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

UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD 9

DISEÑO DE LA BASE DE DATOS DEL SISTEMA DE INFORMACIÓN DE PERFORACIÓN DE POZOS (SIPP)

Autor: Maikel Hugo Álvarez González
Tutor: Lic. Yanet Espinal Martín
Co-Tutor: Ing. Dennis Meriño Menadier

Junio, 2009

DECLARACIÓN DE AUTORÍA

Declaro que soy el único autor del trabajo titulado:

Diseño de la Base de Datos del Sistema de Información de Perforación de Pozos (SIPP).

y autorizo 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 2009.

_____ _____

Firma del Autor Firma del Tutor

Agradecimientos

Primeramente quisiera agradecer a la Revolución Cubana y a Fidel por darme la oportunidad de estudiar en una Universidad como esta.

A mis padres hacer de mí todo lo que soy y siempre haber confiado en todo momento, se que un día como hoy estoy cumpliendo uno de sus mayores anhelos.

A mis abuelos y a mis tías por haberme cuidado siempre y darme la mejor educación que se podría dar, ustedes han sido para mí otros padres.

A mis primas por todo lo que han sido conmigo, por siempre cuidar de mí.

A mi querida novia por su apoyo incondicional y saber comprenderme en todo momento.

A mi futura niña, la cual ha sido mi mayor inspiración.

Agradecer a todos mis profesores que ayudaron en mi preparación como profesional, en especial a Dashiel, Alain, guías, compañeros, amigos y hermanos de nuestro antiguo grupo 9108.

A mis tutores que hicieron posible el desarrollo de este trabajo y por todo lo que hicieron por mí. Dennis, de veras has sido más que un amigo durante el transcurso de este año.

A todos mis amigos y compañeros que siempre estuvieron ahí cuando más lo necesité, en especial a los tres Karel, Henry, Frank, Leandris, Edimir, Rogelio, Raul (el calvo), Wilde, compañeros fieles de recreación y de fiestas en la Universidad.

AGRADECIMIENTOS

A todas mis amigas y compañeras que siempre me ayudaron en los momentos precisos, en especial a Aimé, Nancy, Maryslenis, Karolay, Sulay, Alicia, Yaneysi, Roxayne y Mislaydis.

A todos mis vecinos por aguantarme en el barrio y por brindarme todo su apoyo y darme los mejores consejos desde allá, en especial a Kleisver, Pocholo, Raúl, Mireya, Sara, Maritza, Madelaine, Yaneisy, Kirenia, las Yanet, Niurka, Sara, Amelia, Bertha, Xiomara, Javier, Daris, Ailec, en fin a todos, discúlpenme si se me quedó algunos.

A mi suegra por quererme y confiar en mí.

A mi gran amigo Pavel y a su querida madre Maurene, por haberme ayudado a entrar en esta Universidad.

Agradecer a todo el equipo de trabajo y compañeros del laboratorio por toda la amistad brindada durante el transcurso del año, en especial a los David, Yordan, Michel, HI, Jorge, Noel, Leo (casi te mato cuando me borraste la base de datos), Aniuvys por las cosas lo que hicimos juntos.

A mis compañeros de grupo y de cuarto, tanto del antiguo como el nuevo, Raudel, Yusbel, Yermain, Israel, Pimienta, Alejandro, Michel, Aníbal, Chirino, Raudel, en fin a todos.

A todas las personas que una vez me preguntaron "y la tesis, como va".

Agradezco a las personas que me vieron pasar sin dejar de ser vistos.

Agradezco la palmada oportuna, el buen consejo y los argumentos.

Dedicatoria

A la niña que estoy esperando.

A mis padres, por cumplir su deseo más añorado.

A mi novia por quererme tanto y siempre estar ahí en todo momento.

A toda mi familia por todo lo que hicieron por mí.

A todos mis amigos que se preocuparon y me ayudaron a la realización de este trabajo.

A todas las personas que esperaron este gran momento.

Pensamiento

Figura 1. Aristóteles

La virtud, como el arte, se consagra constantemente a lo que es difícil de hacer, y cuanto más dura es la tarea, más brillante es el éxito.

ARISTÓTELES

RESUMEN

La empresa de Cuba-Petróleo (CUPET) tiene entre sus propósitos automatizar los principales procesos productivos que en ella se realizan, logrando así un mejor flujo y control en la gestión de la información.

En ella existen dos empresas fundamentales, La Dirección de Intervención y Perforación de Pozos (DIPP) y el Centro de Investigaciones del Petróleo (CEINPET), las cuales en integración con la Universidad de Las Ciencias Informáticas (UCI), se han propuesto desarrollar un sistema que ayude a elevar la eficiencia en la gestión y control de la información, auxiliado de una base de datos, que permita almacenar la información generada. Eliminando los problemas de redundancia y desactualización de los datos que son gestionados.

Con la intención de lograr los objetivos propuestos se realizó una investigación de cómo se desarrollan los principales procesos en las empresas mencionadas, además de las principales tecnologías, técnicas y herramientas empleadas en el modelado, diseño y gestión de las bases de datos (BD). Teniendo en cuenta los resultados de los estudios previos se procede a desarrollar el diseño de la base de datos propuesta, la cual fue sometida a un proceso de fundamentación teórica y validación funcional, para analizar si cumplía con los requisitos a tener en cuenta de un buen diseño y arquitectura.

ÍNDICE

ÍNDICE

INTRODUCCIÓN

En la actualidad los softwares de bases de datos han experimentado un auge extraordinario a raíz de la creciente informatización de las empresas, lo que ha traído consigo rapidez, efectividad, menos costos en los procesos de grandes flujos de información, para satisfacer las necesidades a la hora de optimizar servicios y productos. Ante esta notable demanda de soluciones informáticas han surgido multitud de gestores de bases de datos, estos son programas que permiten manejar la información de modo sencillo y que presentan excelentes funcionalidades para el desarrollo y el manejo de las bases datos.

Actualmente La Oficina Central de Exploración Producción de Cuba perteneciente a la empresa Cuba-Petróleo (CUPET) se encuentra inmersa en un proceso de expansión de sus principales procesos productivos, las expectativas que se plantean sobre las posibilidades reales del país de la exploración, perforación y producción de petróleo en las aguas profundas del Golfo de México, han propiciado un ambiente de trabajo favorable para la realización de convenios de trabajos entre la UCI (Universidad de las Ciencias Informáticas) y CUPET, debido a la imperiosa necesidad de informatizar sus principales procesos de producción, que elevaría grandemente la eficiencia y competitividad de esta empresa en el marco internacional.

La DIPP (Dirección de Intervención y Perforación de Pozos) radicada en La Empresa de Producción y Extracción de Petróleo del CENTRO (EPEPC), controla todas las operaciones y demás actividades que se realizan en los pozos de petróleo en perforación e intervención. Hoy en día el proceso de gestión, control y flujo de información se realiza manualmente lo que trae consigo:

- Redundancia en la información.
- Errores e inconsistencia en los partes y reportes emitidos.
- Desactualización de la información.
- Bajo control del orden de almacenamiento.
- Difícil acceso a la información.
- Pérdida y mal cuidado de la misma.

Por otro lado utilizan algunos software para obtener información del proceso de perforación, estos son de corto alcance, pues de manera integral no gestionan toda la información requerida durante la perforación

de los pozos, además de funcionamiento limitado, ya que no se cuenta en muchos casos con la clave o llave para su perfecta instalación y utilización, lo que se puede considerar como software pirateados.

Estas instituciones requieren de una solución inmediata a los problemas antes mencionados para evitar de esta forma el deterioro y pérdida de la información, logrando su perfecto almacenamiento y una adecuada automatización del proceso de gestión de información.

De esta manera queda evidenciado como **problema a resolver:** Inexistencia de un adecuado diseño de la Base de Datos del Proyecto Sistema de Información de Perforación de Pozos (SIPP), para almacenar todos los datos generados en el proceso de gestión de la información durante la perforación de los pozos petroleros.

Con vista a la solución del problema se plantea como **objeto de estudio:** Diseño de Base de Datos, aplicando éste a nuestro **campo de acción**: Base de Datos para el Sistema de Información de Perforación de Pozos para Empresas Petroleras, con **el objetivo general**: Diseñar el sistema de Base Datos del proyecto Sistema de Información de Perforación de Pozos para Empresas Petroleras (SIPP).

Teniendo como **hipótesis**: Un diseño adecuado y flexible de la base de datos del Sistema de Información de Perforación de Pozos para Empresas Petroleras, facilitará un mejor desarrollo de la aplicación y permitirá garantizar el control y la gestión de toda la información relacionada con la perforación de pozos para las Empresas Petroleras.

Para darle solución a los objetivos planteados se proponen las siguientes tareas de investigación:

- Investigar los diferentes procesos relacionados a la perforación de cualquier pozo.
- Realizar un estudio sobre las diferentes herramientas de modelado de base datos existentes.
- Realizar un estudio de los diferentes gestores de base de datos para desarrollar el sistema.
- Realizar Modelo de Objeto Entidad-Relación.
- Realizar todas las especificaciones de los artefactos del modelo de datos.
- Realizar pruebas de volumen a la base de datos para validarla funcionalmente.

Los **métodos científicos** a utilizados se desglosan a continuación:

- **Métodos Teóricos:**

- ❖ Análisis histórico-lógico: Se emplea para conocer cómo se realizan los procesos de gestión de información en las empresas DIPP, CEINPET y en algunos pozos de petróleo existentes en Cuba.
- ❖ Análisis y síntesis: Se utiliza para el análisis de las tendencias y tecnologías actuales de bases de datos a nivel mundial.
- ❖ Modelación: Se utiliza para la representación del modelo lógico y físico de la base de datos.

- ➢ **Métodos Empíricos:**
  - ❖ Observación: Se utiliza para identificar los principales procesos que se desarrollan en torno a las empresas mencionadas y pozos de petróleo.
  - ❖ Entrevistas: Se realiza entrevistas a los trabajadores de las empresas citadas con el fin de validar la propuesta que se presenta, y así recopilar toda la información necesaria para realizar la base de datos propuesta.

Este trabajo está estructurado en 3 capítulos, donde se describe la investigación realizada y los resultados obtenidos de la misma.

Capítulo 1: En el presente capítulo se brinda información sobre el estudio realizado de los diferentes entornos existentes en el negocio identificado, las principales soluciones para la obtención de información que existen en el mundo del petróleo, además se habla acerca del origen, desarrollo, tipos, modelos de las BD, las herramientas, gestores y técnicas empleadas para el diseño de las mismas, todo esto con el fin de alcanzar el objetivo trazado.

Capítulo 2: En este capítulo se presenta la integración de la solución con otros framework, la descripción de la arquitectura a desarrollar, los requisitos funcionales y no funcionales que debe satisfacer la BD, el diagrama de clases persistentes y entidad relación del SIPP, además de la descripción de las tablas obtenidas.

Capítulo 3: En este capítulo se ofrece información acerca de la validación tanto teórica como funcionalmente realizada al diseño de la base de datos propuesta, se muestran las reglas de integridad, la normalización a tener en cuenta para obtener un diseño con calidad; se describe además, como interactúa la BD con la capa de acceso a datos implementada por el framework Symfony, así como, las herramientas para el llenado voluminoso e inteligente de la BD, utilizadas en la validación de la misma.

CAPÍTULO I. FUNDAMENTACIÓN TEÓRICA

Introducción

En el presente capítulo se brinda información acerca del estudio realizado de los diferentes entornos existentes en el negocio identificado, sobre el origen y desarrollo de las Base de datos (BD), las herramientas, gestores, técnicas empleadas para su diseño y modelación, además de valorar cuál es el modelo, diseño, arquitectura, herramientas y gestores más acorde a utilizar.

1.1 Descripción actual de los procesos del negocio.

A nivel internacional están muy desarrolladas las aplicaciones relacionadas con las distintas fases de proceso del crudo encaminadas a resolver disímiles problemáticas que existen dentro de esta esfera, principalmente en el proceso de gestión obtención de reportes, generados en operaciones realizadas en los pozos en perforación.

El punto de partida en la búsqueda del petróleo es la **exploración**, encargada de realizar los estudios preliminares para la localización de un yacimiento, donde se estudian los terrenos minuciosamente analizando cada característica del mismo (24). Como segunda etapa en el proceso de búsqueda se realiza la **perforación**, se comienza la perforación por un pozo "*el pionero*" mediante la utilización de una sonda dependiendo de la región y de la profundidad a la cual se encuentra la estructura geológica o formación seleccionada con posibilidades de contener petróleo. Una vez comprobada la existencia del petróleo, otros pozos se perforarán para evaluarse la extensión del yacimiento. Esta información es la que determina si es comercialmente viable o no extraer el petróleo descubierto (24). En nuestro país fundamentalmente en la parte occidental y central existen varias empresas dedicadas a esta rama de la economía, de las cuales se describen algunas a continuación.

El entorno del negocio estudiado se encuentra dividido en tres áreas fundamentales:

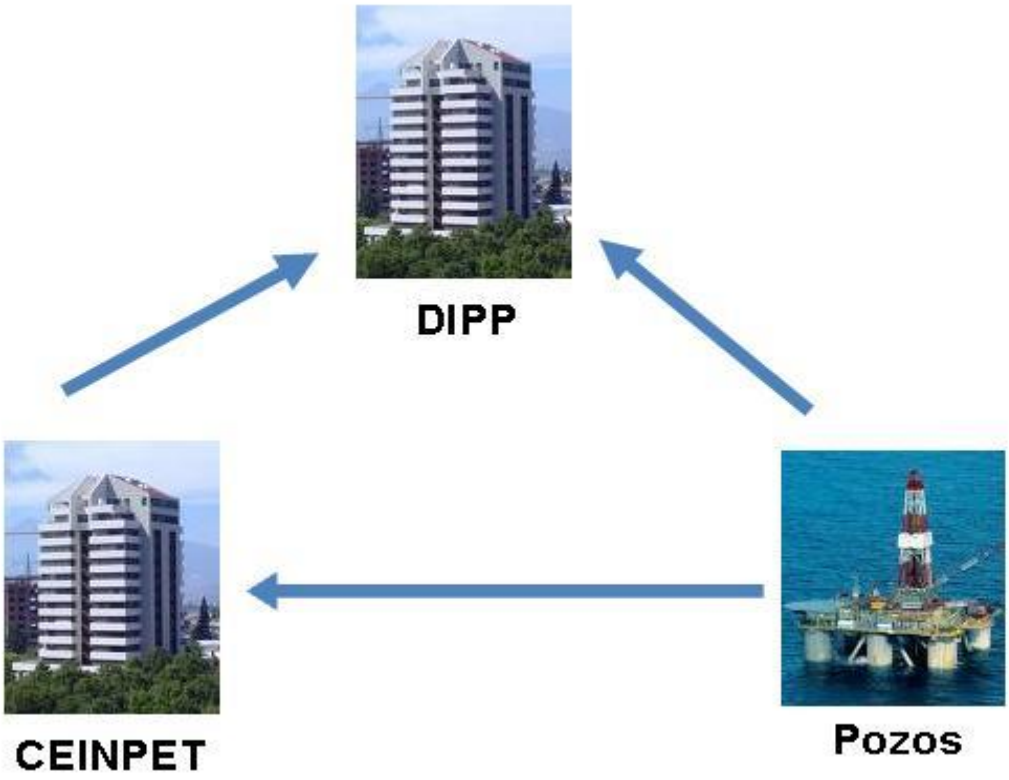

Figura 2. Distintas áreas del negocio

DIPP: Empresa que se encuentra ubicada en el municipio Varadero perteneciente a la provincia de Matanzas, aquí se confeccionan una serie de informes, partes y reportes manualmente utilizando el Excel con las informaciones que se reciben diariamente a las ocho de la mañana por correo electrónico de los pozos petroleros en perforación y de CEINPET, en caso de no existir la conexión se emiten por vía telefónica. Estos documentos generados contienen información de la perforación diaria, donde se incluyen informaciones de la cantidad de combustible, barrenas, herramientas, camisas y productos químicos utilizados en los pozos, así como las operaciones y actividades que se realizarán en el día, además de la cantidad de dinero invertido. Los mismos son manipulados por las secretarias de la DIPP, aprobada por Jefe de Despacho, y guardada en formato duro y digital para tomar decisiones en momentos posteriores.

CEINPET: Perteneciente a la empresa de CUPET, está localizada en Boyeros, provincia Ciudad de La Habana. Aquí diariamente la información es recibida en forma de reporte a las 7 de la mañana de igual forma al procedimiento descrito anteriormente por la DIPP, una vez recepcionada, es gestionada manualmente utilizando el Excel para elaborar el parte diario de geología y la confección de las gráficas correspondiente a la columna litológica, la cual contiene todas las características propias de las diferentes formaciones, es decir, tipos de rocas existentes en los distintos intervalos en los cuales se han perforado, edad de las mismas, para luego ser guardados en formato duro y digital, llevando de esta forma un historial de toda la documentación realizada.

Pozos Petroleros en Perforación: Están ubicados por toda la parte norte de nuestro país principalmente en la provincia de la Habana y Matanzas. En los mismos laboran una serie de personas que contribuyen a la gestión de la información generada en cada uno de los pozos las cuales se describen a continuación.

Direccional*:* Maneja toda la información concerniente a la dirección de las barrenas durante la perforación del pozo, este elabora diariamente el Reporte del Direccional con información obtenida del Well-Wizard (software propietario existente en los pozos de petróleos cubanos que gestiona todos los procesos que ocurren dentro de los mismos).

Químico: Gestiona toda la información relacionada con los productos químicos, así como, los aditivos añadidos al lodo de perforación, este es contratado por empresas extranjeras, diariamente confecciona un reporte con los productos utilizados, la cantidad y el precio de cada uno, lo guarda en formato duro y digital y lo entrega al Supervisor de Pozo.

Geólogo: Realiza diariamente un estudio consecuente de las características de las rocas, del terreno, analizando todo tipo de formación, obtiene información del Well-Wizard, elabora el Reporte Diario de Geología, lo guarda en formato duro y digital y lo entrega al Supervisor de Pozo y además de enviarlo al geólogo que labora en la oficina del CEINPET.

Supervisor de Pozo: Recibe diariamente información en formato duro y digital del químico, geólogo, direccional y analiza minuciosamente el software Well-Wizard, examina los parámetros recibidos, confeccionando con estos datos una serie de reportes, los cuales guarda en formato duro y digital y envía diariamente a las doce de la noche a la DIPP y a CEINPET.

Estos procesos descritos ocurren en cada uno de los pozos existentes en Cuba y en el mundo. Para resolver esta situación en países productores de petróleo utilizan algunos softwares, de los cuales, se describen a continuación sus principales funciones.

1.1.1 Soluciones actuales que generan información referente a los pozos.

Seguidamente se muestran algunos de los softwares que se utilizan a nivel mundial, que se encargan de obtener y manejar la información generada en los pozos.

WELL-LOGGER.

Well-Logger permite crear informes de perforación de suelos y diagramas de construcción de pozos. Los ingenieros de proyecto y los geólogos, pueden al usar este software en un ordenador portátil, completar rápidamente la documentación necesaria, este proceso ocurre normalmente durante las paradas de los trabajos de perforación. Ofrece una sencilla aunque robusta interfaz de usuario que brinda presentaciones personalizables, patrones definidos por el usuario, escala ajustable y vista previa de la impresión. Tiene una interface de hoja de cálculo fácil de usar, con cajas de entrada que simplifican la entrada de datos para cada perforación. La información de entrada incluye litología de la perforación, muestras tomadas, construcción del pozo o detalles anexos de la perforación, y la información general acerca del proyecto y la perforación. Well-Logger puede crear sus informes de perforación y pozo en la mitad del tiempo que los programas tradicionales CAD. (13)

WELLSIGHT.

El software WellSight[1] está compuesto por una serie de módulos utilizados para captación, informes, monitoreo, seguimiento, reconciliación y exportación de datos .Originalmente este software era para el desarrollo petrolífero en la Cuenca occidental de Canadá pero luego de ver sus beneficios se ha extendido su uso a diferentes usuarios del mundo. Dentro de sus diferentes funciones se hará alusión a la de exportación de datos, pues es la función que se encuentra muy relacionada con el negocio a desarrollar.

. (14)

[1] WellSight Systems Inc: Empresa de software Canadiense.

La Exportación de datos del software WELLSIGHT brinda cuatro opciones para extraer datos de la base de datos de WELLSIGHT. Estas opciones representan normas de la industria o formatos específicos de clientes e incluyen lo siguiente:

- Lenguaje para marcar normas para transferir información del sitio del pozo WITSML, por sus siglas en inglés.
- Gestión de datos de Landmark
- Gestión de datos DISWIN de Chevron
- Gestión de datos DBR de Statoil

Características

El WITSML es una iniciativa de la industria petrolera que establece nuevas normas para la transferencia de información sobre perforación lo cual ha sido de gran ayuda en el proceso de control y supervisión de los pozos. Archivo de exportación WITSML Export, versión 1.3.1 (*.XML).Una amplia gama de objetos con datos pueden transferirse a través de WITSML: (14)

- Pozos.
- Datos del pozo.
- Informes de fluidos.
- Inventario de fluidos.
- Volúmenes en tanques.
- Volúmenes de lodo.
- Bombas de lodo.
- Zarandas y mallas de zaranda.
- Descripción de tratamientos.

La Gestión de datos DIMS de Landmark y la exportación de aplicaciones Open Wells® incluye (14):

- Propiedades de fluidos de perforación
- Inventario de lodos

La exportación de la aplicación de Gestión de datos DISWIN de Chevron incluye:

- Propiedades de fluidos.
- Inventario.
- Volúmenes.

- Tratamientos.
- Costo.

Ventajas

- Minimización de tiempo de transferencia de datos con aplicaciones del cliente
- Mejor calidad de datos
- Cuatro opciones diferentes para exportación de datos

STRATER

Potente e innovadora herramienta que tiene como objetivo el registro de datos de los pozos y el trazado de perforaciones. Con esta aplicación los geólogos ya no tendrán que invertir más tiempo y dinero para crear registros profesionales de pozos. (15)

Con Strater se puede visualizar gráficamente:

- Profundidad o elevación
- Notas, comentarios y otros datos de texto
- Petrología, por ciento de petrología y descripciones litológicas.
- Potencial espontáneo, rayos gamma, calibrador, neutrónica, DST(Dirección de Servicios Tecnológicos), densidad aparente, resistividad, ritmo de perforación, gas total, calidad de los gases y datos sónicos
- Contabilidad de impactos, número y tipo de muestras, permeabilidad, RQD, lecturas OVM, por ciento de recuperación
- Concentración de contaminantes, contenido de humedad y detalles de construcción de pozos.
- Datos de ensayos, petrología de mineralización o alteración, valores BTU (Unidades Térmicas Británicas) y datos de contenido de cenizas.
- Virtualmente cualquier tipo de dato de profundidad o intervalo.

“Strater ofrece una flexibilidad insuperable para el diseño y formateo de registros. Strater incluye 13 tipos de registro muy usuales para visualizar sus datos gráficamente: profundidad, línea/símbolo, gráfico cruzado, petrología, barras de zona, barras, porcentajes, postes, postes por clase, gráficos, textos complejos, y registros de construcción de pozos. Cada uno de los registros puede ser modificado para

ajustarse a las necesidades del usuario. Dentro de algunas de las personalizaciones posibles podemos citar:

- Visualizar registros basados en profundidad o elevación.
- Visualizar profundidad y/o líneas de rejilla variable.
- Añadir barras de escala y títulos.
- Configurar diferentes estilos de líneas de contacto entre unidades litológicas.
- Utilizar curvas para crear perfiles litológicos como perfiles de desgaste.
- Crear registros continuos sencillos o de múltiples páginas.
- Formatear diferentes tipos de registro para obtener presentaciones lo más informativas posible.

Strater proporciona varias funcionalidades para simplificar la tarea de importar datos, crear el diseño de la perforación exacta que el usuario requiere y obtener la salida en el formato necesario, ya sea impreso o exportado a formato electrónico para incluir en un informe o presentación. Su robusta base de datos que soporta importación de datos de muchas fuentes diferentes, incluyendo ficheros de texto ASCII, ficheros LAS, ficheros XLS y prácticamente cualquier fichero de base de datos. El usuario puede importar datos referentes a múltiples perforaciones en una tabla de datos e importar múltiples tablas de datos en un proyecto. También puede crear múltiples vistas de perforaciones en un proyecto. Cada vista de perforación puede contener un diseño de perforación diferente, permitiendo así una flexibilidad inigualable en la presentación de sus datos. El almacenamiento de todas las vistas y tablas de datos de las perforaciones en un único proyecto mantiene toda la información fácilmente accesible".

Su versatilidad y facilidad de uso lo hace un software ideal para numerosas industrias, entre las que se incluyen las petrolíferas, geofísicas, minerías, medioambientales, geotécnicas y muchas otras. (15)

1.1.2 Sistema a desarrollar.

Estos software analizados, ayudan a la gestión de la información generadas pero son propietarios, es decir que para obtenerlos hay que pagar una licencia valorada en miles de dólares, lo cual no resuelven la situación existente, debido a esto, nuestro país en aras de desarrollarse se ha propuesto llevar adelante una aplicación que facilite la gestión de información en esta rama, esto infiere la importancia que se le debe aportar a la hora de realizarlo, se espera con el mismo automatizar muchas de las operaciones de

gestión de información que se realizan en los pozos, por ejemplo la generación del Parte Diario de Geología y el Reporte Diario de Perforación, así como, en las empresas implicadas en el negocio del petróleo, y en algún momento extenderlo por el mundo. Para un buen desenvolvimiento de este sistema es necesario hacer un estudio acerca de las bases de datos, específicamente su diseño y su utilización.

1.2 Base de Datos

1.2.1 Surgimiento de las Bases de datos

Tuvo sus orígenes a principios de los años 60 debido a la necesidad de almacenar la información que se gestionaba en la memoria de la máquina. Surgen con el objetivo de guardar grandes cantidades de datos y poder gestionarlos cada vez con mayor facilidad.

Las primeras bases de datos (BD) se gestionaban en ficheros que eran almacenados en tarjetas o soportes magnéticos. Las BD según la definición emitida por el **Dr. Rogelio S Silverio Castro** "*son un conjunto de ficheros o tablas que residen de forma permanente en la memoria de la computadora*" (12).

En el año 1970 se convoca a una Conferencia de Lenguajes de Programación en Estados Unidos, convocada por IBM, donde se establece un modelo llamado CODASYL (Modelo para el tratamiento de bases de datos que fue publicado por E. Codd). Codd, propuso una forma de organizar las bases de datos mediante un modelo matemático lógico, a raíz de esto se estableció un modelo estándar para el desarrollo de las bases de datos y surge años más tarde el lenguaje **SQL**, siendo este el más usado actualmente por los desarrolladores de software y la sociedad (17).

Con la evolución de las computadoras, las BD fueron progresando paralelamente con el desarrollo de la tecnología, haciéndose cada vez más útiles al desarrollo de la sociedad.

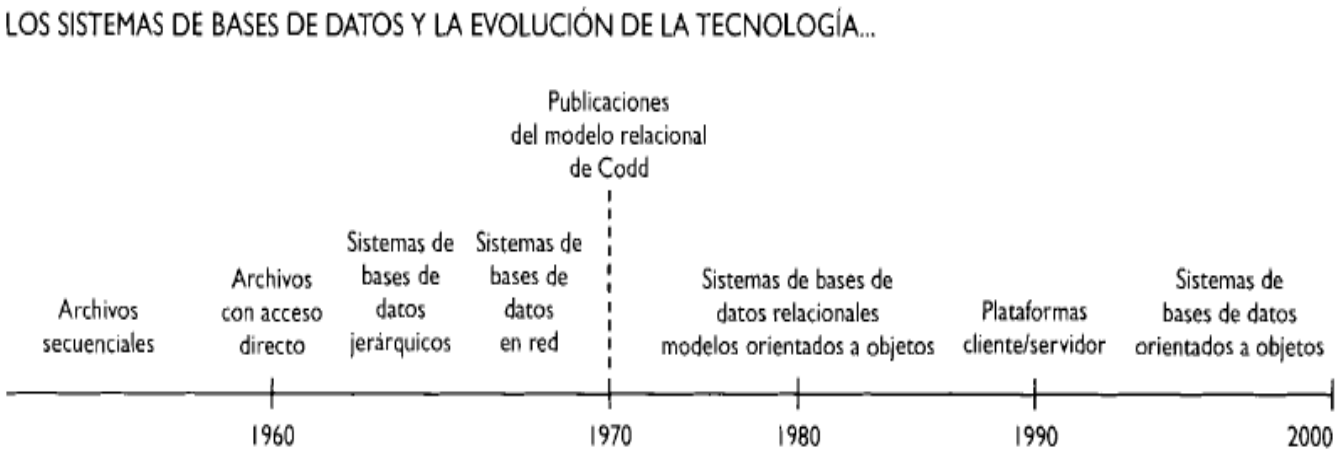

Figura 3. Evolución de los Sistemas de Bases de datos (SBD) [2]

Con la necesidad de lograr la gestión de las Bases de datos surgen los Sistemas Gestores de Bases de Datos (SGBD), que, según el Dr. **Rogelio S Silverio Castro** se definen en *"un tipo de software muy específico, dedicado a servir de interfaz entre las bases de datos y los sistemas que la utilizan. Sus funciones principales son la creación, mantenimiento y eliminación de bases de datos , el control de accesos, la manipulación de información de acuerdo a las necesidades del usuario, el cumplimiento de las normas de tratamiento de datos, evitar redundancias e inconsistencias y mantener la integridad".*(12)

Otra de las definiciones acerca de los Sistema Gestor de Bases de datos (SGBD) o DBMA (Database Management System), explica que los SGBD son una colección de programas cuyo objetivo es servir de interfaz entre la base de datos, el usuario y las aplicaciones. Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Los SGBD permiten definir los datos a distintos niveles de abstracción y manipular dichos datos, garantizando la seguridad e integridad de los mismos. (16)

Hoy en día existe gran cantidad de SGBD, aunque con diferencias de funciones, expresan un mismo objetivo, en el **epígrafe 1.3.1** se muestran algunos de los más usados a nivel mundial, de los cuales se

[2] Tomado de Diseño y Administración de Bases Datos. Hansen, Gary W. y Hansen, James V.

realizará un estudio y comparación de sus características funcionales, concluyendo con la selección del más conveniente según las necesidades existentes.

Luego de haber analizado los conceptos de bases de datos y de los sistemas gestores de bases de datos, se desarrollará un estudio acerca de las clasificaciones, arquitectura y diseño las BD etc.

1.2.2 Clasificaciones de las Bases de Datos.

A continuación se ofrece algunas de las clasificaciones de las bases de datos.

Según la movilidad de sus datos:

- *"Base de Datos Estáticas*: los datos no varían, solamente son consultados. Son utilizadas principalmente para almacenar datos históricos que en un momento dado servirán para la toma de decisiones.
- *Base de Datos Dinámicas*: los datos cambian con el tiempo según las operaciones que se realizan (Insertar, Modificar y Eliminar). " (18)

Según su contenido:

- *Base de Datos Bibliográficas:* Sólo contienen un representante de la fuente primaria, que permite localizarla. Un registro típico de una base de datos bibliográfica contiene información sobre el autor, fecha de publicación, editorial, título, edición, de una determinada publicación, etc. Puede contener un resumen o extracto de la publicación original, pero nunca el texto completo, porque si no estaríamos en presencia de una base de datos a texto completo (o de fuentes primaria). Como su nombre lo indica, el contenido son cifras o números. Por ejemplo, una colección de resultados de análisis de laboratorio, entre otras.

- *Bases de datos de texto completo:* Almacenan las fuentes primarias, como por ejemplo, todo el contenido de todas las ediciones de una colección de revistas científicas". (18)

- *Directorios***:** Proporcionan información sobre individuos, organizaciones y servicios, ejemplos nombres, direcciones entre otras cosas. Un ejemplo, son las guías telefónicas en formato electrónico. (19)

1.2.3 Diseño de las Bases de Datos.

Un diseño correcto de una base de datos permite tener acceso a información exacta y actualizada y a la vez es esencial para lograr los objetivos fijados.

“Parece lógico emplear el tiempo que sea necesario en aprender los principios de un buen diseño, ya que en ese caso, es mucho más probable que la base de datos termine adaptándose a sus necesidades y pueda modificarse fácilmente.

A la hora de realizar el diseño de una base de datos se plantean varios principios:

1. Los datos no deben de ser redundantes: malgastan el espacio en la BD y aumentan la probabilidad de que se produzcan errores e incoherencias.
2. La información debe de estar correcta y completa.

El proceso de diseño consta de los pasos siguientes:

- **Determinar la finalidad de la base de datos:** Permite estar preparado para los demás pasos.
- **Buscar y organizar la información necesaria:** Reúne todos los tipos de información que desee registrar en la base de datos, como los nombres de productos o los números de pedidos.
- **Dividir la información en tablas:** Divide los elementos de información en entidades o temas principales. Cada tema pasará a ser una tabla.
- **Convertir los elementos de información en columnas:** Decide qué información desea almacenar en cada tabla. Cada elemento se debe de convertir en un campo y se mostrará como una columna en la tabla. Por ejemplo, una tabla Empleados podría incluir campos como Apellido y Fecha de contratación.
- **Especificar claves principales:** Se debe de especificar la clave principal de cada tabla. La llave principal es una columna que se utiliza para identificar inequívocamente cada fila, como id. del producto o id. del pedido.
- **Definir relaciones entre las tablas:** Permite examinar cada tabla y posibilita decidir cómo se relacionarán los datos de una tabla con las demás tablas. Con esto se añaden nuevos campos a las tablas o crea nuevas tablas para refinar las relaciones según sea necesario.

- **Ajustar el diseño:** Se debe de analizar el diseño para detectar errores. Cree las tablas y agregue algunos registros con datos de ejemplo. Compruebe si puede obtener los resultados previstos de las tablas. Realice los ajustes necesarios en el diseño.
- **Aplicar las reglas de normalización:** Aplica reglas de normalización de los datos para comprobar si las tablas están estructuradas correctamente. Realiza los ajustes necesarios en las tablas." (20)

Objetivos del diseño de Bases de datos.

- Un rápido y eficiente acceso a la información con la menor redundancia.
- Diseño de esquemas en la forma normal (FN).
- Información adicional.
- Especificación de limitantes.

El diseño de una base de datos está compuesto por tres partes:

1. **Diseño Conceptual:** Permite descubrir el significado de los datos con los que se irán a trabajar, tiene como objetivos comprender:
   - ☞ La perspectiva que tiene cada usuario acerca de los datos.
   - ☞ La naturaleza de los datos, independientemente de su representación física.
   - ☞ El uso de los datos a través de las áreas de aplicación.

   "Le permite al diseñador de bases de datos transmitir a la empresa los conocimientos adquiridos sobre la información que ella maneja.

   Este esquema se construye utilizando la información que se encuentra en la especificación de requisitos de usuarios. Es independiente, completamente de los aspectos de la implementación. El esquema conceptual es una fuente de información para el diseño lógico de la base de datos.
2. **Diseño Lógico:** Permite al diseñador construir un esquema a partir de la información de la empresa, basándose en un modelo de base de datos específicos. Es capaz de llevar del modelo conceptual al modelo lógico. Es fuente de información del modelo físico y juega un papel fundamental en el diseño y desarrollo de la BD. Permite que los posibles cambios que se realicen sobre los programas de aplicación o sobre los datos se representen correctamente en la base de datos". (21)

En esta parte del diseño se aplica la técnica de la normalización de BD para comprobar la validez de los esquemas lógicos del modelo relacional, ya que asegura que los datos en las tablas no contengan datos redundantes.

Tanto el esquema conceptual como el lógico son procesos iterativos, tienen un punto de inicio y se van refinando continuamente. Son dos etapas claves para conseguir que un sistema funcione correctamente.

3. **Diseño Físico: "**El diseño físico es el proceso de producir la descripción de la implementación de la base de datos en memoria secundaria: estructuras de almacenamiento y métodos que garanticen un acceso eficiente a los datos.

   Para llevar a cabo esta etapa, se debe haber decidido cuál es el SGBD que se va a utilizar, debido a que, este esquema se adapta a él. Entre el diseño lógico y el físico hay una realimentación, puesto que, algunas de las decisiones que se tomen durante el diseño lógico para mejorar las prestaciones, pueden afectar a la estructura del esquema físico." (21)
   En general, el propósito del diseño físico es describir cómo se va a implementar físicamente el esquema lógico obtenido en la fase anterior. Concretamente, en el modelo relacional, esto consiste en (24):

   - Obtener un conjunto de relaciones entre tablas y las restricciones que se deben cumplir sobre ellas.
   - Determinar las estructuras de almacenamiento y los métodos de acceso que se van a utilizar para conseguir unas prestaciones óptimas.
   - Diseñar el modelo de seguridad del sistema.

1.2.4 Arquitectura de las Bases de datos

Para que un sistema sea útil, debe recuperar los datos eficientemente. Como muchos usuarios de sistemas de bases de datos no están familiarizados con computadoras, los desarrolladores esconden la complejidad a los usuarios a través de varios niveles de abstracción para simplificar la interacción de los usuarios con el sistema: (23)

- **Nivel Interno:** Es el nivel más cercano al almacenamiento físico de los datos. Permite escribirlos tal y como están almacenados en el ordenador. En este nivel se diseñan los archivos que contienen la información, la ubicación de los mismos y su organización, es decir se crean los archivos de configuración.(25)

- **Nivel Conceptual:** "En este nivel se representan los datos que se van a utilizar sin tener en cuenta aspectos como lo que representamos en el nivel interno.

- **Nivel Externo:** Es el más cercano al usuario. En este nivel se describen los datos o parte de los datos que más interesan a los usuarios". (23)

La arquitectura de las bases de datos tiene como objetivo separar las aplicaciones de los usuarios de la base de datos física, además de brindarles una visión abstracta de los datos.

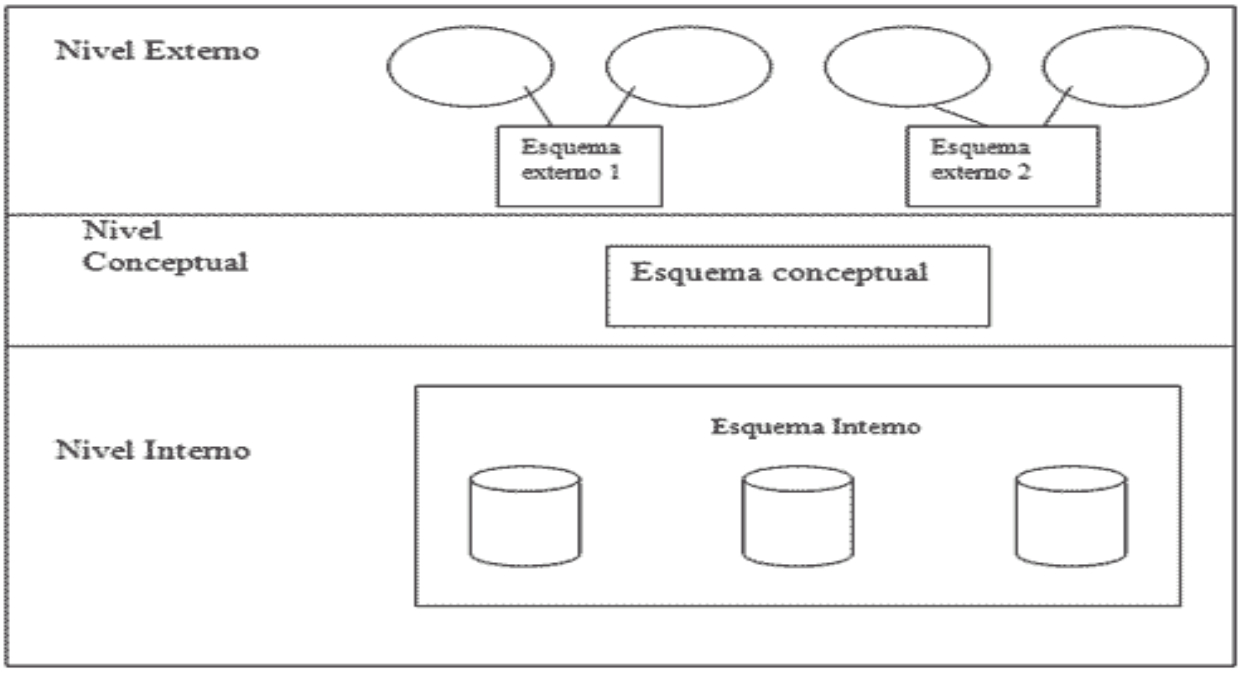

Figura 4. Distintos Niveles de la Arquitectura de Base de Datos. [3]

[3] Tomado de Desarrollo Web Disponible en: http://www.desarrolloweb.com/arquitecturadelasbasesdedatos

1.2.4.1 Diseño de la Arquitectura de Bases de Datos.

Seguidamente se muestra las clasificaciones de la arquitectura de las bases de datos, las cuales nos permite analizar cómo quedarán ubicados y organizados los datos.

Arquitectura Centralizada

Es una base de datos que está físicamente ubicada en un lugar en específico, la cual es controlada por un mismo computador.

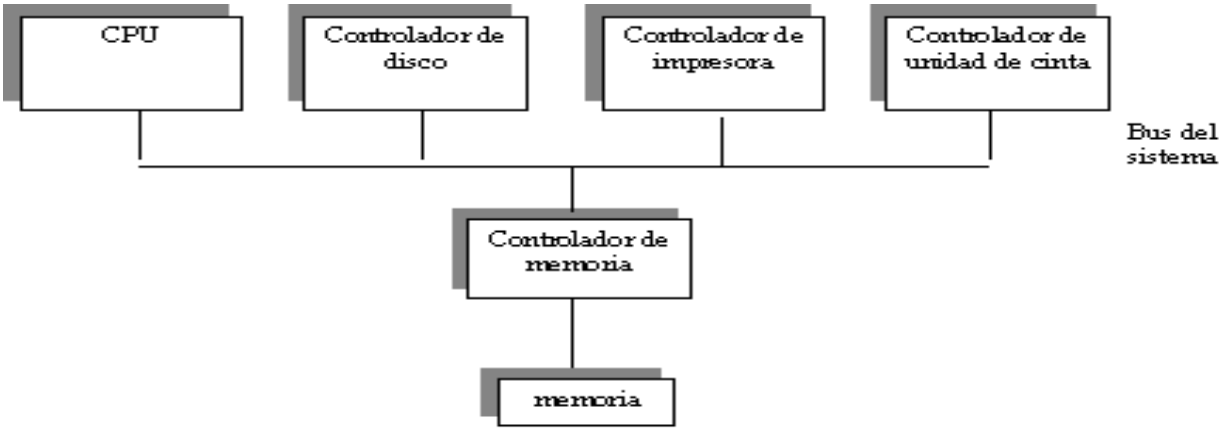

Figura 5. Arquitectura Centralizada. [4]

Características

- Se almacenan los datos en un solo servidor.
- El problema de seguridad es relativamente fácil en estos sistemas de bases de datos.
- Está compuesta por datos, software de gestión (SGBD) y dispositivos de almacenamiento secundario.

Ventajas:

1. Fácil de manipular los datos.
2. Brinda buena seguridad de la información almacenada.
3. Su instalación no es costosa.
4. Evita la redundancia de los datos.

[4] Tomado de Desarrollo Web Disponible en: http://www.desarrolloweb.com/arquitecturadelasbasesdedatos

5. Evita la inconsistencia de la información.
6. Garantiza buena integridad en los datos.

Desventajas:

1. En caso de un fallo de conexión el sistema entero se cae.
2. En caso de catástrofes o desastres, la recuperación de los datos es difícil de sincronizar.
3. Menor proporción precio/rendimiento que los microprocesadores en los sistemas distribuidos.
4. Menor cómputo en las consultas con respecto a los sistemas distribuidos.

Arquitectura Descentralizada

Es una base de datos que está ubicada en varias computadoras la cuales pueden ser administradas tanto global como localmente. Esta es desarrollada con vista a resolver los problemas existentes en las centralizadas.

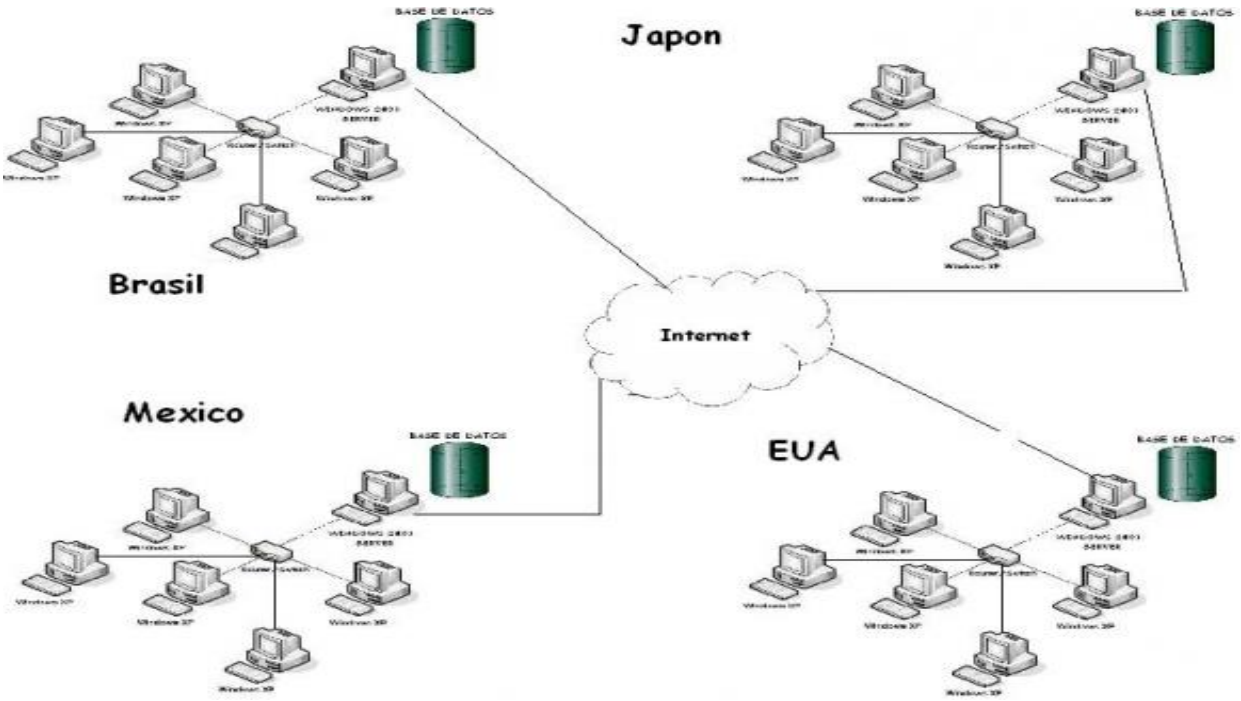

Figura 6. Arquitectura Descentralizada [5]

[5] Tomado de http://www.danervo.com

Ventajas:

1. Mejora el rendimiento.
2. Mayor fiabilidad y control de los datos.
3. Mayor compartición y disponibilidad de los datos.

Desventajas.

1. Redundancia de la información
2. Costo en el desarrollo del software.
3. Mayor posibilidad de errores.

1.2.5 Réplicas

La replicación es el proceso de copiar y de mantener los objetos o tablas de una base de datos en múltiples BD ubicados en diferentes sitios.

Ventajas de las Réplicas:

- El sistema en caso de caída de algunos de los nodos se mantiene funcionando normalmente.
- Todos los nodos pueden realizar consultas a la vez o en paralelo a una misma tabla. Posibilita un mayor tráfico en la red mientras más nodos existan.

1.2.5.1 Entornos de las Réplicas.

Existen varios tipos de entornos de las réplicas.

- **Maestro-Esclavo (Master-Slave)** o de solo lectura, permite a un solo maestro realizar consultas de lectura-escritura, mientras que los esclavos pueden aceptar consultas de lectura.

- **Multi-Maestro (Multi-Slave)** llamada también par-par. Cada sitio es un entorno de la réplica multi-maestro es un maestro y cada uno de ellos se comunican con otros sitios maestros. (31)

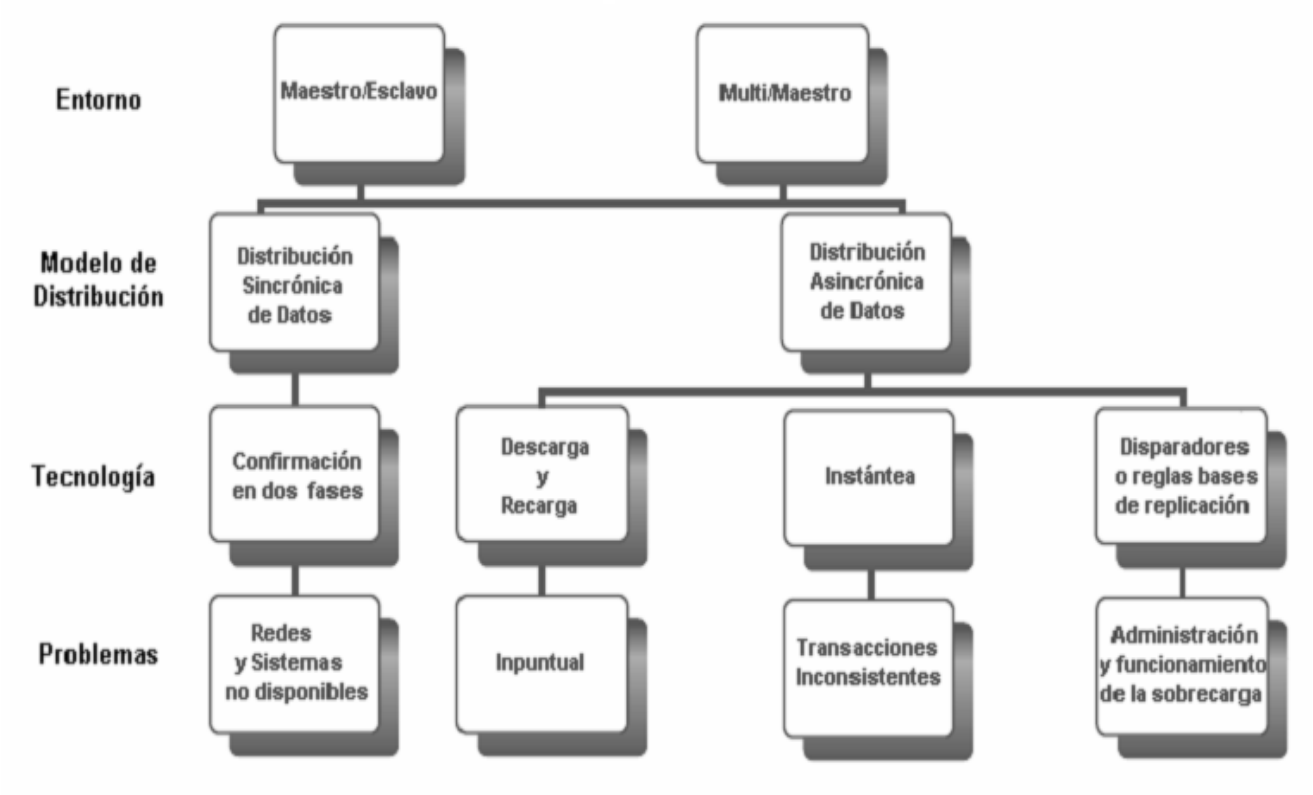

Figura 7 Esquema de Réplicas de Datos

1.2.5.2 Modelos de Distribución de Datos.

- **Sincrónica:** Ejecuta cualquier cambio o algún procedimiento hecho en todos los sitios que participan en la réplica, como parte de una sola transacción. Si en un momento algún procedimiento falla, la transacción entera se cae también. Este tipo de réplica garantiza la consistencia de los datos en tiempo real en todos los sitios. (31)

- **Asincrónica:** A menudo alguna funcionalidad de almacenamiento envía, captura cualquier cambio local, lo almacena en una cola, y en intervalos regulares propaga y aplica cambios en sitios remotos. En este modelo, hay un lapso de tiempo para que los datos alcancen la analogía de los datos. (31)

1.2.5.3 Técnicas de Replicación.

- **Sincrónica** *Confirmación en dos fases (Two Phase Commit)*

  "Surge a mediados de los 80. Permite la sincronización de los datos distribuidos. Cada transacción es realizada solamente si todos los nodos replicados están conectados entre sí y listos para recibir la información, en caso contrario la misma es anulada es decir no se realiza.

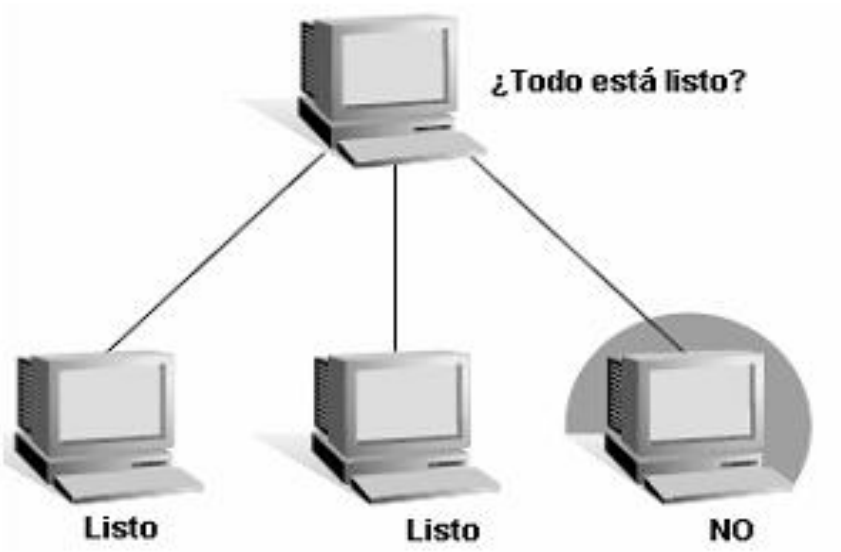

Figura 8. Confirmación en dos fases

- **Asincrónica**

  Descarga y Recarga (Dump and Reload).

  Consiste en hacer un volcado de los datos, copiar la salva para un dispositivo de almacenamiento, para luego distribuirlo hacia los demás servidores. Tiene como inconveniente que en muchos casos pueden existir la desactualización de la información.

  Instantánea (Snapshots)

  Consiste en realizar una instantánea de una tabla dada en un momento determinado y distribuir la misma por todas las réplicas. A pesar de que es mucho más avanzada que la técnica de Descarga y Recarga tiene como inconveniente de que las tablas esclavas son de solo lectura.

Disparadores (Triggers)

Tiene como función, realizar una acción cuando ocurre un suceso en la base de datos, los eventos son de gestión (inserción, actualización y eliminación). Son al igual que las instantáneas, mecanismos asíncronos que proporcionan a la base de datos una forma de replicación de los datos." (31)

1.2.5.4 Conflictos de Replicación.

"Existen varios tipos de conflictos a tener en cuenta, al tratar el tema de la replicación de datos. Los conflictos pueden ocurrir cuando estamos trabajando en un ambiente de replicación que permite actualizaciones concurrentes sobre los mismos datos en múltiples sitios.

1. **Conflicto de Actualización**

Este conflicto ocurre cuando se produce la replicación de una actualización (update) sobre un registro con otra actualización (update) sobre el mismo registro. Este conflicto ocurre cuando dos transacciones originadas desde distintos sitios actualizan el mismo registro, en forma cercana en el tiempo.

Resolución del problema.

- **Prioridad**: Cada servidor obtiene una prioridad única, y el servidor de mayor prioridad "gana", respecto de aquellos con prioridad menor.
- **TimeStamp**: La más nueva o la más antigua de las modificaciones es la considerada correcta, y por defecto, sino se eligió ninguno de los criterios "gana" la más nueva.
- **Particionamiento de datos:** Se garantiza que cada registro sea manipulado por un único servidor, lo que simplifica la arquitectura.

2. **Conflicto de unicidad.**

El conflicto de unicidad sucede cuando la replicación de un registro intenta violar una restricción de integridad, ya sea por llave primaria o única (Primary Key o Unique), por ejemplo: considere lo que sucede, cuando dos transacciones originadas de dos sitios diferentes, cada una inserta un registro, en su respectiva tabla replicada, con el mismo valor de clave primaria. En ese caso sucede un conflicto de unicidad.

Resolución

Para cada servidor brindar un rango distinto de números para los generadores de clave (secuencias). Agregar el identificador del servidor a la clave primaria.

Replicar en tablas separadas, y acceder a los datos a través de una vista formada por la unión de ellas. Para resolver el conflicto de claves duplicadas en la unión se usará una pseudo columna que representa la base de datos fuente.

3. Conflicto de supresión

Un conflicto de supresión ocurre cuando dos transacciones originadas de sitios diferentes, una de ellas intenta borrar un registro, y la otra actualizar o borrar el mismo registro, ya que en este caso el registro no existe, tanto para ser actualizado como borrado.

Resolución

Para evitar este tipo de conflictos, una posible solución es que los sitios marquen lógicamente los registros a ser borrados y que periódicamente el sitio maestro corra un proceso que realice el borrado (delete) físico de los datos, es decir desde los sitios replicados no se puede ejecutar una sentencia para hacer el borrado de los datos (delete).

4. Conflicto de orden

Los conflictos de orden pueden ocurrir en ambientes de replicación con tres o más sitios maestros. Si la propagación al sitio maestro X, está bloqueada por alguna razón, entonces la replicación de modificaciones en datos puede seguir siendo propagada a través de los otros sitios maestros; al finalizar la propagación estas modificaciones debieron ser propagadas al sitio X en un orden diferente a como ocurrieron en los otros sitios maestros, pudiendo producirse un conflicto.

Resolución

Éste tipo de conflicto suele resolverse asignándole distintas prioridades a los sitios maestros, de forma de que se organicen las transacciones de acuerdo a ésta". (31)

1.2.6 Arquitectura propuesta a desarrollar.

Se propone una arquitectura descentralizada con la implementación de réplicas, consiste en una base de datos central ubicada en la DIPP y distintas réplicas de la misma ubicadas en los distintos pozos de petróleo en perforación existentes en el país. De los pozos se replicaría toda la información de las tablas que se modifican hacia la base de datos de la DIPP, actualizando de esta forma los datos que se encuentran en ella. Se utiliza esta arquitectura debido a los problemas de conexión que existe entre los pozos y las empresas implicadas. Esta arquitectura permite no cargar el o los servidores de BD, presenta buena fiabilidad y disponibilidad de los datos, en caso de un error en la conectividad la aplicación seguiría funcionando en cualquier lugar y en más tarde se actualizaría los datos de la base de datos central.

Luego de haber analizado las diferentes arquitecturas existentes y explicar la propuesta a desarrollar, se realiza un estudio de los distintos tipos de modelos de bases de datos que existen.

1.2.7 Modelos de Bases de Datos.

Mediante el desarrollo de las tecnologías de la informática y las comunicaciones, las bases de datos han mejorado su estructura y sus modelos, diferenciándose por funcionalidades y características mejoradas o adicionadas según el propósito que se desee lograr por el diseñador, entre los más usados se encuentran:

- **Modelo Jerárquico**: Este modelo utiliza árboles para la representación lógica de los datos. Este árbol está compuesto de unos elementos llamados nodos. El nivel más alto del árbol se denomina raíz. Cada nodo representa un registro con sus correspondientes campos. (24)

  La representación gráfica de este modelo se realiza mediante la creación de un árbol invertido, los diferentes niveles quedan unidos mediante relaciones. (24)

  - En este modelo sólo se pueden representar relaciones 1:M, por lo que presenta varios inconvenientes:
  - No se admiten relaciones N:M
  - Un segmento hijo no puede tener más de un padre.
  - No se permiten más de una relación entre dos segmentos.
  - Para acceder a cualquier segmento es necesario comenzar por el segmento raíz

- El árbol se debe de recorrer en el orden designado.

☞ **Modelo en Red**: En este modelo las entidades se representan como nodos y sus relaciones son las líneas que los unen. En esta estructura cualquier componente puede relacionarse con cualquier otro. (24)

Los conceptos básicos en el modelo en red son (24):

- Un hijo puede tener varios padres.
- El tipo de registro, que representa un nodo.
- Elemento, que es un campo de datos.
- Agregado de datos, que define un conjunto de datos con nombre.

☞ **Modelo Relacional**: Este modelo es el más utilizado actualmente para modelar los problemas reales y administrar los datos dinámicamente. Su idea fundamental es el uso de "relaciones". Estas relaciones podrían considerarse en forma lógica como conjuntos de datos llamados "tuplas". Esto es pensando en cada relación como si fuese una tabla que está compuesta por registros (las filas de una tabla) que representarían las tuplas y los campos (las columnas de una tabla).

En este modelo, el lugar y la forma en que se representen los datos no tienen relevancia (a diferencia de otros modelos como el de red y el jerárquico). (24)

Algunas de sus principales características son:

- Puede ser entendido y usado por cualquier usuario.
- Permite ampliar el esquema conceptual sin modificar las aplicaciones de gestión.
- Los usuarios no necesitan saber donde se encuentran los datos físicamente.
- El elemento principal de este modelo son las relaciones que se realizan entre las tablas de la base de datos.(25)

☞ **Modelo Orientado a Objetos:** Trata de almacenar en la base de datos los *objetos* completos (estado y comportamiento).

Una base de datos orientada a objetos es una base de datos que incorpora todos los conceptos importantes del paradigma de objetos:

- *Encapsulación:* Cada objeto es consciente de sus propias características. Es decir que permite ocultar la información de las tablas al resto de los objetos.
- *Herencia:* Para facilitar la programación, se puede establecer toda una jerarquía de tipos o clases. Se evita la declaración de datos de forma redundante.
- *Polimorfismo:* Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos.

En bases de datos orientadas a objetos, los usuarios pueden definir operaciones sobre los datos como parte de la definición de la base de datos. Las operaciones (llamada función) se especifican en dos partes; la interfaz de una operación incluye el nombre de la operación y los tipos de datos de sus argumentos (o parámetros). La implementación (o método) de la operación se especifica separadamente y puede modificarse sin afectar la interfaz. Los programas de aplicación de los usuarios pueden operar sobre los datos invocando a dichas operaciones a través de sus nombres y argumentos, sea cual sea la forma en la que se han implementado. Esto podría denominarse independencia entre programas y operaciones. (24)

1.2.7.1 Selección del Modelo de Bases de Datos a utilizar.

Según el estudio realizado acerca de los modelos de BD existentes se decidió escoger como modelo, el relacional debido a:

- Modelo más utilizado en la actualidad para modelar los problemas reales, diseñar y desarrollar las soluciones propuestas a los mismos, del cual se han obtenidos excelentes resultados.
- Permite administrar datos dinámicamente.
- Aplica en su diseño un proceso de normalización de los datos que evita la redundancia y mantiene la integridad de la información.
- De fácil entendimiento hacia los usuarios, cualquier persona observando su diseño puede comprenderlo, aprenderlo y aplicarlo.

A pesar de que el modelo Orientado a Objetos (OO) es un nuevo modelo que presenta una serie de características nuevas tales como; herencia, encapsulación, polimorfismo, además que su modelación representa de manera más completa el mundo real, pues sus datos son objetos con características y

funciones; a la hora de realizar un modelo puede que surjan nuevas dudas y esto trae consigo nuevos problemas tales como atraso en el modelamiento y diseño de la BD, conjuntamente aprenderlo y desarrollarlo requeriría tiempo y dedicación. De acuerdo a las necesidades existentes en el negocio planteado se hace necesario utilizar un modelo eficiente, de fácil comprensión tanto del diseñador, como de los usuarios, por lo tanto este modelo se excluye de los analizados, además el modelo relacional resuelve perfectamente los problemas del negocio antes planteados.

1.3 Tendencias y Tecnologías Actuales

Para lograr un buen diseño y modelado de una base de datos se necesita de una serie de herramientas que permitan la gestión de los procesos inherentes, control sobre la redundancia de los datos, así como su consistencia, comparación de datos por usuarios autorizados, la integridad de los datos, la seguridad con protección frente a usuarios no autorizados, la accesibilidad a los datos a través de SGBD, mantenimiento e independencia de los datos, concurrencia, servicios de copias de seguridad y de recuperación ante fallos, así como, lograr que todo el proceso de su creación y modelado sea lo más sencillo y rápido posible. A continuación se muestra algunos de los SGBD y de las herramientas de administración y modelado existentes en el mundo, los cuales ayudan con el diseño y modelado de datos.

1.3.1 Sistemas Gestores de Bases de Datos.

MySQL

“Es uno de los gestores de bases de datos, más usado en el mundo del software libre, debido a su gran rapidez y facilidad de uso. Esta gran aceptación es debida, en parte, a que existen infinidad de librerías y otras herramientas que permiten su uso a través de gran cantidad de lenguajes de programación, además de su fácil instalación y configuración.

Ventajas:

- Aprovecha la potencia de sistemas multiprocesador, gracias a su implementación multi-hilo.
- Soporta gran cantidad de tipos de datos para las columnas.
- Dispone de API's en gran cantidad de lenguajes (C, C++, Java, PHP, entre otros).
- Gran portabilidad entre sistemas.
- Soporta hasta 32 índices por tabla.

- Gestión de usuarios y contraseñas, manteniendo un muy buen nivel de seguridad en los datos.

Desventajas:

MySQL se incluye en el grupo de sistemas de bases de datos relacionales, pero carece de algunas de sus principales características:

- Select into table: esta característica propia de Oracle, todavía no está implementada.
- Transacciones: a partir de las últimas versiones (5.0.67 y 6.09) ya hay soporte para transacciones, aunque no por defecto (se ha de activar un modo especial).
- Integridad referencial: admite la declaración de claves ajenas en la creación de tablas, aunque internamente no las trate de forma diferente al resto de campos" (29).

PostgreSQL

Es un sistema gestor de bases de datos relacionales orientadas a objetos con cerca de una década de desarrollo. Soporta casi toda la sintáxis SQL (incluyendo sub-consultas, transacciones, y tipos y funciones definidas por el usuario), contando también con un amplio conjunto de enlaces con lenguajes de programación (incluyendo C, C++, Java, Perl, Python, entre otros). (11)

Características:

- Control de Concurrencia Multi-Versión (MVCC, Acceso Concurrente Multi-versión, siglas en inglés). MVCC, es la tecnología que PostgreSQL usa para evitar bloqueos innecesarios. PostgreSQL mantiene una ruta a todas las transacciones realizadas por los usuarios de la base de datos. Además es capaz entonces de manejar los registros sin necesidad de que los usuarios tengan que esperar a que los mismos estén disponibles.(11)

- Integridad referencial.

  PostgreSQL soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.

- Conectividad **TCP/IP** (protocolo de control de transmisión, **Transmission Control Protocol/Internet Protocol**, siglas en Inglés), **JDBC**(es una interfaz de programación de aplicaciones (API) que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java, **Java Database Connectivity**, siglas en Inglés) y **ODBC** (estándar de acceso a bases de datos, **Open Database Connectivity**, siglas en Inglés). (5)

Una lista breve de características técnicas que PostgreSQL ofrece (4):

- Cumple completamente con ACID (acrónimo de Atomicidad, Consistencia, Aislamiento y Durabilidad).
- Cumple con ANSI SQL.
- Replicación (soluciones comerciales y no comerciales) que permiten la duplicación de bases de datos maestras en múltiples sitios de réplica.
- Soporte para consultas con UNION, UNION ALL y EXCEPT.
- Extensiones para SHA1 (Algoritmo de Hash Seguro, **Secure Hash Algorithm**, siglas en inglés), MD5 (abreviatura de **Message-Digest Algorithm** *5*, Algoritmo de Resumen del Mensaje 5), XML (Lenguaje de marcas o etiquetas, abreviatura de E**x**tensible **M**arkup **L**anguage) y otras funcionalidades.
- Incorpora una estructura de datos array's, además incluye herencia entre tablas, por lo que se le incluye entre los gestores objeto-relacionales más importantes.
- Tiene la capacidad de almacenar procedimientos en la propia base de datos, equiparándolo con los gestores de bases de datos de alto nivel, como puede ser Oracle.
- Es el segundo motor de base de datos con más funcionalidades tras Oracle.
- Está considerado como la base de datos de código abierto más avanzada del mundo, pues proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales tales como Oracle. (11)
- La característica de PostgreSQL conocida como Write Ahead Logging (escritura por delante del registro), incrementa la dependencia de la base de datos al registro de cambios antes que estos sean escritos en la base de datos, garantizando que en un momento dado, exista un conflicto de conexión u otro problema, existirá un registro de las transacciones a partir del cual se podrá restaurar la base de datos al resolverse la contrariedad existente. (11)

Microsoft SQL Server

Es un sistema de gestión de bases de datos relacionales (SGBD) basado en el lenguaje Transact-SQL, y específicamente en Sybase IQ (potente motor de bases de datos), capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. (1)

Microsoft SQL Server constituye la alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son *Oracle*, *Sybase ASE*, *PostgreSQL*, *Interbase*, *Firebird* o *MySQL*.

Características de *Microsoft SQL Server*

- Soporte de transacciones.
- Escalabilidad, estabilidad y seguridad.
- Soporta procedimientos almacenados.
- Incluye también un potente entorno gráfico de administración, que permite el uso de comandos DDL y DML gráficamente.
- Permite trabajar en modo cliente-servidor, donde la información y datos se alojan en el servidor y las terminales o clientes de la red sólo acceden a la información.
- Además permite administrar información de otros servidores de datos.

Microsoft SQL Server incluye una versión reducida, llamada MSDE (Microsoft SQL Server 2000 Desktop Engine) con el mismo motor de base de datos pero orientado a proyectos más pequeños, que en sus versiones 2005 y 2008 pasa a ser el SQL_Express Edition, se distribuye en forma gratuita. Para el desarrollo de aplicaciones más complejas (tres o más capas), Microsoft SQL Server incluye interfaces de acceso para varias plataformas de desarrollo, entre ellas .NET, pero el servidor sólo está disponible para Sistemas Operativos Windows. (1)

1.3.1.1 Selección del gestor de base de datos a utilizar.

De acuerdo a las características del negocio tales como, gran flujo de datos enviados simultáneamente, conectividad muy baja o casi nula en varias zonas donde se gestiona los datos, información muy dependiente de otra, además, en este campo se utilizan muchos software privativos por lo que el despilfarro de dinero se hace inmenso, asimismo, existe un alto nivel de seguridad en la compañías que

manejan estos datos. Debido a las particularidades del entorno antes expuestas y las características que PostgreSQL nos ofrece mencionadas con anterioridad, tales como; Write Ahead Logging, permite en caso de problemas de conexión restaurar la base de datos, es decir, incrementa la dependencia de la base de datos al registro antes de que los mismos sean escritos en la BD y al restablecerse la conexión podrá recuperarse la información, igualmente implementa la tecnología MVC, la cual nos facilita acceder de manera concurrente a la información de nuestra base de datos por muchos usuarios; además de brindar una alta seguridad de la información, es un software multiplataforma y libre, como también cumple con la integridad referencial de sus datos, entre otras, se decidió optar por este SGBD, esto se encuentra más fundamentado en el trabajo de diploma del rol de arquitecto del SIPP, *(ver más (30)).*

1.3.2 Herramientas de Gestión de Bases de Datos

Las herramientas de gestión de BD es un conjunto de funcionalidades que posibilitan administrar, manejar las Bases de datos desde un gestor cualesquiera.

PhpMyAdmin

Es un proyecto de código abierto creado con el lenguaje de programación PHP, con la finalidad de facilitar la administración las bases de datos de MySQL, apoyado en la ayuda de un navegador, a través de una interface Web, con la posibilidad de hacer diferentes acciones tales como crear, editar y borrar tanto bases de datos y *tablas*, ejecutar sentencias SQL, exportar e importar información a diferentes formatos, etc.

Alguna de las características de PhpMyAdmin:

- Administración completa de las Bases de datos.
- Administración completa de las tablas.
- Ejecuta sentencias SQL.
- Exporta datos a diferentes formatos.
- Administra usuarios y privilegios de MySQL.
- Es un administrador multiplataforma.
- Es multilenguaje, se encuentra en este momento traducido en más de 50 lenguajes diferentes.
- Escrito en el lenguaje de programación PHP4 compatible con PHP5.

Cómo requisito fundamental, este debe tener instalado y configurado un servidor web con soporte para PHP, además de un motor de Bases de datos, que es justamente el que va a brindar una interface amigable para su administración, y por último un Navegador Web o Browser.(6)
La desventaja que posee es que **no es gratuito**.

PhpPgAdmin

PhpPgAdmin es un poderosa herramienta de administración basada en un interfaz Web para bases de datos PostgreSQL. Además de sus funcionalidades básicas como almacenar, y gestionar toda la información a manejar, dispone de soportes para procedimientos almacenados, triggers y vistas. Las versiones de punta van mano a mano con el desarrollo del servidor PostgreSQL. Esta versión es una de la más famosa de los administradores GUI para PostgreSQL (5).

PSQL

Es la herramienta canónica para la ejecución de sentencias SQL a través del shell del SO. Es una herramienta de tipo frontend que permite describir sentencias SQL, ejecutarlas y visualizar sus resultados. El método de ingreso puede ser mediante la inserción directa del código en la consola, o la ejecución de sentencias dentro de un archivo de texto. Provee de diversas metas-comandos para la ejecución de las sentencias, así como diversas opciones tipo shell propias de la herramienta (5).

PgAdmin

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

PgAdmin III está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar

scripts programados, soporte para el motor de replicación Slony-I y mucho más. La conexión al servidor puede hacerse mediante conexión TCP/IP o Unix Domain Sockets (en plataformas *nix), y puede encriptarse mediante SSL para mayor seguridad. (10)

Características incluidas (9):

- Entradas SQL aleatorias.
- Pantallas de información y Ayudas para bases de datos, tablas, índices, secuencias, vistas, programas de arranque, funciones y lenguajes.
- Preguntas y respuestas para configurar Usuarios, Grupos y Privilegios.
- Control de revisión con mejora de la generación de script.
- Configuración de las tablas de Microsoft MSysConf.
- Ayudas para importar y exportar datos.
- Ayudas para migrar Bases de datos.
- Informes predefinidos en bases de datos, tablas, índices, secuencias, lenguajes y vistas.

1.3.2.1 Selección de la Herramienta de Gestión de Bases de Batos a utilizar.

Se decidió utilizar como herramienta de gestión de bases de datos al PgAdmin III, ya que esta es la más utilizada por el SGBD PostgreSQL, más completa y popular en OpenSource. Es multiplataforma, además de ser libre, brinda buena seguridad de sus datos, ofrece una buena ayuda a los usuarios, conjuntamente de consejos de cómo gestionar mejor la información almacenada, posibilita la facilidad de replicar su información con herramientas como el Slony, PgCluster (solo compatible con Linux) y herramientas compatibles con él, ejemplo el **DEW**, sistema implementado en la UCI, la cual permite realizar réplicas de las BD utilizando los entornos maestro-esclavo y multi-maestro. Así como también esta herramienta viene incluida en la instalación del SGBD PostgreSQL, esto se encuentra más fundamentado en el trabajo de diploma del rol de arquitecto del SIPP, *(ver más (30)).*

1.3.3 Herramientas de Modelado de Bases de datos

Las herramientas de modelado de Bases de datos tienen como objetivo definir, crear, modelar, importar, exportar Bases de datos desde un software específico a partir de funcionalidades que los mismos incluyen con el fin de obtener un modelo de BD los más optimo posible. A continuación se muestran una serie de herramientas usadas a nivel mundial con sus respectivas características.

Embarcadero ER/Studio

ER/Studio es una herramienta CASE para el modelado y diseño de BD, que brinda productividad en el diseño, generación, y mantenimiento de aplicaciones. Desde un modelo lógico de los requerimientos de información, hasta el modelo físico perfeccionado para las características específicas de la BD, permite visualizar la estructura, los elementos importantes, y optimizar el diseño de la BD. (29)

La misma presenta las siguientes funcionalidades (7):

- Capacidad fuerte en el diseño lógico.
- Sincronización bidireccional de los diseños lógico y físico.
- Construcción automática de base de datos.
- Reingeniería inversa de base de datos.
- Documentación basada en HTML.
- Un Repositorio para el modelado.
- Genera automáticamente tablas y miles de líneas de procedimientos almacenados y disparadores para los principales tipos de base de datos.

Es una herramienta de modelado de datos fácil de usar y multinivel, para el diseño y construcción de bases de datos a nivel físico y lógico. Direcciona las necesidades diarias de los administradores de bases de datos, desarrolladores y arquitectos de datos que construyen y mantienen aplicaciones de bases de datos muy grandes. (7)

ER/Studio está equipado para crear y manejar diseños de bases de datos funcionales y confiables. Ofrece fuertes capacidades de diseño lógico, sincronización bidireccional de los diseños físicos y lógicos, construcción automática de BD, documentación y fácil creación de reportes. Soporta el proceso de diseño iterativo inherente en el ciclo de vida de la aplicación. Las capacidades de diseño que contiene, ayudan a crear un diseño lógico que puede transformarse múltiples diseños físicos. (29)

Visual Paradigm

“Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El

software de modelado UML ayuda a una 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. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

Características:

- Soporte de UML
- Diagramas de Procesos de Negocio - Proceso, Decisión, Actor de negocio, Documento Modelado colaborativo con Subversion (nueva característica), Interoperabilidad con modelos UML2 (meta-modelos UML 2.x para plataforma Eclipse) a través de XMI (nueva característica).
- Ingeniería de ida y vuelta
- Ingeniería inversa
  - Código a modelo, código a diagrama.
  - Ingeniería inversa Java, C++, Esquemas XML, XML, .NET exe/dll, CORBA IDL. Generación de código.
  - Editor de Detalles de Casos de Uso.
- Entorno todo en uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso Diagramas EJB.
  - Visualización de sistemas EJB.
  - Generación de código y despliegue de EJB´s.
  - Diagramas de flujo de datos Soporte ORM.
  - Generación de objetos Java desde la bases de datos.
  - Generación de bases de datos.
  - Transformación de diagramas de Entidad-Relación en tablas de base de datos Ingeniería inversa de bases de datos. Desde SGBD existentes a diagramas de Entidad-Relación.
  - Generador de informes para generación de documentación.
  - Distribución automática de diagramas
  - Reorganización de las figuras y conectores de los diagramas UML
  - Importación y exportación de ficheros XMI.
  - Integración con Visio.

- Dibujo de diagramas UML con plantillas (stencils) de MS Visio.
- Editor de figuras

Otras herramientas y plugins de modelado UML:

- Plataforma Java (Windows/Linux/Mac OS X):
- SDE para Eclipse
- SDE para NetBeans
- SDE para Sun ONE
- SDE para Oracle JDeveloper
- SDE para JBuilder
- SDE para IntelliJ IDEA
- SDE para WebLogic Workshop
- SDE para Microsoft Visual Studio "(8)

CASE Studio

"CASE Studio es una herramienta profesional con la que se pueden diseñar bases de datos, la cual facilita la creación de diagramas de relación, modelado de datos y gestión de estructuras. Tiene soporte para trabajar con una amplia variedad de formatos de base de datos (Oracle, SQL, MySQL, PostgreSQL, Access, etc.) y permite además generar scripts SQL, aplicar procesos de retro-ingeniería a las bases de datos, usar plantillas de diseño personalizables y crear informes en HTML y RTF.

Soporta Oracle, MySQL, MS SQL, MaxDB, Firebird, PostgreSQL y otros sistemas. El programa permite generar rápidamente diagramas gráficos de las bases de datos relacionales con las que se trabaja, simplificando mucho el trabajo del programador. Permite realizar Diagramas Entidad-Relación (DER) y Diagramas de Flujos de Datos (DFD) para distintos motores de base de datos. Su principal característica, es su potente sistema de ingeniería inversa, o sea, a partir del modelo de tablas llegar al modelo lógico. Mediante Case Studio se podrá realizar diagramas de flujo con muy poco tiempo y esfuerzo. Permite la generación de disparadores (triggers) para realizar validaciones en las entidades que formarán parte de las tablas de la BD."(26)

1.3.3.1 Selección de la herramienta de modelado a utilizar.

En este caso Visual Paradigm, se presenta como la mejor opción para el modelado del problema teniendo en cuenta que es una herramienta en que los diseñadores poseen cierta experiencia previa. Además ofrece sencillez y claridad en su trabajo. Su opción de repositorio permite el trabajo en equipo muy útil en este caso que es un proyecto grande, y con gran cantidad de información a modelar, es capaz de admitir tipos de datos definidos por el usuario, posee un entorno fácil de manipular y de entender, genera código SQL, así como, para lenguajes de programación más usados en el mundo (C#, C++, Java, PHP, entre otros), permitiéndole una mejor facilidad a los programadores, esto se encuentra más fundamentado en el trabajo de diploma del rol de arquitecto del SIPP, *(ver más (30)).*

Conclusiones Parciales

Luego del estudio realizado sobre las bases de datos y otros aspectos necesarios para poder desarrollar la BD del Sistema de Información de Pozos de Perforación en Perforación:

- Se optó por el modelo de base de datos relacional, pues en este modelo la información puede ser recuperada o almacenada mediante "consultas" que ofrecen una amplia flexibilidad y poder para administrar la información, además por todas las características antes expuestas que lo sitúan a nivel mundial como uno de los modelos más usados.
- Se decidió hacer uso de las réplicas en los pozos de petróleo debido a la mala conexión que existen entre ellos, la DIPP y el CEINPET.
- Se decidió utilizar el SGBD PostgreSQL 8.2 para el almacenamiento de los datos del SIPP, por todas las ventajas y características que el mismo presenta y en conveniente con las particulares que tiene el entorno estudiado.
- Se determinó hacer uso del PgAdmin III para la administración de la base de datos.
- Se escogió Visual Paradigm para el modelado del diagrama de clases persistentes y para el modelo de datos.

CAPÍTULO II: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA.

Introducción

En este capítulo se presentan como se integra el sistema con el framework Symfony, arquitectura a desarrollar, además de los requisitos funcionales y no funcionales que debe satisfacer la BD, así como las clases persistentes del SIPP, se describe el diseño físico de la BD.

2.1 Integración de la solución con otros módulos o sistemas

Para el desarrollo del proyecto SIPP, se hizo necesario integrarlo con el framework: Symfony debido a las características expuestas a continuación.

- Reutilización de código.
- Código Libre, es libre, gratuito y además se le pueden implementar o añadir nuevas funcionalidades según las necesidades del problema a desarrollar y del programador.
- Diseño Orientado a Objetos, trata a las tablas de las bases de datos como clases y sus atributos como objetos.
- Abstracción, Encapsulación, Herencia, Polimorfismo.
- Patrones de Diseño, utiliza el patrón de diseño, Modelo Vista-Controlador (MVC)
- Sistema de configuración basado en ficheros, dispone de ficheros, los cuales se utilizan para configurar este framework de acuerdo a las necesidades del programador.
- Buena estructura de sus ficheros, clara y bien definida la forma que maneja sus carpetas que permite localizar rápidamente todos los archivos que se necesiten en un determinado momento.

La información mínima que necesita Symfony para realizar peticiones a la base de datos es su nombre, los datos de acceso y el tipo de base de datos. Esta información se indica en el archivo databases.yml que se encuentra en el directorio config.

Las opciones de la conexión se establecen para cada entorno. Se pueden definir diferentes opciones para los entornos prod (entorno de producción), dev (entorno de desarrollo) y test (ambiente de prueba), o para cualquier otro entorno definido en la aplicación.

Para el caso de la conexión con el PostgreSQL el prefijo que se debe utilizar es pgsql, para modificar el fichero **.yml** y se debe verificar que las librerías (**pdo_pgsql.dll**, **pgsqll.dll**) necesarias están activas en el servidor web que se esté utilizando.

También es posible redefinir esta configuración en cada aplicación, estableciendo diferentes valores para las opciones en un archivo específico de la aplicación, de esta forma se pueden disponer de políticas de seguridad diferentes para las aplicaciones públicas y las aplicaciones de administración del proyecto, y definir distintos usuarios de bases de datos con privilegios diferentes.

Para que este framework pueda interactuar con la bases de datos manejada por el SGBD PostgreSQL es necesario realizar una serie de cambios en algunos de los archivos de configuración que ofrece el gestor. *(Ver Anexo 1)*

2.2 Descripción de la Arquitectura.

La base de datos del Sistema de Información de Pozos en Perforación para Empresas Petroleras en su primera versión está dividida en 1 esquemas de trabajo:

- **public**: esquema implementado por el SGBD PostgreSQL, conformado por 40 tablas dentro de ellas 7 nomencladores, (nbarrena, n_operacion, n_actividad, n_prod_quim, n_unidad_med, herram, n_equipo), este esquema se encarga de almacenar la información que se genera en los pozos de petróleo.

Esta BD estará aplicada en una aplicación web, estructurada por tres capas siguiendo el patrón de arquitectura Modelo Vista-Controlador implementado por el framework Symfony.

2.3 Análisis y Optimización de consultas

A la hora de realizar consultas y procedimientos almacenados es necesario tener en cuenta, que estos deben de estar implementados lo más optimizados posibles, así se obtendrían los resultados en un menor

tiempo y de esta forma no sobrecargar el ambiente de red sobre el cual se esté trabajando. Para ello se tienen en consideración dos factores fundamentales.

- Evaluación de qué consulta es algebraicamente más correcta.
- Valoración de la carga de los recursos del sistema.

La optimización de las consultas incide fundamentalmente en el coste de las comunicaciones de acceso a los almacenamientos secundarios, además del valor computacional que pueda tomar a la hora de realizarlas. Esta se apoya en informaciones estadísticas, tales como cardinalidad de las tablas y de los dominios, es independiente de las estrategias de acceso con respecto de la organización física de la base de datos.

Debido a que las consultas se realizarán sobre el Symfony, por la razón de que este framework nos ofrece una serie de funcionalidades, clases y paquetes como el *Propel, Criteria, Peer, Creole, e*l acceso a los datos de la BD del SIPP se implementará mediante objetos, utilizando la capa ORM de *Propel*, que transforma las llamadas a los objetos en consultas SQL optimizadas para el gestor de bases de datos que se está utilizando y la capa de abstracción de la BD *Creole*. Esto nos facilita en un futuro si se desea cambiar de gestor por cualquier motivo, no haría falta gestionar de nuevo los datos para el otro SGBD seleccionado, además, una vez utilizada la capa de abstracción del Creole se brinda una mayor seguridad de los datos, evitando consultas por *injection* SQL por los usuarios de internet. (3)

Ahora, cuando se quiere obtener más de un registro de la BD, se puede utilizar el método **doSelect** () de la clase *Peer c*orrespondiente a los objetos que se desean. El primer parámetro de este método es un objeto de la clase *Criteria*, para definir consultas simples sin usar SQL para conseguir la abstracción de la base de datos. Cuando se invoca el método **doSelect** (), el cual es mucho más potente que ejecutar una consulta SQL. En primer lugar, se optimiza el código SQL para la base de datos consultada. En segundo lugar, todos los valores pasados a *Criteria* se escapan antes de insertarlos en el código SQL, para evitar los problemas de *injection* SQL. En tercer lugar, el método devuelve un array de objetos y no un **result set**. El ORM crea y rellena de forma automática los objetos en función del **result set** de la base de datos. Este proceso se conoce con el nombre de *hydrating*. (3)

Para las selecciones más complejas de objetos, se necesitan equivalentes a las sentencias WHERE, ORDER BY, GROUP BY y demás de SQL. El objeto *Criteria* dispone de métodos y parámetros para indicar todas estas condiciones. (3)

De manera general el Symfony garantiza la optimización de las consultas y seguridad del sistema a implementar mediante las funciones explicadas anteriormente, además se ajusta al desarrollo de aplicaciones medianas y grandes en diferentes entornos de trabajos.

2.4 Requisitos Funcionales.

Condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente.

1. **Gestionar Reporte Diario de Perforación**
   - ☞ Insertar Datos del Reporte Diario de Perforación
   - ☞ Actualizar Datos del Reporte Diario de Perforación

Este requisito tiene como función gestionar el Reporte Diario de Perforación, el cual elabora el supervisor de pozo y para la confección del mismo se requiere la recopilación de información relacionada con los partes que envía el Direccional del Pozo, el Geólogo del Pozo, el Químico del Pozo, la compañía encargada de la perforación del pozo.

2. **Generar Reporte Diario Operativo del Pozo**

Este requisito tiene como función generar el Reporte Diario Operativo del Pozo el cual se genera a partir de la información obtenida en el Reporte Diario de Perforación

3. **Gestionar Cronograma Diario de Perforación y Metraje**
   - ☞ Insertar Datos del Cronograma Diario de Perforación y Metraje.
   - ☞ Actualizar Datos del Cronograma Diario de Perforación y Metraje.

Este caso de uso se encarga de gestionar Cronograma Diario de Perforación y Metraje el cual es elaborado por el Supervisor de Pozo y consta de una tabla donde describen las relaciones existentes entre el tiempo de explotación del pozo con los metros perforados por el mismo.

4. **Generar Reporte de Record de Barrenas**

Este caso de uso se encarga de generar automáticamente el Reporte de Record de Barrenas el cual es elaborado por el Supervisor de Pozo y consta de una tabla donde se tienen en cuenta una serie aspectos vinculados a las barrenas, así como, su comportamiento durante el tiempo utilizado.

5. **Gestionar Reporte Composición de Herramientas**

- ☞ Insertar Datos de Composición de Herramientas.
- ☞ Actualizar Datos del Composición de Herramientas.

Este caso de uso se encarga de gestionar El Reporte Composición de Herramientas (BHA) el cual es elaborado por el Supervisor de Pozo y consta de una tabla donde se describen las relaciones de los distintos equipamientos, piezas y materiales que se encuentran en el pozo y a que profundidad

6. **Gestionar Reporte de Propiedades de Fluido de Perforación**

- ☞ Insertar Datos de Propiedades de Fluido de Perforación.
- ☞ Actualizar Datos de Propiedades de Fluido de Perforación.

Este caso de uso se encarga de gestionar El Reporte de Fluido de Perforación el cual es elaborado por el Supervisor del Pozo, el cual consta de una tabla con propiedades del lodo a distintas profundidades del pozo. Este reporte se gestiona diariamente.

7. **Generar Reporte de Costo de Productos Químicos**

Este caso de uso se encarga de generar automáticamente El Reporte de Costo de Productos Químicos a partir de los datos enviados por el Químico del pozo, consta de una tabla con la descripción de las relaciones entre los distintos productos químicos utilizados en el proceso de perforación del pozo hasta el momento, el precio de compra y las unidades de media de los mismos.

8. **Gestionar Reporte de Distribución del Tiempo Diario**

Este requisito tiene como función generar automáticamente el reporte de distribución de tiempo diario en el pozo. A partir de los tiempos empleados en las diferentes actividades en el pozo. Estos datos se introducen en el Reporte Diario de Perforación.

9. Generar Reporte de Distribución del Tiempo por Etapas

Este requisito tiene como función generar automáticamente el Reporte de Distribución de Tiempo por Etapas a partir de los Reportes de Distribución de Tiempo Diarios

10. Generar Reporte de Presupuesto Diario

Este requisito tiene como función generar automáticamente el Reporte de Presupuesto diario

11. Generar Reporte de Presupuesto Semanal

Este requisito tiene como función generar la información de los costos semanales del pozo. Esta actividad es realizada por el supervisor el pozo.

12. Generar Resumen de Costo del Pozo

Este requisito tiene como función generar automáticamente el Resumen de Costos del Pozo, a partir del Reporte de Presupuesto Semanal.

13. Gestionar Reporte de Encamisado de Superficie. ***(No se valora en esta iteración)***

- Insertar Datos del Reporte de Encamisado de Superficie.
- Actualizar Datos del Reporte de Encamisado de Superficie.

Este requisito tiene como función gestionar la información resultante del descenso de la camisa de superficie o conductora, más conocida como Tranque de Agua.

14. Gestionar Reporte de Encamisado Técnico. ***(No se valora en esta iteración)***

- Insertar Datos del Reporte de Encamisado Técnico.
- Actualizar Datos del Reporte de Encamisado Técnico.

Este requisito tiene como función gestionar la información resultante del descenso de la camisa técnica o intermedia.

15. Gestionar Reporte de Encamisado Liner *(No se valora en esta iteración)*

- ☞ Insertar Datos del Reporte de Encamisado Técnico.
- ☞ Modificar Datos del Reporte de Encamisado Técnico.

Este requisito tiene como función gestionar la información resultante del descenso de la camisa liner.

16. Generar Reporte de Evaluación Técnica. *(No se valora en esta iteración)*

- ☞ Insertar Datos del Reporte de Encamisado Técnico.
- ☞ Actualizar Datos del Reporte de Encamisado Técnico.

Este requisito tiene como función generar de manera automática el Reporte de Evaluación Técnica, el cual toma información que necesita del Reporte de Distribución de Tiempo.

17. Gestionar Resumen de Pozo. *(No se valora en esta iteración)*

- ☞ Insertar Datos del Resumen del Pozo.
- ☞ Actualizar Datos del Resumen del Pozo.

Este requisito tiene como función gestionar toda la información de generada durante el tiempo de vida del pozo. Este resumen es elaborado por el Supervisor del Pozo.

18. Gestionar Reporte de Construcción de Pozo *(No se valora en esta iteración)*

- ☞ Insertar Datos del Reporte de Construcción de Pozo.
- ☞ Actualizar Datos del Reporte de Construcción de Pozo.

Este requisito tiene como función gestionar la información de la construcción integra del pozo, este reporte se va elaborando a medida que se van terminando lo intervalos de perforación.

19. Gestionar Reporte Diario de Geología

- Insertar Datos del Reporte Diario de Geología
- Actualizar Datos del Reporte Diario de Geología

Este requisito tiene como función gestionar la información generada por el geólogo del pozo. Importante resaltar que para confeccionar este informe el geólogo del pozo se apoya en los reportes de la compañía direccional y la compañía de servicios de lodo.

20. Gestionar Reporte de Lodo

- Insertar Datos de Reporte de Lodo
- Actualizar Datos del Reporte de Lodo

Este requisito tiene como función gestionar la información generada por la Compañía que presta servicios de Lodo al pozo, al cual es enviada al Supervisor del Pozo.

21. Gestionar Reporte Direccional

- Insertar Datos del Reporte Direccional
- Actualizar Datos del Reporte Direccional

Este requisito tiene como función la gestión de la información que proporciona la compañía que presta los servicios direccionales del pozo, la cual es enviada al Supervisor del Pozo.

22. Generar Informe Semanal al MINBAS *(No se valora en esta iteración)*

Este parte se Genera a través de la información recopilada de dos reportes que se envían de cada uno de los pozos en perforación, más informaciones suministradas de otros departamentos de la DIPP de perforación. (No se realizó la visita a estos departamentos)

23. Gestionar Plan Operativo Mensual. *(No se valora en esta iteración)*

Este plan se realiza de informaciones que se envían al Despacho del DIPP de diferentes departamentos de la DIPP *(No se realizó la visita a estos departamentos)*

24. Generar Reporte Diario de Consumo de Combustible de la Perforación

Este Reporte se pudiera generar automáticamente en caso de que exista un reporte de consumo de combustible o uno de los reportes generados en el pozo tuviera esta información.

25. Generar Parte Diario de Perforación

Este requisito tiene como función la generación automática del Parte Diario de Perforación, tomando la información proveniente de los pozos en perforación. Esta información viene íntegra en el Reporte Diario Operativo del Pozo.

26. Gestionar Parte Diario de Intervención

- Insertar Datos del Parte Diario de Intervención
- Actualizar Datos del Parte Diario de Intervención

Este requisito tiene como función la gestión de la información recopilada de los pozos todos los pozos que se encuentran en intervención.

27. Gestionar Parte Diario Operativo de Perforación

- Insertar Datos del Parte Diario Operativo de Perforación
- Modificar Datos del Parte Diario Operativo de Perforación

Este requisito tiene como función gestionar la información diaria de todos los pozos en Perforación e Intervención, así como las actividades del día siguiente y las principales incidencias del día.

28. Gestionar Información Inicial del Pozo

- Insertar Información Inicial del Pozo
- Modificar Información Inicial del Pozo

Este requisito tiene como función gestionar la información inicial necesaria para que la aplicación inicie el trabajo en el pozo.

29. Gestionar Parte Diario de Geología

- Insertar Datos del Parte Diario de Geología
- Actualizar Datos del Parte Diario de Geología

Este requisito tiene como función gestionar el Parte Diario de Geología, el cual es un compendio de las informaciones geológicas de todos los pozos en perforación, esta información es extraída de los Reportes Diario de Perforación de Cada Pozo.

30. Construir la Columna Litológica del Pozo

Este requisito tiene como función agrupar las funcionalidades para construir la columna litológica del pozo, la cual es realizada por un especialista en la materia (geólogos).

31. Generar Informe Final del Pozo *(No se valora en esta iteración)*

Este requisito tiene como función generar de forma automática el Informe Final del Pozo, el cual toma los datos que necesita de varios reportes creados en el tiempo que el pozo estuvo en perforación.

32. Gestionar Usuarios

- Insertar Información de Usuarios
- Actualizar Información de Usuarios
- Eliminar Usuario

Este requisito es uno de los encargados de la seguridad del sistema, tiene como funcionalidad el trabajo con los datos de los usuarios que están autorizados a trabajar con el sistema.

33. Autenticar Usuarios

Este requisito es uno de los encargados de la seguridad del sistema; de esta manera sólo proporciona el el acceso inicial al sistema solo a aquellos usuarios registrados

34. Gestionar Permisos de Usuarios

- Asignar Permisos de Usuarios
- Actualizar Permisos de Usuarios

Este requisito es uno de los encargados de que el sistema sea seguro, además de gestionar los permisos de cada usuario, que no es más que delimitar el horizonte de trabajo del especialista solamente a los lugares donde su trabajo lo requiera.

35. Gestionar Registros de Seguridad

- Guardar Registros de Seguridad
- Consultar Registros de Seguridad

Este requisito es uno de los encargados de la seguridad del sistema, el cual le da la funcionalidad al sistema de almacenar todas las acciones realizadas por los usuarios del sistema.

36. Gestionar Trabajo con Archivos

- Cargar Archivos
- Guardar Archivos
- Exportar Archivos
- Imprimir Archivos

Este requisito se encarga del tratamiento de los reportes como archivos de distintos tipos (.pdf, .txt, .xls, .ppt)

37. Generar Gráficas de Reportes

Este requisito se encarga de visualizar la información de los reportes en gráficas, con el cual se dará la posibilidad de mostrar la información de otra manera.

38. Visualizar Información de Reportes

Este requisito se encarga de mostrar la información contenida en los reportes, lo cual en algún momento serviría para exponer una información determinada de manera más explícita.

39. Consultar Información

Este requisito se encarga de brindar la oportunidad a los usuarios del sistema de buscar cualquier información referente a los pozos que desee. Esto debe ocurrir de manera rápida y precisa, optimizando el tiempo de búsqueda.

2.5 Requisitos No Funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener. Son las características que hacen al producto atractivo, usable, rápido y confiable. (Booch, 2000)

Usabilidad

- Preparar a los Administradores en la gestión de Roles y Permisos.
  Esta acción específica del sistema, lleva un alto grado de complejidad, se debe preparar un curso para instruir a las personas que interactuarán con estas funcionalidades.
- Preparar a los usuarios del sistema en la gestión con los Reportes y Partes.
  Esta acción específica del sistema lleva una complejidad media, se debe preparar una instrucción referente al manual de usuario que acompañará a la aplicación para instruir a los que trabajarán con estas funcionalidades.

- Visibilidad del estado del sistema.
  La aplicación debe mantener siempre informado al usuario del estado del sistema así como de los caminos que este pueda tomar con una retroalimentación visual apropiada en un tiempo razonable. El sistema ofrecerá al usuario una respuesta que le indique lo que está sucediendo en cada una de las operaciones que realiza.
- Control y libertad del usuario.
  Se le proporcionarán salidas de emergencia sin tener que pasar por un grupo de diálogos. Se deberá proporcionar también acciones de deshacer y rehacer sin perder el trabajo realizado hasta el momento. Se debe tener en cuenta que existen diferentes tipos de usuarios, con diferentes niveles de conocimientos informáticos y percepción de la aplicación.
- Consistencia y estándares
  Una buena interfaz contribuye al aumento de la productividad si es consistente en todos los diálogos que desarrolla, basándose en el conocimiento que el usuario ha adquirido con otras aplicaciones y en la aplicación propia. Se debe mantener la consistencia en todas las aplicaciones relacionadas. Deberán implementar las mismas reglas de diseño para mantener la consistencia en toda la interacción. El usuario debe ser capaz de saber en cada momento en qué contexto está

trabajando, de donde viene y a donde va. Esto se puede realizar con indicadores gráficos como iconos o colores diferentes para cada situación.

Rendimiento

- Tiempo de respuesta de transacción (menos de 3 segundos).
- El sistema necesita PC clientes con 256-512 de RAM y un Navegador Web que soporte Javascript.
  Hay funcionalidades que están desarrolladas en javascript por lo que es imprescindible que las PC clientes lo soporten.
- El sistema debe permitir trabajar conectados concurrentemente 250 usuarios como máximo
  La cantidad de usuarios que se encuentran en los pozos no sobrepasan los 6, pero en CEINPET y la DIPP pueden interactuar con la aplicación un número de usuarios que no sobrepasa la cifra antes expuestas.
- El sistema necesita un servidor de base de datos en una PC o servidor con (1 o superior) giga de RAM y 2 discos duros de 160 Giga.

Soporte

- El soporte y/o mantenimiento del sitio no debe detener el servicio.
  No debe existir ningún problema técnico en las PC que utilizarán los usuarios, ya que traería problemas en el funcionamiento de la aplicación por lo que es necesario su chequeo y revisión diaria.
- El framework tiene en cuenta las posibles restricciones de diseño para la aplicación.
  Estos son: ajuste a estándares (una determinada manera de codificar un dato), limitaciones hardware (por los equipos disponibles), seguridad (por los distintos niveles de acceso a la información que deben tener los usuarios), mantenimiento (se debe tener en cuenta la ampliación del sistema), adaptación al entorno y políticas de borrado
- El diseño para la aplicación será de acuerdo a los distintos paradigmas y adecuado con la arquitectura seleccionada.
  El diseño de la aplicación se hará aplicando Programación Orientada a Objetos. Diseño e implementación de una arquitectura flexible, que permita la fácil integración o desintegración de

componentes. La arquitectura debe soportar migrar la interfaz de usuario sin impactos considerables en re-implementación.

Adquisición de Componentes

- Esta gestión la realizará CUPET.
  CUPET se encargará de la adquisición de los componentes necesarios, dígase PC y servidores como soporte para la aplicación.

Interfaz

- Interfaz de interacción con el usuario.
  Interfaces amigables, fáciles de interactuar con ellas.

Requerimientos de Hardware

Servidor Web

- Hardware de la estación de trabajo del servidor Web, donde se ejecutará el sistema.

  Procesador Pentium 3 (o superior)

  Memoria RAM mínima de 256-512 MB o superior
  Disco duro con capacidad de 160 GB

Servidor de Bases de datos

- Hardware de la estación de trabajo servidor de base de datos.

  Procesador Pentium 3 (o superior)

  1GB Memoria RAM

  Almacenamiento en discos SCSI en espejo y capacidad igual o superior a los 160 GB cada disco. Tecnología de respaldo de datos históricos. En el caso de los pozos con 2 discos de 80 gigas es óptimo.
- Hardware de la estación de trabajo del cliente.Procesador Pentium 3 (o superior). Memoria RAM mínima de 256 MB.

Redes

- Redes de la estación de trabajo del cliente

  La red existente en las instalaciones debe ser capaz de soportar transacciones de paquetes de información de al menos 12 máquinas a la vez.

  El tipo de cableado debe ser de par trenzado UTP con soporte de 100Mpbs.

  Debe estar protegido contra fallos de corriente y conectividad, además de parametrizar los tiempos de realización de copias de seguridad. La transmisión se implementarán utilizando el protocolo TCP/IP que permite la recuperación de datos.

Requerimientos de Software

PC Cliente

- Software instalado en la estación de trabajo del cliente

  Sistema Operativo tanto Windows (win9.x o versión superior) como Linux (cualquiera de sus distribuciones).

  El Navegador Web compatible con HTML 2.0 y CSS, podrá ser Netscape 3 (o superior), Mozilla 2.0, Internet Explorer 4.2 (o superior) y compatibles.

PC Servidor Web

- Software instalado en el Servidor Web.

  Servidor Web Apache 2.2. X o superior.

Servidor de Bases de datos

- Software instalado en el Servidor de Base de datos.

  Servidor de bases de datos PostgreSQL8.2.

Seguridad

- Seguridad del sistema (C.I.D).

  Confidencialidad: La información manejada por el sistema está protegida de acceso no autorizado y divulgación.

Integridad: La información manejada por el sistema es objeto de cuidadosa protección contra la corrupción y estados inconsistentes, de la misma forma es considerada igual a la fuente o autoridad de los datos.

Disponibilidad: Los usuarios autorizados (autenticados por dominio y según su roll) se les garantizarán el acceso a la información, los dispositivos o mecanismos utilizados para lograr la seguridad, no ocultarán o retrasarán a los usuarios para obtener los datos deseados en un momento dado.

- Seguridad del sistema (otros aspectos).

  La seguridad se tratará desde la fase de diseño del sistema. Se garantizará un fuerte tratamiento de excepciones. Parte de la seguridad corre por parte de los framework propuestos (Symfony). Los usuarios de la BD solamente tendrán acceso a las tablas y bases de datos a las cuales se les designe según el rol que desempeña, no se utilizarán usuarios con privilegios administrativos para realizar las conexiones entre servidores. Se registrarán y auditarán las trazas dejadas por los usuarios al realizar las diferentes operaciones en el sistema. La programación de las auditorías será programada según corresponda.

Reusabilidad

- Estándares de codificación de la aplicación

  La aplicación se construirá utilizando estándares internacionales y patrones, para facilitar su integración futura, con componentes desarrollados por cualquier empresa y garantizar posibilidades de un mantenimiento ágil. La documentación de la arquitectura debe ser reutilizable para poder documentarla como una familia de productos. Cada módulo del sistema se desarrollará de forma tal, que pueda ser reutilizada o capaz de funcionar por sí solo y no como parte del paquete general.

Portabilidad.

- Portabilidad de la aplicación

  El sistema deberá poder ser usado desde cualquier Sistema Operativo. Además el traslado de la aplicación no llevará muchas complicaciones debido a la propia estructura que brinda el framework para la compilación de la aplicación.

Escalabilidad

- Escalabilidad de la aplicación
  La propia arquitectura en que se encuentra estructurada la aplicación permite que si se necesita migrar por ejemplo de una base de datos a otra, sólo se vea afectada la capa de la modelo sin esto inferir cambios extremos en las vistas de la aplicación.

2.6 Diagrama de Clases Persistentes.

(*Ver Anexo 4*).

2.7 Modelo Físico

Debido a la gran cantidad de atributos contenidos, en las representaciones gráficas se muestran las diferentes clases, algunos de los atributos y sus respectivas relaciones. La descripción de los atributos que cada una de ellas contiene aparece relacionada a continuación de cada modelo.

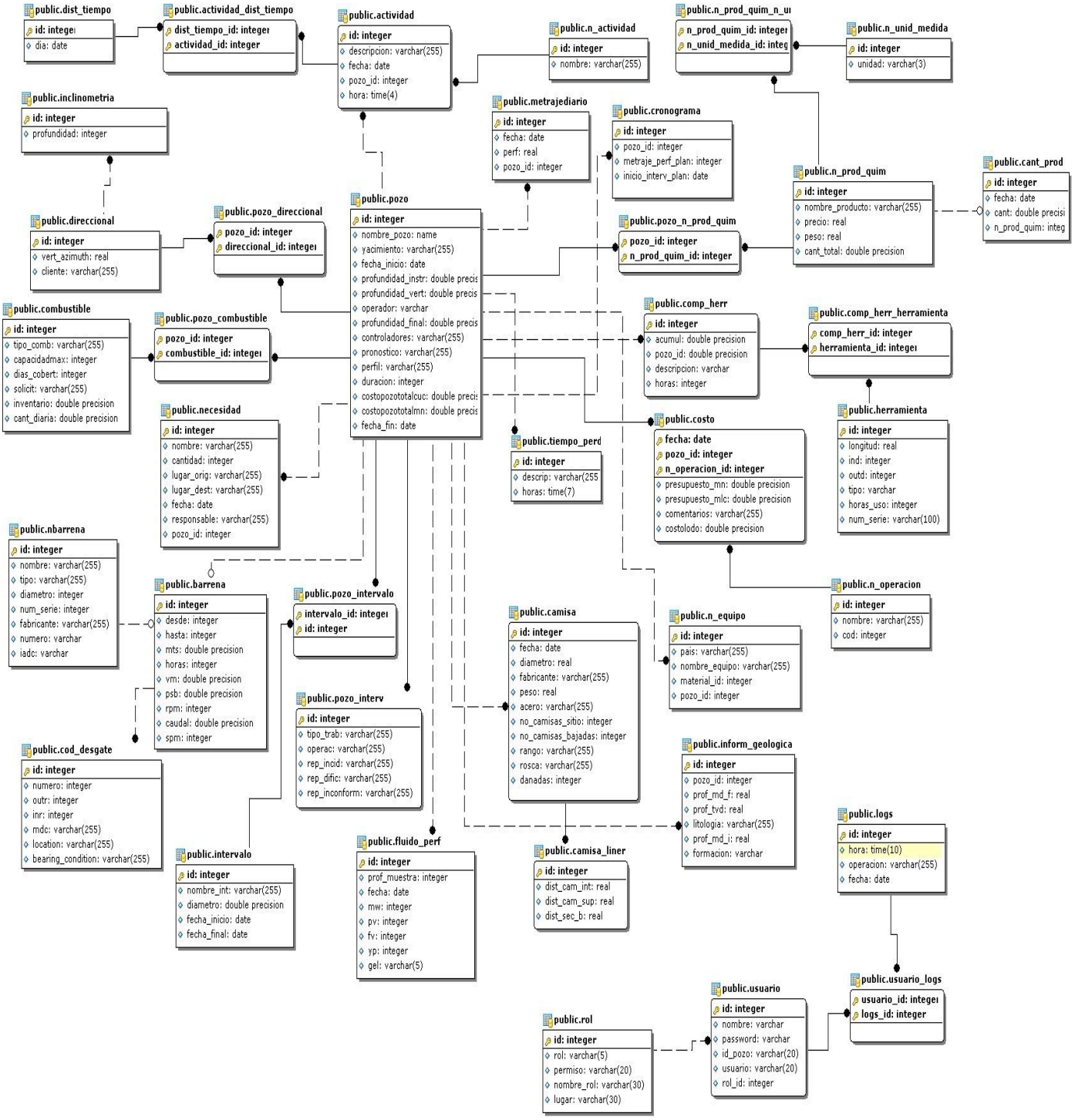

Figura 9. Modelo Entidad Relación.

2.8 Descripción de las tablas.

(Ver Anexo 2)

Conclusiones Parciales

En este capítulo se analizó las clases persistentes implicadas en nuestro negocio, así como, se describió cada una de ellas, se explica cómo se integra el sistema con el framework Symfony, se habla acerca de la importancia de la optimización de las consultas. También se realizó una breve descripción de la arquitectura de cómo quedaría implementadas las clases en el framework utilizado.

CAPÍTULO III: VALIDACIÓN DEL DISEÑO REALIZADO.

Introducción

En este capítulo se brinda información sobre la validación tanto teórica como funcionalmente realizada al diseño de la base de datos propuesta, se muestran las reglas de integridad, como de normalización a tener en cuenta para obtener un diseño con calidad, además de las herramientas para el llenado voluminoso e inteligente de la BD para validarla funcionalmente.

3.1 Validación teórica del diseño.

La validación teórica del diseño incluye fundamentalmente un análisis muy detallado de la integridad de la información, característica altamente deseada, porque asegura la calidad de almacenamiento y disponibilidad de los datos, con su tratamiento se evita errores de entrada introducidos por los usuarios descuidados o cualquier otra circunstancia de intento de violar la información existente en la base de datos.

3.1.1 Integridad

La integridad es uno de los factores más importantes a la hora de realizar el diseño de una base de datos. Esta se divide en varios aspectos como son:

- **Integridad referencial: "**Garantiza interrelaciones válidas entre entidades. Implica que los datos sean correctos, sin duplicaciones, pérdida de datos o relaciones mal resueltas. Todas las bases de datos relacionales incluyen ésta propiedad pues el software gestor es responsable de su cumplimiento.

  Chequeo de validez: Cada nueva tabla describe sus columnas con un tipo específico de datos y el SGBD garantiza que sólo los datos del tipo especificado sean ingresados en la tabla.

  Datos Requeridos: Establece que al realizar una operación de INSERT sobre una fila, una columna marcada como NOT NULL no debe recibir valores nulos.

- **Integridad de Dominio:** La integridad de dominio viene dada por la validez de las entradas de los datos para una columna determinada. Se puede exigir la integridad de dominio para restringir el

tipo mediante tipos de datos, el formato mediante reglas y restricciones CHECK, o el intervalo de valores posibles mediante restricciones FOREIGN KEY, definiciones DEFAULT, definiciones NOT NULL.” (32)

- **Integridad de la Entidad:** define una fila como entidad única para una tabla determinada. La integridad de entidad exige la integridad de las columnas de los identificadores o la clave principal de una tabla, mediante índices y restricciones UNIQUE, o restricciones PRIMARY KEY. (28)

En el caso de la base de datos desarrollada para verificar la integridad referencial se validó en la implementación, por ejemplo en la tabla barrena se necesita el id de la tabla n_barrena y el id del cod_desgaste (ambos llaves foráneas de barrena) para poder añadir o modificar alguna tupla en la tabla barrena es necesario verificar que los identificadores (id de la n_barrena y el id del cod_desgaste) existan en sus tuplas correspondientes. Además de garantizar la integridad de dominio con la validación de que los atributos y los dominios posean los valores que los campos admitan.

3.1.2 Normalización de la Base de datos.

El proceso de normalización de la base de datos es de importancia primordial a la hora de realizar su diseño, el mismo evita la redundancia de información, integridad de los datos a la hora de realizar cualquier operación en las tablas, ya sea de inserción, eliminación o actualización de los datos, y así permite optimizar los procesos de gestión de información almacenada. Este proceso está dividido en varios niveles, siendo los 3 primeros los más importantes.

1ra Forma Normal (FN): Una relación está en 1ra FN si cumple con la propiedad de que sus dominios no contengan elementos que a su vez sean conjuntos.

- La relación no incluye elementos repetitivos.
- Toda relación normalizada, o sea, con valores atómicos de los atributos.

2da FN: Está basada en el concepto de dependencia funcional total, es decir que los atributos no primos dependen totalmente de los atributos llaves y debe de cumplir con lo establecido en la 1ra FN.

La 2da FN se aplica solamente a los esquemas de relación que tienen claves primarias compuestas por dos o más atributos. Si un esquema de relación está en 1FN y su clave primaria es simple entonces está

en segunda forma normal (2FN). Las relaciones que no están en 2FN pueden sufrir anomalías cuando se realizan actualizaciones.

Para la transformación de una relación de 1FN a 2FN hay que eliminar las dependencias parciales de su clave primaria. Para ello, se eliminan los atributos que son funcionalmente dependientes y se descomponen en una nueva relación con una copia de los atributos de la clave primaria de los que dependen.

3era FN: Está basada en el concepto de dependencias transitivas.

- Si y sólo si, la relación está en 2da FN.
- Ningún atributo no primo de R depende transitivamente de la clave primaria.

O sea: Los atributos no llaves deben depender de toda la llave (2FN) de manera directa y no porque dependan de otro atributo no llave de la relación (ejemplo una llave foránea) que a su vez depende de la llave primaria.

La base de datos del SIPP actualmente se encuentra en 3era FN, porque cumple con las especificaciones de las tres formas expuestas anteriormente, es decir en ella no existen atributos multievaluados, dependencias transitivas entre las relaciones y redundancia de la información. La 3era FN es la más usada casi en la totalidad de los productos que utilizan bases de datos , puesto que estas garantizan poca o una redundancia casi nula de información, además esta hace disponer de buena optimizaciones de las consultas, factor muy fundamental en el diseño y la implementación de una BD.

3.1.3 Análisis de redundancia de información.

La redundancia de información se refiere a aquellos datos duplicados que genera inconsistencia en la BD, requiriendo más espacio en disco; sin embargo en proyectos grandes es imposible evitarla completamente, lo que a veces se hace que se desee por cuestiones de rendimiento.

Los procesos de normalización mejoran en buena medida el comportamiento de este parámetro. En el caso de la base de datos del SIPP, no existe información redundante, en la misma se analizaron las tablas con sus respectivas relaciones y se fueron eliminando consecuentemente todos los datos reiterativos en la base de datos.

3.1.4 Trazabilidad de la acciones.

Trazabilidad: Es la capacidad que tiene una organización o sistema para rastrear, detectar, reconstruir o restablecer relaciones entre objetos monitoreados para analizar y especificar situaciones específicas (Jeimy J. Cano, 2005).

La trazabilidad de las acciones que ocurren en la BD serán manejada desde la misma aplicación, es decir que a la hora de efectuarse una nueva función se realizará una consulta con la respectiva acción realizada y se le insertará en la base de datos, con el nombre de la acción, una descripción y la fecha con la hora que se realizó la misma, nuestro módulo de seguridad garantiza esto.

3.1.5 Análisis de seguridad de la BD.

Las bases de datos son víctimas constantemente de ataques por piratas informáticos que tratan de acceder con el objetivo de robar o modificar alguna que otra información o simplemente violar la seguridad del sistema. Debido a esto cada BD debe de tener bien implementada una buena seguridad para evitar estos acometidos.

Los SGBD existentes garantizan en cierta medida el control de acceso por parte de los usuarios a la BD de acuerdo a los privilegios que tengan asignados y los lugares donde ellos se encuentren ubicados, así como los datos manejados deben de estar encriptados, garantizando así su seguridad.

En nuestra base de datos se tuvieron en cuenta todos los términos referidos al control de acceso, integridad y disponibilidad de los datos. El framework Symfony te ofrece una serie de aspectos de seguridad los cuales se encuentran en el fichero security.yml, perteneciendo este a la carpeta config del módulo en el cual se esté trabajando, ejemplo: (apps/frontend/modules/mi_modulo/config/security.yml), en este fichero existen varias configuraciones de variables, por ejemplo: is_secure, se encuentra en off, es decir, que este módulo es público a los usuarios, así estén o no logueados, en caso de que esté en **on** (activado) sólo es posible acceder a ella logueándose al sistema.

Aparte de estos elementos ofrecidos por el framework utilizado, se decidió crear un esquema de seguridad compuesto por 4 tablas (usuario, rol, logs, usuario-logs) para validar los usuarios con los respectivos

permisos a tener en cuenta a la hora de autenticarse al sistema y almacenar las acciones que se realicen en el mismo.

3.1.6 Análisis sobre el acceso a datos. Implementación.

Todas las funciones se implementarán sobre el framework de Symfony, debido a todas las facilidades de consultas que este nos ofrece mencionadas anteriormente en el **Capítulo 2 Epígrafe 2.3**.

El Symfony crea una capa de abstracción donde construye por cada tabla de la base de datos 4 clases (BaseBarrena, Barrena, BarrenaPeer, BaseBarrenaPeer) con atributos y métodos para acceder a ellas, guardando esta información en varios ficheros con los nombres de dichas tablas en la carpeta *lib/model*. A la hora de realizar las consultas este implementa una clase *Criteria***:** la cual permite mediante el acceso a todos los objetos, facilitar la creación de las distintas funcionalidades, guardando las consultas en la clase *Peer*, esta nos da la posibilidad de almacenar todas estas funciones. A continuación se muestra un ejemplo de una consulta, de la misma se obtienen todos los intervalos donde el nombre del intervalo no sea nulo.

$c=**new Criteria ();**

$c->**add (**IntervaloPeer::NOMBRE_INT,null,**Criteria**::NOT_EQUAL**)**;
$**this->exec**=IntervaloPeer::**doSelect(**$c**)**;

3.2 Validación funcional.

3.2.1 Herramienta para un llenado voluminoso e inteligente de la base de datos.

A la hora de realizar el llenado de datos de la Base de Batos desarrollada se decidió utilizar la herramienta del PS/SQL Developer PostgreSQL DataGenerator; esta permite un llenado voluminoso con la cantidad de datos que se desee, entrar valores desde otras tablas garantizando la integridad referencial existente, y cada vez que van incrementando la cantidad de información añadida el tiempo de llenado se irá haciendo más pequeño.

Para realizar esta operación se seleccionaron las siguientes tablas, las cuales arrojaron los estos resultados.

1era Prueba		
Tabla	Cantidad de Registros	Generados a la BD (seg)
pozo	2000	11
n_operacion	2000	5
costo	2000	6
2da Prueba		
Tabla	Cantidad de Registros	Generados a la BD (seg)
pozo	20000	47
n_operacion	20000	37
costo	20000	48

Tabla 1 Resultados de las Pruebas de Volumen

Para las restantes tablas se generaron diferentes cantidades de datos y se obtuvieron tiempos de respuesta adecuados, teniendo en cuenta el gran volumen de datos insertados y la búsqueda que realiza el software en el catálogo de llaves para validar las llaves foráneas y los dominios de los campos.

Conclusiones Parciales

En este capítulo se han analizado aspectos a tener en cuenta como son de integridad, normalización, redundancia de la información a la hora de validar teórica como funcionalmente el diseño de la base de datos para el SIPP y como el framework utilizado implementa las funcionalidades se integra la base de datos al mismo.

Conclusiones

Con el desarrollo del presente trabajo de diploma se lograron todos los objetivos trazados:

- Se realizaron investigaciones acerca de los sistemas de bases de datos.
- Se utilizaron herramientas multiplataforma, pertenecientes a la familia del software libre, debido a las características y condiciones actuales que existen en nuestro país y en el mundo de la informática.
- Se decidió hacer uso de PostgreSQL como SGBD debido a sus particularidades y ventajas antes expuestas con respecto a otros sistemas gestores de bases de datos.
- Se utilizó el Symfony como framework de desarrollo web conjuntamente con php5.
- Se recurrió al Visual Paradigm como herramienta CASE para el modelado de los datos.

Se debe complementar que se obtuvo el diseño de una base de datos relacional utilizando réplicas, específicamente el entorno maestro-maestro para garantizar la disponibilidad de los datos, en caso de problemas con la conexión, se analizó la integridad y la redundancia de información. Cuenta con todos los requisitos funcionales implementados, se logra además almacenar toda la información con la mayor seguridad posible. Indistintamente se le hicieron pruebas de volumen a la BD, ya que la información generada es un tanto grande, de las cuales los resultados fueron satisfactorios.

Referencias Bibliográficas

1. Características de SQL Server publicado el: 16/04/2008 de última actualización: 16/04/2008. Disponible en: http://www.shica19.tripod.com/sql.html.

2. Article_MySQL-PostgreSQL. Disponible en: http://www.phpbuilder.com/columns/tim20000705.php3?page=1 .

3. **Potencier Fabien, Zaninotto François**. *Symfony La Guía Definitiva*. 2008.

4. Ventajas de PostGreSQL publicado el: 09/01/2008 de 2007, última actualización: 09/01/2008. [Consultado el: 09/01/2008]. Disponible en: http://soporte.tiendalinux.com/portal/Portfolio/postgresql_ventajas_html.

5. **Espinoza, Humberto**. *PostgreSQL, una nueva alternativa de DBMS Open Source*. 2005.

6. http://www.aplicacionesempresariales.com/gestionando-las-bases-de-datos-mysql-con-phpmyadmin.html

7. Modelado de Base de Datos, ER-Studio 6.0.1. Disponible En: http://www.taringa.net/posts/downloads/884761/Modelado-de-Base-de-Datos-,--ER-Studio-6_0_1.html

8. Free Download Manager. Free Download Manager. [En línea] [Citado el: 25 de noviembre de 2008.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(Iglesia_Anglicana)_%5Bcuenta_de_Linux_14716_p/

9. http://es.tldp.org/Postgresql-es/web/navegable/user/app-pgadmin.html

10. http://www.guia-ubuntu.org/index.php?title=PgAdmin_III

11. http://www.manualesdeayuda.com/manuales/bases-de-datos/postgresql/

12. **Silverio Castro, Rogelio S.** *La problemática de las Bases de datos Distribuidas.* 2005.

13. Free Download Manager. Free Download Manager. [En línea] [Citado el: 25 de noviembre de 2008.] http://www.freedownloadmanager.org/es/downloads/Bien_Maderero_7024_p/.

14. WellSight. Wellsight.com. [En línea] 3 de 3 de 2009. [Citado el: 6 de diciembre de 2008.] http://www.wellsight.com

15. AddLink. AddLink. [En línea] [Citado el: 22 de 11 de 2008.] http://www.addlink.es/productos.asp?pid=430.

REFERENCIAS BIBLIOGRÁFICAS.

16. CAVSI. CAVSI. [En Línea] [Citado el: 22 de 11 de 2008.] http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd/
17. http://html.rincondelvago.com/base-de-datos_3.html
18. **Silvente, Serguéi Frómeta.** *Modelo lógico y físico de la base de datos del módulo de Investigaciones Forenses del proyecto CICPC*. Ciudad de la Habana.
19. Leyva, Arnoldo Domínguez. *Modelo Lógico y Físico de la Base de Datos.*
20. http://office.microsoft.com/es-es/access/HA012242473082.aspx
21. **Marqués Andrés, María Mercedes**. *Apuntes de Ficheros y Bases de Datos.* 2001 Disponible en: http://www3.uji.es/~mmarques/f47/apun/node1.html
22. **Rodríguez González, Evelio**. *Perfeccionamiento de la tecnología y desarrollo de dispositivos mecánicos para la perforación inclinada de pozos de petróleo.*
23. **ELMASRI; NAVATHE.** *Arquitectura Base De Datos.* Disponible en: http://www.mitecnologico.com/Main/ArquitecturaBaseDeDatos
24. **Pérez, Sara**. *Modelos de Bases de Datos*. Disponible en: http://www.desarrolloweb.com/articulos/modelos-base-datos.html
25. **Pérez, Sara**. *Arquitectura de las Bases de Datos*. Disponible en: http://www.desarrolloweb.com/articulos/arquitectura-base-de-datos.html
26. http://es.wikipedia.org/wiki/Base_de_datos
27. **Frómeta Silvente, Serguéi**. *Modelo lógico y físico de la base de datos del módulo de Investigaciones Forenses del proyecto ICPC*.
28. **Microsoft**. *Integridad de los datos*. Disponible en: http://msdn.microsoft.com/es-es/library/ms184276.aspx
29. http://www.netpecos.org/docs/mysql_postgres/x57.html
30. **Buzón Tur, Michel.** *Propuesta de Arquitectura para el SIPP.*
31. **Fajardo Vega, Norge**. *Sistema de réplica para bases de datos distribuidas en PostgreSQL*.
32. **Fernández Macías, Dayron** *Diseño de una base de datos para controlar la información de Unión de Jóvenes Comunistas en la UCI, 2008.*

Bibliografía

1. **W. Hansen, Gary and V. Hansen, James.** *Diseño y Administración de Bases de datos .*

2. **Silverio Castro, Rogelio S.** *La problemática de las Bases de datos Distribuidas.* 2005.

3. **Jaen de La Vega, Jose David and Secades Pastor, Francisco Javier.** *Base de Datos Distribuidas*.

4. **Informática, Departamento de.** BASES DE DATOS DISTRIBUIDAS . [Online] http://cmapspublic.ihmc.us/rid=1161027353218_44637313_464/2.pdf.

5. **Gonzalez Martin, Oscar and Ruiz Gonzalez, Francisco.** *Arquitectura de Sistemas de Bases de datos .* 1999/2000.

6. **Espinoza, Humberto.** *PostgreSQL, una nueva alternativa de DBMS Open Source.* 2005.

7. **Escoí, Francisco D. Muñoz.** Revista del Instituto Tecnológico de Informática. *Revista del Instituto Tecnológico de Informática.* [Online] [Cited: 02 16, 2009.] http://web.iti.upv.es/actualidadtic/2004/03/2004-03-globdata.pdf.

8. **Digitales, Dpto Sistemas.** http://teleformacion.uci.cu/. [Online] 1.0, 2006. [Cited: 12 01, 2008.] http://teleformacion.uci.cu/mod/resource/view.php?id=3741.

9. **Diaz, A.** *Sistema de Bases de datos Distribuidas.*

10. **Espinoza, Humberto.** *PostgreSQL, una nueva alternativa de DBMS Open Source. 2005.*

11. **KRONOS, J. T.** *Introducción a la Documática. Los sistemas de bases de datos y los SGBD.*

12. **Tripod.** *Tripod.* [Online] http://www.shica19.tripod.com/sql.html.

13. **Php-Builder.** [Online] http://www.phpbuilder.com/columns/tim20000705.php3?page=1.

14. **TiendaLinux.** [Online] [Cited:] http://soporte.tiendalinux.com/portal/Portfolio/postgresql_ventajas_html.

15. **Aplicaciones Empresariales**. [Online] http://www.aplicacionesempresariales.com/gestionando-las-bases-de-datos-mysql-con-phpmyadmin.html.

16. **Taringa**. [Online] http://www.taringa.net/posts/downloads/884761/Modelado-de-Base-de-Datos-,--ER-Studio-6_0_1.html.

BIBLIOGRAFÍA

17. **FreeDownloadManager.** [Online] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_(Iglesia_Anglicana)%5Bcuenta_de_Linux_14716_p/.

18. **TDLP**. [Online] http://es.tldp.org/Postgresql-es/web/navegable/user/app-pgadmin.html.

19. **Ubuntu.** [Online] http://www.guia-ubuntu.org/index.php?title=PgAdmin_III.

20. **Tramullas**. [Online] http://tramullas.com/documatica/2-4.html.

21. **FreeDownloadManager**. [Online] http://www.freedownloadmanager.org/es/downloads/Bien_Maderero_7024_p/.

22. **WellSight.** [Online] http://www.wellsight.com.

23. **AddLink**. [Online] [Cited: Diciembre 14, 2008.] http://www.addlink.es/productos.asp?pid=430..

24. **Peloton**. [Online] http://www.peloton.com/es.

25. **DE BASES DE DATOS AVANZADAS, DPTO.** *BASES DE DATOS DISTRIBUIDAS.*

26. **SyBase.** *Replication Strategies: Data Migration, Distribution and Synchronization.* 2003.

27. **López, Guadalupe Lamadrid, Lissethe.** *Bases de datos Distribuidas .*

28. **Pérez Mora, Oscar; Gibert Ginesta, Marc.** *Base de Datos en Postgre.*

29. **Avanzadas, Grupo de Bases de datos .** *Fundamentos de Bases de datos Distribuidas.*

30. **Olarte, Carlos A.** *Bases de datos Distribuidas.*

31. **OEI, Dpto de.** *Diseño y Optimización de Bases de datos . Bases de datos Distribuidas.*

32. **Monge, Raul.** *"Base de Datos Distribuidas: Replicación".*

33. **CAVSI** [Online] Cited: Noviembre 22, 2008]. http://www.cavsi.com/preguntasrespuestas/que-es-un-sistema-gestor-de-bases-de-datos-o-sgbd

34. **Leyva, Arnoldo Domínguez**. *Modelo Lógico y Físico de la Base de Datos.*

35. **Microsoft.** [Online] http://office.microsoft.com/es-es/access/HA012242473082.aspx

36. Marqués Andrés, María Mercedes. *Ficheros y Bases de datos.* 2001

37. **ELMASRI; NAVATHE.** *Arquitectura Base De Datos.* [Online]: http://www.mitecnologico.com/Main/ArquitecturaBaseDeDatos

38. Pérez, Sara. *Modelos de Bases de datos*. [Online] Cited: enero 26, 2008]: http://www.desarrolloweb.com/articulos/modelos-base-datos.html

39. Frómeta Silvente, Serguéi. *Modelo lógico y físico de la base de datos del módulo de Investigaciones Forenses del proyecto ICPC*.

40. Fajardo Vega, Norge. *Sistema de réplica para bases de datos distribuidas en PostgreSQL.*

41. Buzón Tur, Michel. *Propuesta de Arquitectura para el SIPP, 2009.*

42. Potencier Fabien, Zaninotto François. *Symfony La Guía Definitiva*. 2008.

43. Fernández Macías, Dayron *Diseño de una base de datos para controlar la información de Unión de Jóvenes Comunistas en la UCI, 2008.*

GLOSARIO DE TÉRMINOS

CAD: Estilo para el tratamiento de datos gráficos.

Creole: Sistema de abstracción de la base de datos, que proporciona una interfaz entre el código PHP y el código SQL de la base de datos, permitiendo cambiar fácilmente de sistema gestor de bases de datos.

Criteria: Objeto de una clase cualesquiera, que permite manejar condiciones de cualquier nivel de complejidad.

DBR: Brinda datos según sus reportes diarios los cuales son consultados en la toma de decisiones.

DISWIN: Proporciona capacidades de envío completas a todas las empresas de transporte por camión clasificadas. DISWIN es el software para maximizar la productividad y hacer ahorrar tiempo y el dinero.

Dominio: Conjunto de valores de los cuales los atributos obtienen sus valores.

DST: Genera informes relacionados con la Dirección de Servicios Tecnológicos.

Landmark: Integración visual y orientada a la secuencia de tareas utilizando la misma visión 3D para todos los datos sísmicos y de pozo, interpretaciones, modelado y simulación de reservorios y planeamiento de perforaciones.

Llave Primaria: Llave con valores únicos, es decir, no ocurren más de una vez en el atributo.

Tupla: Hilera o fila en una tabla.

Peer: son clases que contienen métodos estáticos para obtener registros de la bases de datos, con la cual se está trabajando.

Propel: capa de abstracción que se utiliza para el ORM. Proporciona persistencia para los objetos y un servicio de consultas optimizadas.

WITSML: Well-Site Information Transfer Standard Markup Language. Lenguaje para marcar normas para transferir información del sitio del pozo.

ANEXOS

Anexo 1

Conexión a la Base de Datos

Para lograr la conexión a la BD con el SGBD PostgreSQL, se configuran los archivos postgresql.conf y el pg_hba.conf. En el primero, se busca la configuración llamada --Connection Settings-- y dentro de ella listen_addresses, ahí se debe de especificar si se desea que la conexión sea solo de tu equipo (localhost) o que todos se puedan conectar (*), además del puerto (port) por el cual se conectarán a la base de datos (5432), pero si se quiere mantener este, se añade a los puertos permisibles por el firewall de Windows; en otro caso se especifica el deseado. En el segundo citado se localiza la configuración llamada --IPv4 local connections--, añadiéndole las bases de datos, los usuarios, los host a permitir y el método de encriptación de los datos a utilizar, el cual viene por defecto el md5, ejemplo:

Host all all 10.0.0.1/8 md5

Es decir que se podrán conectar todos los host desde cualquier trama de IP contenida en esa dirección IP a todas las bases de datos que existen en el servidor.

Anexo 2

2.8 Descripción de las tablas.

Nombre de la tabla: actividad		
Descripción: Almacena el comportamiento de las actividades que se realizan diariamente en los pozos de petróleo.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el identificador de la actividad, es tomado como llave primaria de esta tabla.
descripción	varchar	Almacena la descripción de cada actividad.
hora	integer	Almacena la cantidad de horas que en que duran las actividades.

Tabla 2. Tabla Actividad

Nombre de la tabla: n_actividad		
Descripción: Almacena los nombres de las actividades que se realizan en los pozos de petróleo. Estas actividades están definidas por los clientes, y son almacenadas en esta tabla como nomencladores.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el identificador de la actividad, es tomado como llave primaria de esta tabla.
nombre	varchar	Almacena el nombre de las actividades realizadas.

Tabla 3. Tabla nomenclador de actividades

Nombre de la tabla: usuario		
Descripción: Almacena los usuarios que interactúan con el sistema propuesto.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del usuario, es la llave primaria de esta tabla.
password	varchar	Almacena la contraseña del usuario.
id_pozo	integer	Almacena el id del pozo en el cual está trabajando el usuario. Puede ser un atributo nulo. Se necesita para saber si este usuario pertenece a un pozo.
nombre	varchar	Almacena el nombre completo del usuario.
Usuario	varchar	Almacena el usuario por el cual podrá acceder al sistema.

Tabla 4. Tabla Usuario

Nombre de la Tabla: rol		
Descripción: Almacena los roles de los usuarios que interactúan diariamente en los pozos de petróleos		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de esta tabla.
rol	varchar	Almacena el rol del usuario. Que es un identificador del rol.
permiso	varchar	Almacena los permisos que irá a tener un usuario determinado. Específicamente para determinar diferentes niveles de acceso al sistema.
nombre_rol	varchar	Almacena el nombre del rol.
lugar	varchar	Almacena el lugar donde se encuentran los usuarios, ya sea pozo, DIPP, CEINPET.

Tabla 5. Tabla Rol

Nombre de la Tabla: n_prod_quim		
Descripción: Almacena los productos químicos que se utilizan diariamente en los pozos de petróleos		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el identificador de los productos químicos, es tomado como llave primaria de esta tabla.
precio	double	Almacena el precio de los productos químicos.
peso	double	Almacena el peso que tienen los productos químicos.
nombre_producto	varchar	Almacena el nombre de los productos químicos.
cant	double	Almacena la cantidad de productos químicos a

		utilizar.

Tabla 6. Tabla n_prod_quim

Nombre de la Tabla: costo		
Descripción: Almacena los presupuestos gastados diariamente en los pozos.		
Atributo	**Tipo**	**Descripción**
presupuesto_MN	double	Almacena el presupuesto en moneda nacional.
presupuesto _MLC	double	Almacena el presupuesto en moneda libremente convertible.
fecha	Date	Almacena la fecha en que se realiza el costo. Actúa como llave primaria
pozo_id	integer(serial)	Almacena el id del pozo. Actúa como llave primaria.
n_operaciones_id	integer(serial)	Almacena el id de las operaciones. Actúa como llave primaria.

Tabla 7. Tabla Costo

Nombre: n_equipo		
Descripción: Almacena los equipos de perforación que se encuentran en los pozos de petróleo.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el identificador de los equipos de perforación.
país	varchar	Almacena el país al cual pertenecen los equipos.
nombre_equipo	varchar	Almacena el nombre de los equipos.
Material_id	integer(serial)	Almacena el id del material pesante utilizado en los pozos perteneciente al equipo.
Pozo_id	integer(serial)	Almacena el id del pozo al que pertenece el

		equipo.

Tabla 8. Tabla nomenclador de los equipos de perforación.

Nombre: nbarrenas		
Descripción: Almacena los distintos tipos de barrenas que se utilizan en la perforación de pozos de petroleros.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el identificador de esta tabla.
num_serie	varchar	Almacena el número de serie de las barrenas
iadc	integer	Almacena el código de dureza de la barrena.
fabricante	varchar	Almacena el fabricante de la barrena.
Nombre	varchar	Almacena el nombre de la barrena.
Tipo	varchar	Almacena el tipo de barrena.
diametro	double	Almacena el diámetro de la barrena.

Tabla 9. Tabla nomenclador de barrenas

Nombre: barrenas		
Descripción: Almacena el comportamiento y uso que se le da a las barrenas en un pozo de petróleo determinado.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de comportamiento de la barrena.
inicio	integer	Almacena el primer valor de profundidad a partir del cual se empezó a usar la barrena.
fin	integer	Almacena el valor final de profundidad a partir del cual se terminó de usar la barrena.
canthoras	integer	Almacena la cantidad de horas que se usó la barrena.

mts	double	Almacena la cantidad de metros que se perforó con la barrena.
vm	integer	Almacena la velocidad media de la barrena.
psb	double	Almacena la presión que se ejerce sobre la barrena.
rpm	double	Almacena la cantidad de revoluciones de la barrena.
caudal	double	Almacena el caudal de la barrena.
spm(stroke por min)	integer	Almacena la cantidad de golpes por min que recibe la barrena.
presion	double	Almacena la presión que ejerce la barrena.
peso	double	Almacena el peso de la barrena
jets	varchar	Componente de la barrena que se le añade a la hora de perforar, aquí se guarda el diámetro del mismo. (ej. 3X20)
bha	integer	Almacena el tamaño del orificio inferior de montaje de la barrena.
rop	double	Almacena el índice de penetración de la barrena.

Tabla 10. Tabla Barrenas

Nombre: cod_degaste		
Descripción: Almacena la información referente al código de desgaste de cada una de las barrenas utilizadas en un pozo determinado.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del código de desgaste de la barrena.
out_r	integer	Almacena el código en un rango del 1-8 el uso de los dientes exteriores (outer row) de la

		Barrena.
inr	integer	Almacena el código en un rango del 1-8 el uso de los dientes interiores (inner row) de la Barrena.
mdc	varchar	Almacena el valor de las principales causas de desecho de la barrena.
location	varchar	Almacena la localización del desgaste de la barrena.
bearing_condition	varchar	Almacena la condición de la barrena.
gauge_wear	double	Almacena la cantidad de uso de la barrena.
rp	varchar	Almacena la razón por la cual se desecha la barrena.
odc	varchar	Almacena otras características de desecho de la barrena.

Tabla 11. Tabla Código de Desgaste.

Nombre: pozo		
Descripción: Almacena la información referente a los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del pozo
yacimiento	varchar	Almacena el tipo de yacimiento que existe en el pozo.
fecha_inicio	Date	Almacena la fecha de cuando se empezó a perforar el pozo.
prof_final	double	Almacena la profundidad final del pozo.
duracion	int	Almacena la duración de la perforación del pozo.
prof_vertical	double	Almacena la profundidad vertical del pozo.

profund_real	double	Almacena la profundidad real del pozo.
desp	double	Almacena el desplazamiento de la perforación.
operador	varchar	Almacena el nombre del operador del pozo.
controladores	varchar	Almacena los controladores de los pozos.
pronostico	varchar	Almacena el pronóstico del pozo.
costopozocuc	double	Almacena el costo en CUC de los pozos.
costopozomn	double	Almacena el costo en MN de los pozos.

Tabla 12. Tabla Pozo

Nombre: direccional		
Descripción: Almacena la información de la dirección de la perforación en los pozos de petróleo.		
Atributo	**Tipo**	**Descripción**
id_d	integer(serial)	Almacena el id del direccional.
vert_azimuth	double	Almacena el azimuth por la vertical de la dirección.
cliente	varchar	Almacena el cliente para el que se está trabajando.
field	varchar	Almacena el campo donde se está trabajando.
direccional	varchar	Almacena el nombre de la persona que trabaja como direccional.
fecha_dir	date	Almacena la fecha emitida por el direccional.
tort	double	Almacena coordenadas de dirección.
ahd	double	Almacena la profundidad dada las coordenadas.
fecha_reporte	date	Almacena la fecha donde fue emitido el reporte.
longitud	double	Almacena la longitud de direccionamiento.
location_grid_e	double	Almacena las coordenadas por el este.

location_grid_n	double	Almacena las coordenadas por el norte.
grid_conv_angle	double	Almacena el ángulo de convergencia
grid_scale_fac	double	Almacena el factor de la escala de dirección.
survey	varchar	Almacena la forma aplicada para la perforación.
met_comp	varchar	Almacena el método empleado para la dirección.
vert_orig	double	Almacena el origen de la vertical del pozo.
tvd_ref	varchar	Almacena los datos de las coordenadas TVD
tvd_elev	varchar	Almacena las coordenadas TVD de elevación.
sea_bed	double	Almacena la posición del pozo sobre el nivel del mar.
fecha	Date	Almacena la fecha en que se emitió el informe.

Tabla 13. Tabla Direccional

Nombre: intervalo		
Descripción: Almacena los distintos intervalos reales que existen en los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del intervalo.
nombre_int	varchar	Almacena el nombre real del intervalo.
diametro	double	Almacena el diámetro real del intervalo.
fecha_inicio	Date	Almacena la fecha de inicio del intervalo.
fecha_fin	Date	Almacena la fecha de fin del intervalo.

Tabla 14. Tabla Intervalo

Nombre: inclinometria		
Descripción: Almacena la información de la inclinación que se va dando a las barrenas durante la perforación de los pozos de petróleo.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de la inclinometría.
depth	double	Almacena la profundidad.
descrip	varchar	Almacena una breve descripción hecha por el survey.
inclination	double	Almacena la inclinación del pozo.
azimuth	double	Almacena el azimuth del pozo.
tvd	double	Almacena las coordenadas de TVD del pozo.

Tabla 15. Tabla Inclinometría

Nombre: fluido_perf		
Descripción: Almacena información acerca de los fluidos de perforación utilizados en los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del fluido de perforación.
prof_muestra	integer	Almacena una muestra de lodo a cierta profundidad.
gel	varchar	Almacena el tipo de gel utilizado en el lodo.
fecha	date	Almacena la fecha en la cual se analiza la muestra de lodo.
mw	integer	Almacena el peso del lodo (Mud Weight).
fv	integer	Almacena la viscosidad del embudo.
ph	double	Almacena la acidez del lodo.
sand	double	Almacena la cantidad de arena en el lodo.
solids	double	Almacena la cantidad de sólidos que llevan los productos químicos

cl (cloro)	double	Almacena la cantidad de cloro aplicado al lodo.
k (potasio)	double	Almacena la cantidad de potasio aplicado al lodo.
ca (calcio)	Integer	Almacena la cantidad de calcio aplicado al lodo.
fann3	Integer	Almacena la cantidad de fann3 (tipo de fluido de perforación usado en la perforación).
fann6	Integer	Almacena la cantidad de fann6 (tipo de fluido de perforación de la familia del fann3 usado en la perforación).
pv	float	Almacena la viscosidad plástica.
ph	double	Almacena la acidez del fluido de perforación.
perdidas_sup	double	Almacena la cantidad de fluido de perforación que se pierde en la superficie.
perdidas_form	double	Almacena la cantidad de fluido de perforación que se pierde en las formaciones geológicas.
horas	time	Almacena la cantidad de horas de uso.
mudcleaner	double	Almacena la cantidad de fluidos limpiadores de lodo que se utiliza.
yp	double	Almacena el punto de excedente (yield point).
apifiltrate	double	Almacena el filtrado del lodo.
cake	double	Almacena el valor del revoque del lodo.

Tabla 16. Tabla Fluido de Perforación

Nombre: metrajediario		
Descripción: Almacena el metraje diario de perforación de los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del metraje diario.

fecha	Date	Almacena la fecha de perforación.
perf	double	Almacena la cantidad de metros que se perforan.

Tabla 17. Tabla Metraje Diario

Nombre: n_unidad_medida		
Descripción: Almacena las unidades de medidas		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de las unidades de medida.
unidad	varchar	Almacena el nombre de las unidades de medidas

Tabla 18. Tabla Unidad de Medida

Nombre: cronograma		
Descripción: Almacena los datos del cronograma de perforación de los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del cronograma.
Pozoid	integer(serial)	Almacena el id del pozo.
metraje_perf_plan	int	Almacena la cantidad de metros que se irán perforando diariamente.
inicio_interv_plan	date	Almacena el inicio del intervalo según lo planificado.
fin_intev_plan	date	Almacena el fin del intervalo según lo planificado.
cant_dias	int	Almacena la cantidad de días que se perforarán en los pozos.

Tabla 19. Tabla Cronograma

Nombre: tiempo_perdido		
Descripción: Almacena los datos de los tiempos perdidos durante la perforación de los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del tiempo perdido.
descrip	varchar	Almacena una descripción con la razón por la cual se perdió tiempo durante la perforación.
horas	time	Almacena la cantidad de horas perdidas.
responsable	varchar	Almacena el nombre de la persona que fue responsable de la pérdida.
Pozo_id	serial	Almacena el id del pozo.

Tabla 20. Tabla Tiempo Perdido

Nombre: herramientas		
Descripción: Almacena los datos de las herramientas utilizadas durante la perforación de los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de las herramientas.
longitud	double	Almacena la longitud de las herramientas
ind	int	Almacena el diámetro interior de la herramienta.
outd	int	Almacena el diámetro exterior de la herramienta.
tipo	int	Almacena el tipo de herramienta.

Tabla 21. Tabla Herramientas

Nombre: necesidad		
Descripción: Almacena los datos de las necesidades que surgen durante la perforación de los pozos.		
Atributo	**Tipo**	**Descripción**

id	integer(serial)	Almacena el id de la necesidad.
nombre	varchar	Almacena el nombre de las necesidades existentes.
cantidad	integer	Almacena la cantidad de necesidades.
lugar_orig	varchar	Almacena el lugar de origen de la necesidad.
lugar_dest	varchar	Almacena el lugar de destino de la necesidad.
fecha	date	Almacena la fecha de la necesidad.
responsable	varchar	Almacena el nombre del responsable el cual cumplirá con las necesidades.

Tabla 22. Tabla Necesidad

Nombre: inform_geologica		
Descripción: Almacena los datos de la información geológica de los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de la información de geología.
prof_md	double	Almacena el valor de la profundidad donde se encuentra la perforación.
prof_tvd	double	Almacena el valor de la profundidad por el tvd, es decir la profundidad que se va obteniendo con respecto a la vertical.
litologia	varchar	Almacena una serie de características de la información litológica encontradas según las distintas formaciones existentes durante la perforación de los pozos.

Tabla 23. Tabla Información Geológica

Nombre: pozo_interv		
Descripción: Almacena los datos de los pozos que están en intervención.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de los pozos en intervención.
tipo_trab	varchar	Almacena el tipo de trabajo que se realizará.
operac	varchar	Almacena las operaciones que se realizan en los pozos en intervención.
equipo_interv	time	Almacena el nombre del equipo que se encuentra en perforación.
paila	time	Almacena la hora que llega la paila(Herramienta usada en los pozos)
cement	time	Almacena la hora que llega la herramienta de cementación.
rastra	time	Almacena la hora que llega la rastra.
rep_incid	varchar	Almacena una descripción de los reportes de incidencias.
rep_dific	varchar	Almacena una descripción de los reportes de las dificultades.
rep_inconform	varchar	Almacena una descripción de los reportes de las inconformidades.
evaluacion	varchar	Almacena la evaluación que se realiza.
pronostic	varchar	Almacena el pronóstico que se realiza.

Tabla 24. Tabla Pozo Intervención

Nombre: combustible		
Descripción: Almacena los datos de combustible utilizados en los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id del combustible.

tipo_comb	varchar	Almacena el tipo de combustible a utilizar en los pozos.
capac_max	double	Almacena la capacidad máxima que tienen los tanques
Días_cobert	integer	Almacena los días con que se cubrirán la cantidad de combustible asignada.
solicitud	varchar	Almacena las solicitudes que se realizan.
cant_diaria	double	Almacena la cantidad diaria de combustible que se usa.
invent	double	Almacena la cantidad de combustible que se encuentra en inventario.

Tabla 25. Tabla Combustible

Nombre: logs		
Descripción: Almacena los distintos registros con las acciones que realizo determinado usuario.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de los logs.
hora	time	Almacena la hora que se realizó el registro.
operacion	varchar	Almacena la operación que se realizó.
fecha	date	Almacena la fecha en que se realizó el registro.
usuario_id	Serial	Almacena el id del usuario que realizó determinada operación.

Tabla 26. Tabla Logs

Nombre: dist_tiempo		
Descripción: Almacena los datos de la distribución del tiempo de las actividades.		
Atributo	**Tipo**	**Descripción**

id	integer(serial)	Almacena el id de la distribución de tiempo de las actividades.
día	integer	Almacena el número del día en que se realiza la actividad correspondiente al día de perforación de los pozos.

Tabla 27. Tabla Distribución del Tiempo

Nombre: cant_prod		
Descripción: Almacena los datos de la cantidad de productos químicos que se compran en un día determinado.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el id de la compra de los productos químicos.
fecha	date	Almacena la fecha de compra de los productos.
cant	integer	Almacena la cantidad de productos que se compran
n_prod_quim	integer	Almacena el id del producto químico que se compró.

Tabla 28 Tabla Cantidad de Productos Químicos

Nombre de la tabla: n_operacion		
Descripción: Almacena los nombres de las operaciones que se realizan en los pozos de petróleo. Estas operaciones están definidas por los clientes, y son almacenadas en esta tabla como nomencladores.		
Atributo	**Tipo**	**Descripción**

id	integer(serial)	Almacena el identificador de las operaciones, es tomado como llave primaria de esta tabla.
nombre	varchar	Almacena el nombre de las operaciones realizadas.
Cod	integer	Almacena el código de las operaciones este número es predefinido por los clientes

Tabla 29 Tabla N_Operaciones

Nombre de la tabla: comp_herram		
Descripción: Almacena la composición de las herramientas que se utilizan en los pozos de petróleo.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el identificador de la tabla, es tomado como llave primaria de esta tabla.
acumulado	double	Almacena el acumulado del tamaño de las herramientas usadas.
descripcion	varchar	Almacena una descripción de las composiciones de las herramientas.
horas	integer	Almacena la cantidad de horas que se usa.

Tabla 30. comp_herram

Nombre de la tabla: material		
Descripción: Almacena el material que se utilizan en los pozos.		
Atributo	**Tipo**	**Descripción**
id	integer(serial)	Almacena el identificador de la tabla, es tomado como llave primaria de esta tabla.

Lugar	varchar	Almacena el lugar de donde se necesita el material
invent	double	Almacena la cantidad en inventario del material.
cons_prom	double	Almacena el consumo promedio del material.
dias_cobert	integer	Almacena los días de cobertura de un material.
solic	varchar	Almacena la solicitud que se realiza.

Tabla 31 Tabla Material

Anexo 3

Diagrama de Clases Persistentes.

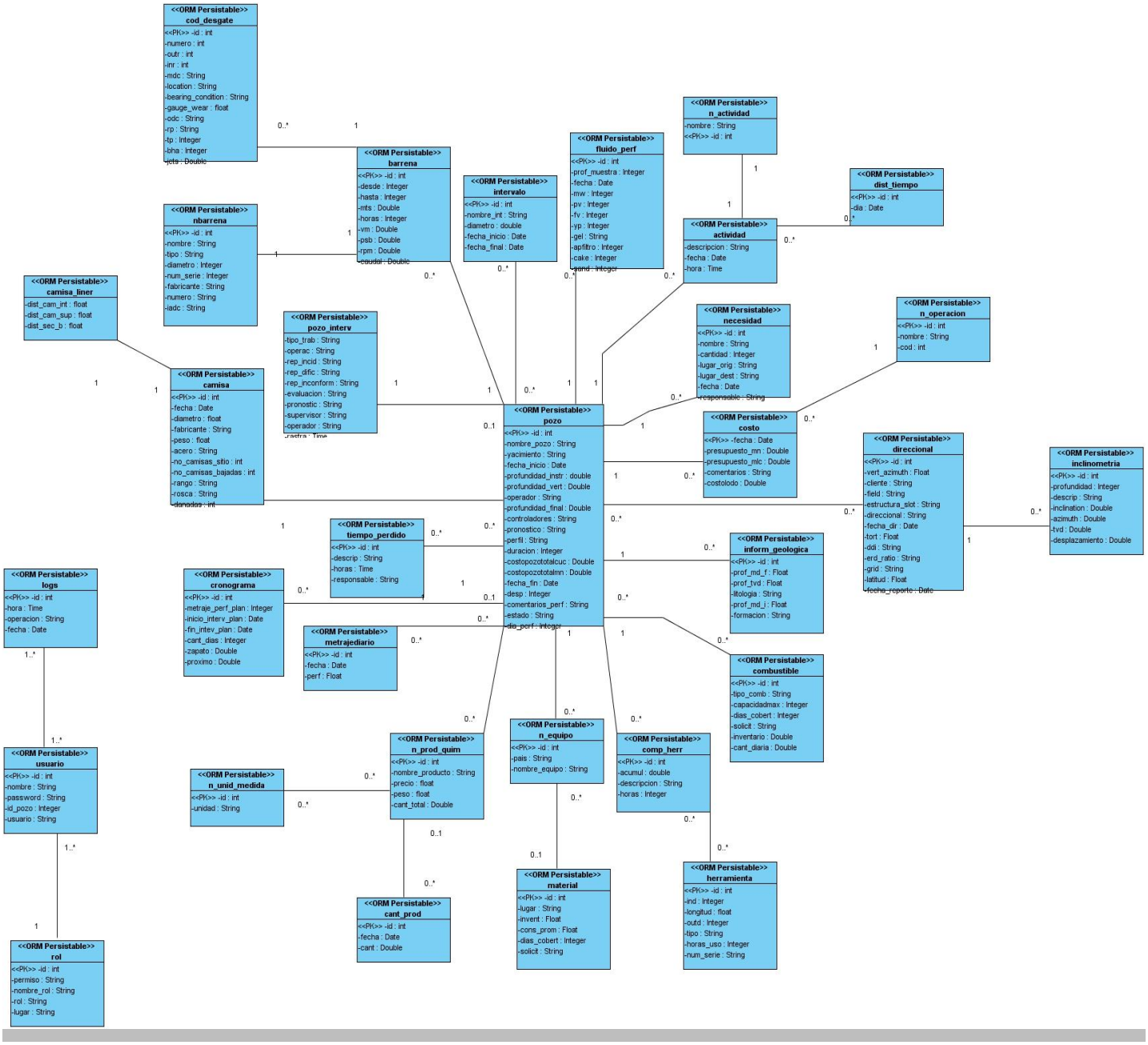

Figura 10 Diagrama de Clases Persistentes.