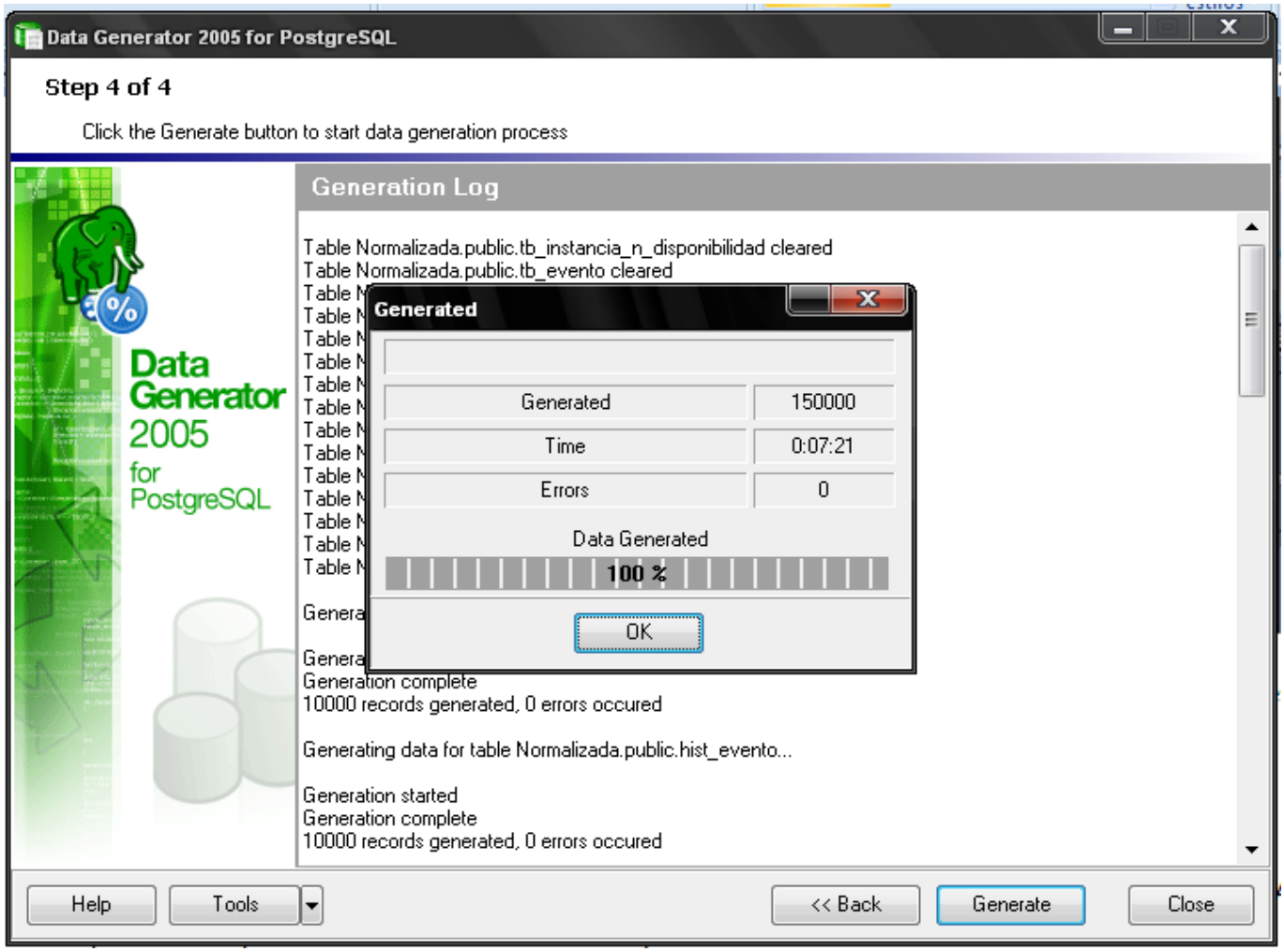

Ilustración 15: Resultados Obtenidos luego de la Generación de los datos

3.3.2. Pruebas de carga intensiva.

Las pruebas de carga intensiva centran su trabajo en probar el funcionamiento de la BD través del grado de concurrencia de las consultas. En esta se comprueba el comportamiento de la BD frente a las consultas más frecuentes que se realizarán a la misma, verificando que esta se comporte adecuadamente. Para la realización de este tipo de pruebas a la BD genérica para sistemas SCADA se utilizó la herramienta EMS SQL Query para PostgreSQL, con el objetivo de probar el correcto funcionamiento de la misma, así como los tiempos de ejecución. Las consultas generadas para la realización de pruebas, se escogen de acuerdo con las operaciones que se esperan sean las más utilizadas una vez implantada la BD.

Seleccionar el valor de una instancia y el tipo de la misma.

```
SELECT dat_instancia.valor, nom_tipo_instancia.denominacion

FROM (dat_instancia JOIN nom_tipo_instancia ON

    ((nom_tipo_instancia.idtipoinstancia = dat_instancia.idtipoinstancia)));
```

Se obtuvo un resultado de 10000 filas en un tiempo de 63 ms.

Obtener todos los eventos ocurridos en un día y una hora determinados.

```
SELECT his_evento.idtipoevento, his_evento.valor, his_evento.fechainicio

FROM his_evento

WHERE (his_evento.fechainicio = '2010-05-02 10:28:06'::timestamp without time zone);
```

Se obtuvo un resultado de 32 filas en un tiempo de 78 ms.

Se realizaron pruebas a la BD, pero el resultado de estas nunca puede tomarse como resultados reales, aunque dan un aproximado al mismo. Aún así, la implantación y utilización de una BD, está marcada por varios factores entre ellos: cantidad de usuarios a conectarse y el grado de concurrencia de las solicitudes hechas por lo mismo, los cuales hay que tener en cuenta siempre.

3.4. Resultados

Luego de haber concluido el desarrollo del módulo de base de datos genérica para los sistemas SCADA se puede decir que se obtuvieron los resultados esperados:

- Diagrama de clases persistentes refinado donde las relaciones entre las tablas fueron construidas adecuadamente
- Diseño del modelo lógico normalizado hasta la 3 FN, y sin información redundante en el mismo.
- Implementación del modelo físico de la base de datos.

CONCLUSIONES

Una vez concluida la presente investigación para el desarrollo del módulo de base de datos para un SCADA, ha sido cumplido su objetivo general.

En la investigación se arribó a las siguientes conclusiones:

1. Del estudio realizado sobre los distintos SGBD se decidió usar el PostgreSQL para desarrollar el módulo de BD y que se realizaría una BD de tipo Histórica.
2. La BD diseñada tiene como principales características que la misma es genérica y fácilmente adaptable para cualquier sistema SCADA.
3. Se desarrolló el módulo de base de datos genérico para los SCADA de las FAR que permite tener un histórico de los valores obtenidos de los sensores implantados en el sistema.
4. El diseño de la Base de Datos propuesto, ha sido validado teórica y funcionalmente para garantizar que el mismo presenta la calidad necesaria y esperada. Para validación teórica se ha tenido en cuenta importantes criterios como son la normalización, integridad y seguridad de los datos, con los que ha cumplido satisfactoriamente. Como parte de la validación funcional, este diseño ha sido implementado utilizando las herramientas propuestas en la investigación, al realizarse las pruebas de su funcionamiento se obtuvieron buenos resultados en las mismas.

RECOMENDACIONES

Al culminar el desarrollo del trabajo de Diploma se recomienda:

- Aplicar la propuesta de Base de Datos a los SCADA de las FAR.
- Robustecer el diseño antes de aplicarlo a un SCADA específico.
- Dar continuidad al trabajo desarrollando una Base de Datos en Tiempo Real.

BIBLIOGRAFÍA

1. Brunet Granado, Y. (2009). *Modelo lógico y físico de la base de datos para la Plataforma de Transmisión Abierta de Radio y Televisión.* La Habana.
2. Chavarría Meza, L. E. (2007). *SCADA SYSTEM´S & TELEMETRY.* Mexico. Disponible en: http://www.aiu.edu/applications/DocumentLibraryManager/upload/SCADA%20System%C2%B4s%20&%20Telemetry.pdf
3. Cuba, lo más novedoso en informatización. (8 de 3 de 2009). *Juventud Rebelde* . Disponible en:http://www.juventudrebelde.cu/cuba/2009-03-08/cuba-lo-mas-novedoso-en-informatizacion/
4. Daneels, A. & Salter ,W. (1999). *WHAT IS SCADA?* CERN, Geneva, Switzerland. Disponible en: http://www.elettra.trieste.it/ICALEPCS99/proceedings/papers/mc1i01.pdf
5. Díez Barrero, Domingo. *SIMULACIÓN DE SCADA CON COMUNICACIÓN GSM.* Universidad de Huelva. España. Disponible en: http://www.uhu.es/fernando.gomez/transydat_archivos/Escada_GSM.PDF
6. Domínguez Vaillant, A. E., & Miranda Gutiérrez, D. (2007). *Sistema de Manejo de Datos de Ensayos Clínicos: Diseño e implementación de la Base de Datos.* La Habana.
7. Ezequiel Rozic, S. (2004). *Base de Datos y su aplicación con SQL.* Buenos Aires: MP Ediciones. Disponible en: http://bibliodoc.uci.cu/pdf/reg03438.pdf
8. Fernández Padrón, Y. (2009). *Plataforma VideoWeb.Rol Diseñador ddee Basee de Datos.* La Habana.
9. Hernández Castellanos, Y., & Pérez García, W. (2007). *Diseño de Bases de Datos para la Intranet 2.0.* La Habana.
10. Hernández León, R. A., & Coello González, S. (2002). *EL PARADIGMA CUANTITATIVO DE LA INVESTIGACIÓN CIENTIFICA.* Ciudad de la Habana, Cuba: Editorial Universitaria.
11. *Informática Habana*. (2009). Recuperado el 28 de 1 de 2010, Disponible en: http://www.informaticahabana.cu/?q=en/node/787
12. Johnston, Peter D. (2009). *ASSESSING CRITICAL INFRASTRUCTURE. A CASE STUDY AT AN ENERGY FACILITY.* Canadá. Disponible en: http://www.rebootconference.com/energy2009/presentations/PeterJohnston-TECH-CriticalinfrastructurePresentation-Reboot.ppt

13. National Instruments (2008).*IMPLEMENTACIÓN DE SISTEMAS SCADA CON LABVIEW.* Ciudad de México, México. Disponible en: ftp://ftp.ni.com/pub/branches/latam/Mexico/Implementacion%20de%20Sistemas%20SCADA%20con%20LabVIEW.pdf
14. Orallo Hernández, José. La disciplina de los sistemas de Bases de Datos. Historia, Situación Actual y Perspectivas. Valencia: s.n., 2002. Disponible en: http://palomo.usach.cl/Docs/BD/JHernandezO-La_disciplinade_los_sistemas.pdf
15. Sauver, Joe St. Ph.D. (2004). *SCADA SECURITY AND CRITICAL INFRASTRUCTURE.* University of Oregon Computing Center. Disponible en: http://www.uoregon.edu/~joe/scadaig/infraguard-scada.ppt
16. Tello, R. Y. (2003). Base de Datos en la Ingeniería y los Negocios. *Industrial Data , 6*, págs. 79-82. Disponible en: http://redalyc.uaemex.mx/redalyc/pdf/816/81606109.pdf

REFERENCIAS BIBLIOGRÁFICAS

Brunet Granado, Y. (2009). *Modelo lógico y físico de la base de datos para la Plataforma de Transmisión Abierta de Radio y Televisión.* La Habana.

Caballero, A. (2009). *ISeries AS400*. Recuperado el 05 de 03 de 2010, de http://www.recursos-as400.com/dissenybasededades.shtml

Castillo, C. (2008). *Sistemas Gestores de Base de Datos.*

Chavarría Meza, L. E. (2007). *SCADA SYSTEM´S & TELEMETRY.* Mexico.

Corrales, L. (2007). *DISEÑO Y CONSTRUCCIÓN DE UN SISTEMA DE ADQUISICIÓN DE. DATOS PARA EL ANÁLISIS DE LOS ELECTROOCULOGRAMAS* (Vol. Parte 2). Quito, Ecuador.

Cuba, lo más novedoso en informatización. (8 de 3 de 2009). *Juventud Rebelde* .

Definición.de. (2008). Recuperado el 5 de 2 de 2010, de http://definicion.de/modelo-de-datos/

Domínguez Vaillant, A. E., & Miranda Gutiérrez, D. (2007). *SIMDEC Sistema de Manejo de Datos de Ensayos Clínicos: Diseño e implementación de la Base de Datos.* La Habana: Universidad de las Ciencias Informáticas.

Domínguez Vaillant, A. E., & Miranda Gutiérrez, D. (2007). *Sistema de Manejo de Datos de Ensayos Clínicos: Diseño e implementación de la Base de Datos.* La Habana.

Eduardo. (31 de marzo de 2003). *MySQL Hispano.* Recuperado el 10 de abril de 2010, de http://www.mysql-hispano.org/page.php?id=16&pag=1.

Fernández Padrón, Y. (2009). *Plataforma VideoWeb.Rol Diseñador ddee Basee de Datos.* La Habana.

Ferrari, J. P. (2005). *Sistemas de control distribuido.*

Hernández Castellanos, Y., & Pérez García, W. (2007). *Diseño de Bases de Datos para la Intranet 2.* Ciudad de La Habana: Universidad de las Ciencias Informáticas.

Hernández Castellanos, Y., & Pérez García, W. (2007). *Diseño de Bases de Datos para la Intranet 2.0.* La Habana.

Herrera Vázquez, M. (2008.). *Introducción a la arquitectura del "Guardián del ALBA". [En linea].*

Interactive Programmers Community. (1 de 11 de 2007). *La Web del Programador*. Recuperado el 17 de 2 de 2010, de http://www.lawebdelprogramador.com

Marqués, M. (2001). *Apuntes de Ficheros y Bases de Datos.* España.

Mendiburu Díaz, H. (2008). *SISTEMAS SCADA.*

Mesquida Peña, M. G., & Basulto Uribe, E. (2009). *Propuesta de diseño de la Base de Datos Histórica de la gestión académica de Akademos v_2.0.*

National Comunications System. (2004). *Supervisory Control and Data.* Virginia.

Navarrete Navarrete, C. B. (2004). *Introducción a las bases de datos.*

Núñez Camallea, N. L. (2004). *Gestión de Base de Datos con ADO.NET.* Colombia: Editorial Científico-Técnica.

Olivares Rojas, Juan Carlos;Tecnológico de Monterrey. (2009). *Aplicaciones Mecatrónicas distribuidas.*

Riordan, R. M. (2000). *Diseño de Base de Datos Relacionales Con Access y SQL Server* (1ra edición ed.). España: Mcgraw-Hill.

Universidad de Oviedo. *SCADA - Supervisory Control and Data Acquisition.*

Zorrilla Pantaleón, M. E. (2003). *Introducción a las Bases de Datos.*

Zorrilla Pantaleón, M. E. (2009). *Modelos de datos.*

GLOSARIO DE TÉRMINOS

1. **BD:** Base de Datos.
2. **BDH:** Base de Datos Histórica.
3. **BSD:** es el acrónimo en inglés de Distribución de Software Berkeley (Berkeley Software Distribution) y se utiliza para identificar un sistema operativo derivado del sistema Unix nacido a partir de las aportaciones realizadas a ese sistema por la Universidad de California en Berkeley.
4. **CPU:** es el acrónimo en inglés de Unidad Central de Procesamiento (Central Processing Unit). Es la componente central de la computadora donde se realizan las funciones lógicas y aritméticas básicas.
5. **DFD:** Diagrama de Flujo de Datos es una representación gráfica del "flujo" de datos a través de un sistema de información.
6. **FTP:** es el acrónimo en inglés de Protocolo de Transferencia de Archivos (File Transfer Protocol) es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red, basado en la arquitectura cliente-servidor.
7. **Hacker:** Persona con especial habilidad con los ordenadores que puede resolver por sí mismo los problemas que su funcionamiento le plantee.
8. **Herramientas CASE:** es el acrónimo en inglés de Ingeniería de Software Asistida por Ordenador (Computer Aided Software Engineering). Son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el coste de las mismas en términos de tiempo y de dinero.
9. **Kernel:** Núcleo de un sistema operativo, es decir, bloque de código con la parte central del funcionamiento y arranque del sistema.
10. **LDD:** Lenguaje de Definición de Datos.
11. **MMI:** es el acrónimo en inglés de Interfaz Hombre-Máquina **(**Mano Machine Interface). Son las pantallas gráficas que el operador está usando para supervisar y controlar el proceso.
12. **MTU:** es el acrónimo en inglés de Unidad Terminal Maestra (Máster Terminal Unit). Es el sistema electrónico o de computación que adquiere todos los datos procedentes de las unidades terminales remotas y que la presenta de una forma a una RTU para ejecutar una acción de control remoto.

13. **ODBC:** es el acrónimo en inglés de Conexión abierta a base de datos (Open Data Base Connectivity). Forma de conexión a bases de datos, independientemente del lenguaje o programa que estemos utilizando. Cada fabricante provee su propia librería con las características de conexión a las mismas.
14. **OLAP:** es el acrónimo en inglés de **Procesamiento analítico en línea** (*On-Line Analytical Processing*). Es una solución utilizada en el campo de la Inteligencia empresarial cuyo objetivo es agilizar la consulta de grandes cantidades de datos. Para ello utiliza estructuras multidimensionales que contienen datos resumidos de grandes Bases de datos o Sistemas Transaccionales (OLTP). Se usa en informes de negocios de ventas, marketing, informes de dirección, minería de datos y áreas similares.
15. **OLTP:** es el acrónimo en inglés de **Procesamiento de Transacciones En Línea** (OnLine Transaction Processing) es un tipo de sistemas que facilitan y administran aplicaciones transaccionales, usualmente para entrada de datos y recuperación y procesamiento de transacciones.
16. **Open source:** Código abierto es el término con el que se conoce al software distribuido y desarrollado libremente.
17. **PC:** es el acrónimo en inglés de Computadora personal (Personal Computer). Es una computadora digital personal basada en un microprocesador y diseñada para ser utilizada por una sola persona a la vez.
18. **PDVSA:** Petróleos de Venezuela Sociedad Anónima es una empresa estatal venezolana que se dedica a la explotación, producción, refinación, mercadeo y transporte del petróleo venezolano.
19. **PLC:** es el acrónimo en inglés de Controladores Lógicos Programables (Programmable Logic Controller). Dispositivo electrónico muy usado en automatización industrial. Un PLC controla la lógica de funcionamiento de maquinas, plantas y procesos industriales, procesan y reciben señales digitales y analógicas y pueden aplicar estrategias de control.
20. **RPC**: es el acrónimo en inglés de Llamada a Procedimiento Remoto (Remote Produce Call). Es un protocolo que permite que los programas llamen a subrutinas que se ejecutan en un sistema remoto, incluyendo códigos de retorno y variables predefinidas para soportar el procesamiento distribuido.
21. **RTDB:** es el acrónimo en inglés de Base de Datos en Tiempo Real (Real Time Database).

22. **RTU:** es el acrónimo en inglés de Unidades de Terminal Remota (Remote Terminal Unit). Es un dispositivo basado en microprocesadores, el cual permite obtener señales independientes de los procesos y enviar la información a un sitio remoto donde se procese.
23. **SCADA:** es el acrónimo en inglés de **Adquisición de Datos y Control de Supervisión** (Supervisory Control and Data Acquisition).
24. **SFTP:** es el acrónimo en inglés de Protocolo Seguro de Transferencia de Archivos (Secure File Transfer Protocol). Es un protocolo de red que proporciona la funcionalidad necesaria para la transferencia y manipulación de archivos sobre un flujo de datos fiable.
25. **SGBD: Sistema Gestor de Bases de Datos** (en inglés database management system, abreviado DBMS) es un tipo de software específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.
26. **Socket:** Interfaz de comunicación que ofrece un mecanismo de comunicación general entre dos procesos cualquiera que pertenezcan a un mismo sistema o a dos sistemas diferentes.
27. **Software:** Es el conjunto de los programas de cómputo, procedimientos, reglas, documentación y datos asociados que forman parte de las operaciones de un sistema de computación.
28. **TCP/IP:** es el acrónimo en inglés de Protocolo de Control de Transmisión/Protocolo Internet (Transmission Control Protocol/Internet Protocol). Es un conjunto de protocolos de red en los que se basa Internet y que permiten la transmisión de datos entre redes de computadoras.
29. **UML:** es el acrónimo en inglés de Lenguaje Unificado de Modelado (Unified Modeling Language) es un lenguaje de modelado de sistemas de software. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.
30. **XML:** es el acrónimo en inglés de Lenguaje de Marcas Extensible (Extensible Markup Language), es un metalenguaje extensible de etiquetas. Permite definir la gramática de lenguajes específicos. XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades.