UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

FACULTAD 6

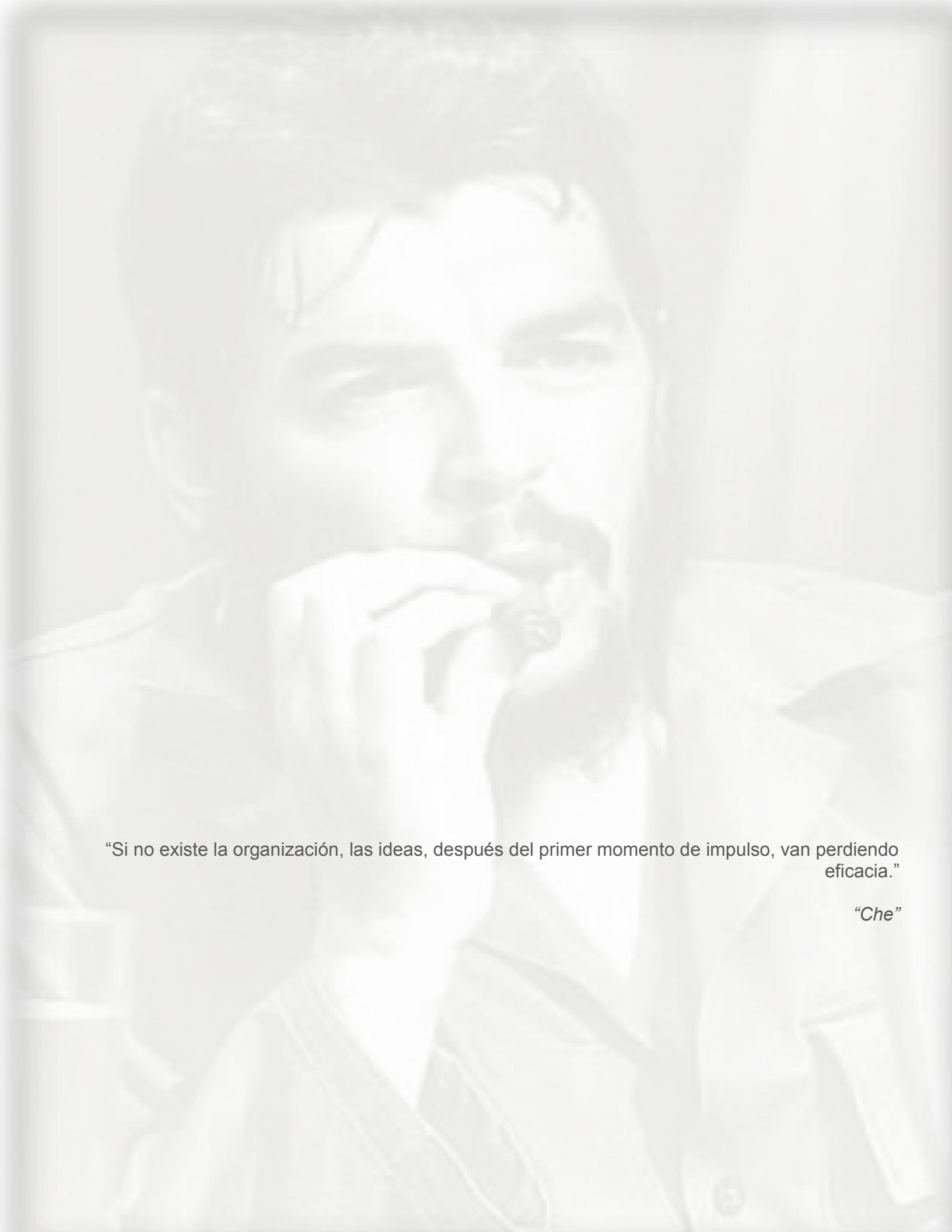
Módulo Editor de Consultas a la Base de Datos para el Sistema de Gestión de Datos Geológicos

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autor: Eddy Ernesto Enrique Hernández

Tutor: Ing. Joel Macías Roque

La Habana
"Año 55 de la Revolución"



“Si no existe la organización, las ideas, después del primer momento de impulso, van perdiendo eficacia.”

“Che”

Declaración de Autoría

Declaro ser autor de la presente tesis y reconozco 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 _____.

Eddy Ernesto Enrique Hernández

_____
Firma del Autor

Ing. Joel Macías Roque

_____
Firma del Tutor

Datos de contacto

Datos del Tutor:

Nombre y apellidos: Ing. Joel Macías Roque

Correo electrónico: jmroque@uci.cu

Año de graduación: 2010

Profesión: Ingeniero en Ciencias Informáticas.

Breve descripción: Graduado en la Universidad de las Ciencias Informáticas. Actualmente Jefe de desarrollo del proyecto Sistema de Gestión de Datos Geológicos del Departamento Geoinformática del Centro de Desarrollo “Geoinformática y Señales Digitales” de la facultad 6.

Agradecimientos

Hay momentos en la vida, que son especiales por si solos, compartirlos con las personas que quieres, los convierte en inolvidables. Cuando la gratitud es tan absoluta las palabras sobran y precisamente en este momento no tengo palabras suficientes para agradecer a tantas personas que han tenido que ver con este momento tan especial en mi vida.

Quiero agradecer muy en especial a la Revolución y al Comandante en Jefe Fidel Castro Ruz, por darme la posibilidad de pertenecer a este hermoso proyecto que es la UCI.

A mis queridos padres, por ser las personas más importantes y especiales de mi vida, por estar siempre a mi lado, por su entrega incondicional en mi educación y formación, por ser los motores impulsores de mis sueños, por ser mis guías, mis amigos, por darme las fuerzas y el amor que me han dirigido por la vida.

A mi abuelita linda, que sin sus consejos y su educación no habría podido convertirme en la persona que soy hoy día.

A mi querido tío Luis, por representar un ejemplo a seguir, por su cariño y amor, porque sin él hubiera sido imposible realizar este sueño.

A mis hermanos, por estar siempre ahí cuando los necesité, por transmitirme tanto amor y por representar un ejemplo para mí.

A mi familia en general, por el gran apoyo que siempre me han brindado, por el cariño que me profesan y por estar siempre pendientes de mí.

A mi novia y amiga Maydelín, por su ayuda, comprensión y cariño en todo momento. Por estar siempre a mi lado cuando más lo necesito. A su familia por aceptarme, por su preocupación y cariño.

A mi tutor Joel, por su ayuda, dedicación y apoyo incondicional para la realización de este trabajo.

A todos los profesores que han contribuido con mi formación profesional.

A todos mis amigos y compañeros de grupo.

Dedicatoria

Hay momentos en la vida difíciles, la vida te pone obstáculos los cuales uno tiene que pasar y seguir adelante con más fuerza. Durante los años de universidad el destino me arrebató a mi papá y a mi abuelita y no quiso que ellos estuvieran conmigo para celebrar este triunfo, pero yo se que ellos me están mirando y están muy orgullosos, decirles que ellos han sido la principal fuente de inspiración para que yo me convirtiera en ingeniero y que a ellos está dedicado en especial mi tesis.

Le dedico la tesis con el mayor amor que puede entregar una persona a otra, a mi mamá, el mayor tesoro que me ha dado la vida, que con mucho sacrificio y amor se ha entregado a mí y me ha educado para que yo sea un hombre de bien.

A mi tío, que más que tío ha sido un padre para mí, es una de las personas más nobles que he conocido y lo amo.

A mis hermanos, que han sido vitales en mi vida para seguir cada día adelante.

Resumen

La Oficina Nacional de Recursos Minerales es en Cuba la encargada de llevar el control estatal de las actividades de minería, geología y petróleo, para ello gestiona gran cantidad de información, la cual posee más de cien años de antigüedad. El surgimiento de las Tecnologías de la Información y las Comunicaciones posibilitaron que se digitalizaran la mayoría de los procesos de la mencionada entidad, esto sucedió con la creación del Sistema de Gestión de Datos Geológicos. Sin embargo todavía existen problemas en cuanto al acceso a la información contenida en la base de datos. La presente investigación, que lleva por título “Módulo Editor de Consultas a la Base de Datos para el Sistema de Gestión de Datos Geológicos”, tiene como objetivo principal el desarrollo de una aplicación web, que pueda ser integrada al SGDG y que mejore el acceso a la información digital que posee la Oficina Nacional de Recursos Minerales. Los flujos de trabajo que define la metodología Proceso Unificado de Desarrollo son utilizados como guía para la construcción del sistema. Como resultado de la investigación se obtiene la documentación técnica asociada al editor de consultas y una herramienta que podrá ser utilizada por los especialistas de la ONRM para construir consultas de forma fácil y obtener cualquier información de la base de datos.

Palabras clave

Aplicación Web, Editor de Consultas, Oficina Nacional de Recursos Minerales, Sistema de Gestión de Datos Geológicos.

ÍNDICE DE CONTENIDO

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE TABLAS

INTRODUCCIÓN

Los minerales son tan antiguos como el ser humano. Las primeras nociones sobre los minerales se formaron en la antigüedad. El conocimiento del mundo mineral comenzó por el uso directo de las piedras recogidas en la superficie de la tierra como instrumentos de trabajos simples y armas, el desarrollo del hombre está estrechamente relacionado con la forma de explotación y uso que ha sabido darle a los recursos minerales. Están presentes en las casas en las cuales viven las personas, en los medios de transporte, en los objetos tecnológicos, en los medicamentos e incluso en los alimentos. Debido a su gran utilidad hoy día resulta indispensable para un país tener el control de los mismos debido a que el mundo gira alrededor de ellos por sus múltiples aplicaciones en los diversos campos de la actividad humana.

Los minerales son recursos naturales no renovables, por ello se hace necesaria una explotación controlada de sus yacimientos. Para poder garantizar un uso racional, debido a que muchos de ellos tienen vida limitada es necesario que existan instituciones capaces de controlar el proceso de explotación. Es por ello que en Cuba se crea en el año 1995, con la promulgación de la Ley de Minas (Alarcón De Quesada, 1995), la Oficina Nacional de Recursos Minerales (en lo adelante ONRM), la cual se subordina al Ministerio de la Industria Básica (MINBAS) y actúa como entidad rectora de los procesos de minería que se realizan en Cuba. La oficina cuenta en sus archivos con la información de la actividad geológica generada en Cuba desde hace más de 100 años por lo que posee un alto valor docente, económico, histórico y científico.

La ONRM presenta entre sus funciones principales:

- ✓ Velar por el aprovechamiento racional de los recursos minerales del país.
- ✓ Ejercer con eficiencia, rigor técnico y responsabilidad el control estatal sobre las actividades de geología, minería y petróleo.

Las Tecnologías de la Información y la Comunicaciones (TIC), se encargan del estudio, desarrollo, implementación, almacenamiento y distribución de la información mediante la utilización de hardware y software. Las TIC convierten la información, tradicionalmente sujeta a un medio físico, en inmaterial. Mediante la digitalización es posible almacenar gran cantidad de información. A su vez los usuarios pueden acceder a información ubicada en dispositivos electrónicos lejanos, que se transmite utilizando las redes de comunicación, de una forma transparente e inmaterial. (Rosario, 2005)

Cuba ha identificado desde muy temprano la conveniencia y necesidad de dominar e introducir en la práctica social las TIC y lograr una cultura digital como una de las características imprescindibles del hombre nuevo, lo que facilitaría a la sociedad acercarse más hacia el objetivo de un desarrollo sostenible.

A partir de la intención nacional de informatizar la sociedad, comenzó un proceso gradual con este objetivo en la mencionada entidad cubana. Sin embargo todavía existen algunas limitaciones e ineficiencias en los procesos de trabajo que se llevan a cabo en la entidad. En ocasiones es necesario consultar la información almacenada para satisfacer la necesidad de determinado cliente, incluso información que en la actualidad no está pensada, lo cual resulta engorroso para los especialistas de la ONRM debido a que la forma fundamental de acceder a la información digital es a través de consultas predefinidas, trayendo como consecuencia la imposibilidad de obtener una información específica de manera rápida, influyendo de forma negativa en la capacidad de respuesta de la oficina.

El proceso de búsqueda de información requiere de gran cantidad de tiempo debido a que no se puede acceder de manera fácil y rápida a toda la información, lo que trae como consecuencia que el trabajo por parte de los especialistas no pueda desarrollarse de manera dinámica. Además si un cliente solicita una determinada información y esta no puede gestionarse de manera digital los reportes habría que realizarlo manualmente por lo que estos podrían contener errores.

A partir de la problemática existente se define para esta investigación el siguiente **problema a resolver**: Las deficiencias en el acceso a la información digitalizada que se consulta en la ONRM afectan la capacidad de respuesta de la mencionada entidad. Como consecuencia se determina que el **objeto de estudio** es: los mecanismos para el acceso y la recuperación de información en bases de datos.

Debido a que la digitalización de la información se realizó mediante la creación de un sistema nombrado Sistema de Gestión de Datos Geológicos (SGDG), se propone integrar el editor de consultas que se pretende a dicha aplicación, para ello se plantea como **objetivo general**: Desarrollar el módulo editor de consultas a la base de datos para el Sistema de Gestión de Datos Geológicos, siendo el **campo de acción**: los mecanismos utilizados para la recuperación de la información de la base de datos del Sistema de Gestión de Datos Geológicos en la ONRM.

Para dar cumplimiento al objetivo general se proponen las siguientes **tareas de la investigación**:

1. Caracterizar el estado del arte sobre editores de consultas a base de datos.

2. Caracterizar la arquitectura base del Sistema de Gestión de Datos Geológicos.
3. Elaborar la documentación técnica correspondiente al módulo editor de consultas a la base de datos.
4. Implementar el módulo editor de consultas a la base de datos según la documentación técnica generada.
5. Realizar las pruebas al módulo editor de consultas a la base de datos.

Luego de realizadas las tareas de la investigación se esperan como **posibles resultados**:

1. La documentación técnica relacionada con el desarrollo del módulo editor de consultas a la base de datos.
2. Una herramienta que será utilizada en la Oficina Nacional de Recursos Minerales para la consulta de la información que se manipula y transfiere en dicha entidad, desarrollada completamente con tecnologías libres.

Durante el período de investigación se utilizan un conjunto de métodos científicos:

Métodos Teóricos:

Analítico-sintético: Se aplica en la investigación para realizar el análisis de las diferentes bibliografías y poder realizar la fundamentación teórica del Editor de Consultas, así como realizar una síntesis de los conceptos principales relacionados al dominio del problema.

Análisis Histórico-lógico: Se observa cuando se realiza el estudio de investigaciones anteriores sobre los procesos de gestión de información, publicaciones y trabajos realizados sobre editores de consultas de base de datos, para a partir de esto, fomentar el desarrollo de la presente investigación.

Métodos Empíricos:

Observación: Se utiliza para entender cómo se realiza el trabajo en la ONRM y cómo funciona el sistema. La técnica aplicada fue la observación externa no incluida que es donde el observador realiza la investigación desde fuera del grupo que se estudia.

La investigación se encuentra dividida en cuatro capítulos:

Capítulo 1. "**Fundamentación teórica**": En este capítulo se enuncian los conceptos relacionados a la construcción del editor de consultas y a la información que se va a manejar en el mismo. Se aborda el

objeto de estudio así como se plantea de manera detallada los principales problemas a los que se enfrenta hoy la ONRM en cuanto al acceso a la información contenida en la base de datos y el por qué es necesario un editor de consultas. Se realiza el análisis de soluciones existentes a nivel internacional con el objetivo de buscar una forma de dar solución al problema.

Capítulo 2. "Metodología, Tecnologías y Herramientas": En este capítulo se define la metodología de desarrollo de software a utilizar así como las tecnologías, herramientas y las ventajas que estas brindan.

Capítulo 3. "Características del Sistema": Se describe la solución propuesta para luego realizar la implementación correspondiente, se analiza tanto el negocio como el sistema dejando reflejados los actores, casos de uso, las descripciones de los requisitos funcionales y no funcionales y se modelan los diagramas correspondientes.

Capítulo 4. "Construcción y validación de la solución propuesta": En este capítulo se plantea la construcción de la solución propuesta a través de los flujos de trabajo Análisis y Diseño, Implementación y Prueba, generando los artefactos correspondientes.

CAPÍTULO 1: Fundamentación teórica

1.1 Conceptos generales de la investigación

1.1.1 Información Geológica

El término información según el diccionario de la Real Academia de la Lengua Española es definido como: "Comunicación o adquisición de conocimientos que permiten ampliar o precisar los que se posee sobre una materia determinada." (RAE, 2012)

La Real Academia de la Lengua Española define la geología como: "ciencia que trata de la forma exterior e interior del globo terrestre, de la naturaleza de las materias que lo componen y de su formación, de los cambios o alteraciones que estas han experimentado desde su origen y de la colocación que tienen en su actual estado". (RAE, 2012)

El concepto de geología tiene su origen en dos vocablos griegos: "*geo*", que significa tierra y "*logos*" que significa estudio. La geología es considerada una ciencia sumamente compleja debido a que relaciona una gran cantidad de disciplinas, cuyo único objetivo es: lograr una visión unitaria del planeta tierra. (DefiniciónDe, 2008)

Una vez definidos los conceptos de información y geología, el autor de la presente investigación considera que información geológica es un conjunto de datos que representan los conocimientos adquiridos mediante el estudio de materias relacionadas con la geología.

1.1.2 Editor de Consultas

Editor, del latín "*ed*ĭ*tor*", es aquel o aquello que edita. El verbo editar, por su parte, refiere a publicar una obra a través de algún soporte o a corregir y adaptar una obra de acuerdo a ciertas reglas y normas. En el ámbito de la informática, un editor es un programa (software) que permite corregir, crear o almacenar algún tipo de archivo. (DefiniciónDe, 2008)

La Real Academia de la lengua Española puntualiza que editor es un programa que permite redactar, corregir o archivar textos registrados en ficheros de símbolos (RAE, 2012). El Diccionario de la Lengua Española precisa que el término consulta significa: "Parecer o dictamen que por escrito o de palabra se pide o se da acerca de algo" (RAE, 2012). Una consulta es un conjunto de instrucciones en un lenguaje específico de manipulación de datos con el que se pueden seleccionar, ordenar, filtrar, mostrar, modificar,

añadir y borrar datos de una base de datos (Ibaceta, 2012).

Las consultas SQL se dividen en tres elementos sencillos que especifican una acción, el nombre de una tabla de la base de datos y un conjunto de parámetros. Suele comenzar con una palabra clave de acción, o comando que especifica la operación que quiere realizar.” (Parsons, 2008)

El autor considera, teniendo en cuenta los conceptos antes expuestos que un editor de consultas no es más que un programa que permite efectuar consultas con el fin de recuperar, corregir y modificar información en bases de datos.

1.1.3 Módulo

Es un fragmento de un programa que se desarrolla de forma independiente del resto del programa. Esta independencia hace posible un mecanismo de compilación por separado que limita la complejidad del programa que se está desarrollando. (López, y otros, 2010) En el caso específico del SGDG los módulos son una forma de estructurar la aplicación, de forma tal que esté organizada. También pudiera llamarse a los módulos subsistemas.

1.1.4 Sistema de Gestión de Datos Geológicos

Aplicación informática desarrollada en el Centro de Desarrollo de Software GEySED, perteneciente a la facultad 6 de la Universidad de las Ciencias Informáticas. El objetivo de esta aplicación web es informatizar los procesos que tienen lugar en las distintas áreas de la Oficina Nacional de Recursos Minerales.

1.2 Base de Datos

A continuación se citan definiciones expuestas por diferentes autores sobre el concepto de base de datos:

“Conjunto de datos interrelacionados entre sí, almacenados con carácter más o menos permanente en la computadora. O sea, que puede considerarse una colección de datos variables en el tiempo” (Matos, 2001).

Una base de datos es como una especie de armario electrónico para archivar, es decir, es un depósito o contenedor de una colección de archivos de datos computarizados. Los usuarios del sistema pueden realizar una variedad de operaciones sobre dichos archivos, por ejemplo:

- ✓ Agregar nuevos archivos vacíos a la base de datos.
- ✓ Insertar datos dentro de los archivos existentes.
- ✓ Recuperar datos de los archivos existentes.
- ✓ Modificar datos en archivos existentes.
- ✓ Eliminar datos en archivos existentes.
- ✓ Eliminar archivos existentes de la base de datos. (Date, 2003)

Una base de datos es una colección o depósito de datos integrados, almacenados en soporte secundarios (no volátil) y con redundancia controlada. Los datos, que han de ser compartidos por diferentes usuarios y aplicaciones, deben mantenerse independientes de ellos, y su definición –estructura de la base de datos- única y almacenada junto con los datos, se ha de apoyar en un modelo de datos, el cual ha de permitir captar las interrelaciones y restricciones existentes en el mundo real. Los procedimientos de actualización y recuperación, comunes y bien determinados, facilitarán la seguridad del conjunto de los datos. (Castaño, 1999)

Estudiados los conceptos expuestos por las diferentes bibliografías se concluye que una base de datos es un conjunto de datos almacenados que se relacionan entre sí, además posibilita guardar información por un largo período de tiempo, permitiendo el acceso de manera directa a dicha información y posibilitando que se le puedan realizar operaciones. La utilización de base de datos evita la redundancia e inconsistencia de la información.

1.2.1 Usuarios y Grupos de Usuarios en base de datos

Los usuarios son los actores de la base de datos. Cada vez que se hace una operación sobre la base de datos se realiza a través de un usuario determinado. El Sistema Gestor de Base de Datos debe permitir la operación dependiendo del usuario que la está efectuando. Los usuarios son responsables de todas las actividades llevadas a cabo con su código de identificación de usuario y sus claves personales. (Onorat, 2009)

Grupos de Usuarios o Roles son aquellos a los cuales se le otorgan una serie de privilegios. Luego cuando se crea un usuario este puede hacerse miembro de un Grupo de Usuarios existente, y tiene los mismos privilegios que él, o sea, hereda los privilegios del Grupo de Usuarios al que pertenece.

Tipos de Usuarios de la base de datos

Usuarios normales: Son usuarios no sofisticados que interactúan con el sistema mediante un programa de aplicación con una interfaz de formularios, donde puede rellenar los campos apropiados del formulario. Estos usuarios pueden también simplemente leer informes generados de la base de datos.

Programadores de aplicaciones: Son profesionales informáticos que escriben los programas de aplicación, utilizando herramientas para desarrollar interfaces de usuario, como las herramientas de desarrollo rápido de aplicaciones, que facilitan crear los formularios e informes sin escribir directamente el programa.

Usuarios sofisticados: Interactúan con el sistema sin programas escritos, usando el lenguaje de consulta de base de datos para hacer sus consultas. Los analistas que envían las consultas para explorar los datos en la base de datos entran en esta categoría, usando ellos las herramientas de procesamiento analítico en línea (OLAP, *OnLine Analytical Processing*), o herramientas de recopilación de datos.

Usuarios especializados: Son usuarios sofisticados que escriben aplicaciones de bases de datos especializadas y adecuadas para el procesamiento de datos tradicional. Entre estas aplicaciones están los sistemas de diseño asistido por computadora, sistemas de base de conocimientos y sistemas expertos, sistemas que almacenan datos de tipos de datos complejos (como gráficos y de audio) y sistemas de modelado de entorno.

Administradores de la base de datos (ABD): Son las personas que tienen el control central del SGBD. Entre las funciones del ABD se encuentran:

- ✓ Definición del esquema de la base de datos.
- ✓ Definición de la estructura y el método de acceso.
- ✓ Modificación del esquema y la organización física.
- ✓ Concesión de autorización para el acceso a los datos.
- ✓ Mantenimiento rutinario. (Onorat, 2009)

1.2.2 Objetos y privilegios en base de datos

Los objetos de la base de datos son los elementos a los cuales se puede aplicar la protección. La seguridad se aplica generalmente a tablas y vistas.

Los privilegios son las acciones que un usuario tiene permitido efectuar o no sobre los objetos de una base de datos. Las acciones más comunes que realiza un usuario sobre un objeto son las de SELECT, INSERT, DELETE y UPDATE.

1.2.3 Conceptos asociados a base de datos

Entidad

El término entidad es empleado comúnmente en los círculos de bases de datos para referirse a cualquier objeto distinguible que va a ser representado en la base de datos. (Date, 2003) Además se define que son objetos concretos o abstractos que presentan interés para el sistema y sobre los que se recoge información que será representada en un sistema de bases de datos. (Riquelme, y otros, 2010)

Interrelaciones

Una interrelación es en sí misma un conjunto de objetos consistente de pares de instancias tomadas de dos conjuntos de objetos que se relacionan. Son asociaciones entre entidades, de las cuales existen distintos tipos, uno a uno, uno a muchos y muchos a muchos. (Hansen, 2000)

Dos relaciones están interrelacionadas cuando una posee una clave foránea de la otra. Cada una de las llaves foráneas de una relación establece una interrelación con la relación donde esa llave es la principal. Según esto, existen dos tipos de interrelación: la interrelación entre entidades fuertes y débiles y la interrelación pura, entre entidades fuertes. (Coronado, 2005)

Atributo

Es la unidad menor de información sobre un objeto almacenada en la base de datos y representa una propiedad de un objeto. (García, 1999) Como otra definición se plantea que es una unidad básica e indivisible de información acerca de una entidad o una relación. (Riquelme, y otros, 2010)

Llave

Es un atributo o conjunto de atributos de un artículo que define que cada ocurrencia de artículo de la base de datos sea único. En principio, cada artículo tiene una llave, ya que se tiene como hipótesis que cada elemento u ocurrencia del artículo es diferente de las demás. (García, 1999)

En una tabla relacional a veces es necesario poder determinar una tupla concreta, lo cual es posible mediante la llave. Se debe elegir la llave entre los atributos, de forma que no puedan existir valores

duplicados (la llave puede contener uno o más atributos). Hay varios tipos: primaria (la llave principal), ajena o foránea (la que corresponde a una primaria de otra tabla). (Riquelme, y otros, 2010)

1.3 Tipos de base de datos

Las bases de datos se pueden clasificar de acuerdo a la variabilidad de los datos y al contenido que se encuentra almacenado en estas:

La clasificación según la variabilidad depende de cómo perdura la información dentro de la base de datos: si se mantienen sin sufrir modificaciones o si los datos cambian. Conceptualmente se catalogan como: (Barajas, 2009)

Bases de Datos Estáticas: Son base de datos de solo lectura, utilizadas primordialmente para almacenar datos históricos que posteriormente se pueden utilizar para estudiar el comportamiento de un conjunto de datos a través del tiempo, realizar proyecciones y tomar decisiones.

Bases de Datos Dinámicas: Son bases de datos donde la información almacenada se modifica con el tiempo, permitiendo operaciones como actualización y adición de datos, además de las operaciones fundamentales de consulta.

Según el contenido la clasificación puede ser catalogada como directorios, base de datos bibliográficas y base de datos de texto completo.

Directorios: Los directorios y guías telefónicas en formato electrónico.

Base de datos bibliográficas: Solo contienen un representante de la fuente primaria, por ejemplo: una colección determinada de análisis o análisis de alguna materia.

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.

1.4 Modelos de Base de Datos

Un modelo de datos es básicamente una descripción de lo que se conoce como contenedor de datos –donde se guarda la información–, así como de los métodos para almacenar y recuperar información de esos contenedores. (Date, 2003)

Existen varios modelos que con frecuencia son utilizados para el diseño de las bases de datos, entre ellos el Modelo Jerárquico, el Modelo de Red, el Modelo Orientado a Objetos y el Modelo Relacional.

1.4.1 Modelo Jerárquico

Una base de datos jerárquica consiste en una colección de registros que se conectan entre sí por medio de ligas. Los registros y las ligas son similares a los del modelo de red, pero en el modelo jerárquico se organiza en forma de árbol con raíz (donde la raíz es nodo ficticio); de tal manera que una base de datos jerárquica es una colección de árboles de este tipo, formando un bosque. A cada árbol con raíz se le denomina árbol de base de datos. En este modelo un registro puede repetirse en varios sitios pudiendo ocasionar los siguientes problemas:

- ✓ Riesgos de la inconsistencia al llevar a cabo actualizaciones.
- ✓ Inevitable desperdicio de espacio en el medio de almacenamiento secundario. (Rob, 2004)

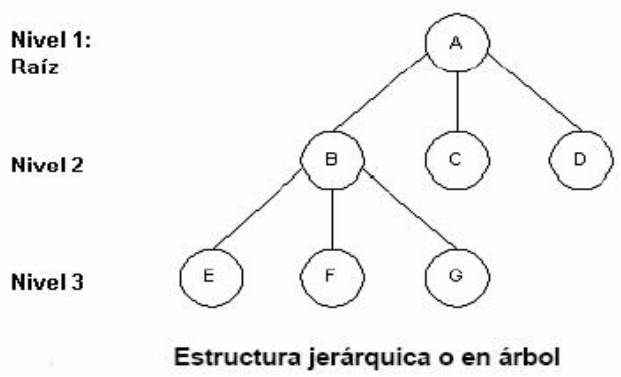


Ilustración 1: Estructura jerárquica o en árbol

1.4.2 Modelo de Red

Es un modelo ligeramente distinto del jerárquico. Su diferencia fundamental es la modificación del concepto de un nodo, permitiendo que un mismo nodo tenga varios padres (algo no permitido en el modelo jerárquico). (Hansen, 2000)

Constituye una mejora con respecto al modelo jerárquico, ya que ofrece una solución eficiente al problema de redundancia de datos. Pero aún así, la dificultad que implica administrar la información en una base de datos de red, ha propiciado que sea utilizado más por programadores que por usuarios finales.

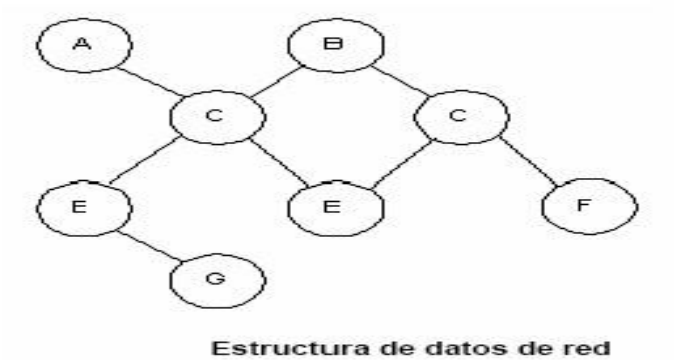


Ilustración 2: Estructura de datos de red

1.4.3 Modelo Orientado a Objetos

Diseñadas para ser eficaces, desde el punto de vista físico, con el objetivo de almacenar objetos complejos y métodos (Date, 2003). El desarrollo del diseño orientado a objetos no niega las ideas existentes, sino que se basa en ellas y las mejora. Un modelo orientado a objetos tiene obligatoriamente que cumplir con lo siguiente: la estructura básica de trabajo son los objetos, no algoritmos; donde cada objeto no es más que una instancia de una clase ya definida y que dichas clases estarán relacionadas únicamente por relaciones de herencia. En caso de que no se cumpla una de las condiciones antes mencionadas, pues no se estaría en presencia de un modelo orientado a objetos. Una BD orientada a objetos incorpora todos los conceptos importantes del paradigma de objetos:

- ✓ Encapsulación: Propiedad que permite ocultar la información al resto de los objetos, impidiendo así accesos incorrectos o conflictos.
- ✓ Herencia: Propiedad a través de la cual los objetos heredan comportamiento dentro de una jerarquía de clases.
- ✓ Polimorfismo: Propiedad de una operación mediante la cual puede ser aplicada a distintos tipos de objetos. (Booch, 1996)

1.4.4 Modelo Relacional

Es el modelo más utilizado en la actualidad para modelar problemas reales y administrar datos dinámicamente. Su idea fundamental es el uso de entidades (tablas), compuestas por registros (las filas de una tabla), que representarían las tuplas y campos (las columnas de una tabla). (Hansen, 2000)

El modelo relacional de datos tiene varios aspectos que lo caracterizan, desde el punto de vista estructural se puede plantear que este modelo organiza la información en forma de tablas y de esta manera es que sus usuarios la perciben y no de otra. Desde el punto de vista de la integridad se puede decir que el modelo establece varias restricciones de integridad, las cuales deben ser cumplidas por todas sus tablas; y desde el punto de vista de la manipulación, tiene definidos un grupo de operadores a través de los cuales interactúa con la información almacenada en las tablas de la base de datos y sus resultados los devuelve en tablas. (Date, 2003)

Entre estos operadores se encuentran proyectar, restringir y juntar. Debido a que la información consultada en cada operación se encuentra en forma de tabla, existe la posibilidad de concatenar unos operadores con otros. Este modelo también está capacitado para hacer procesamiento de conjunto, esto significa que al devolver el resultado de cualquier operación, el resultado lo devuelve en tablas que están conformadas por un conjunto de filas, no por filas individuales. (Date, 2003)

Además está compuesto por 5 componentes que lo identifican: un conjunto abierto de tipos escalares, un generador de tipos de relación y una interpretación propuesta de dichos tipos, herramientas para definir variables de dichos tipos de relación generados, una operación asignación relacional para asignar valores de relación a las variables de relación mencionadas y un conjunto abierto de operadores relacionales genéricos para derivar valores de relación de otros valores de relación. (Date, 2003)

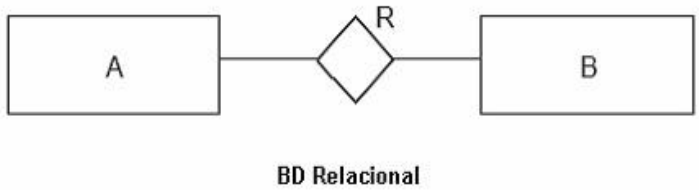


Ilustración 3: Base de datos relacional

En este modelo, el lugar y la forma en que se almacenen los datos no tienen relevancia (a diferencia de otros modelos como el jerárquico y el de red). Esto tiene la considerable ventaja de que es más fácil de entender y de utilizar para un usuario casual de la Base de Datos. La información puede ser recuperada o almacenada por medio de consultas que ofrecen una amplia flexibilidad y poder para administrarla.

Lenguaje de Consulta Estructurado. SQL

El origen del Lenguaje de Consulta Estructurado (SQL, por sus siglas en inglés) está ligado al origen de la base de datos relacional. Representa un estándar para los lenguajes relacionales y todos los Sistemas Gestores de Base de Datos Relacionales hacen uso de este estándar, aunque cada SGBD desarrolla su propio SQL con pequeñas invariantes a partir del SQL estandarizado. El SQL se considera un lenguaje declarativo ya que las sentencias SQL describen qué información se quiere procesar pero no cómo llevarla a cabo, el cómo procesar la información que se quiere lo determina el SGBD. (MUKHAN, y otros, 2002)

Para realizar consultas a la bases de datos se utiliza el lenguaje de manipulación de datos conocido como DML (*Data Manipulation Language*). Para consultar los datos de la base de datos el DML cuenta con ciertos comandos que permiten recuperar los datos almacenados en la base de datos y actualizar la base de datos añadiendo nuevos datos, suprimiendo datos antiguos o modificando datos previamente almacenados.

Estructura básica de los comandos DML.

Select

Las consultas de selección se utilizan para extraer información de las bases de datos. Recuperan datos de una o más tablas y muestran el conjunto de registros en una hoja de datos. También pueden ser utilizadas para agrupar datos y para calcular sumas, recuentos, promedios y otros tipos de totales. (Monte, y otros, 2003)

SELECT A1,A2,...,An **FROM** r1,r2,...,rn **WHERE** P
A1,A2,...,An : Describe la salida deseada.
r1,r2,...,rn : Nombre de las tablas de donde se extraerán los datos.
P: Conjunto de condiciones.

Insert

Lo más importante en este tipo de consultas es que el número de campos que devuelve la consulta sea el mismo que intentas insertar. También es fundamental que los tipos de los campos coincidan. (Casares, 2013)

INSERT INTO T (campo1, campo2, .., campoN) **VALUES** (valor1, valor2, ..., valorN)
T: Tabla en la cual se va a insertar.
campo1, campo2, .., campoN: Atributos en los cuales se va a insertar.
valor1, valor2, ..., valorN: Valores que se van a insertar.

Delete

Para saber con seguridad qué registros se eliminarán, primero se puede ejecutar una consulta de selección que utilice el mismo criterio y después ejecutar la consulta de borrado. Es importante mantener copias de seguridad de los datos por si se eliminan los registros equivocados, podrán ser recuperados. (Casares, 2013)

DELETE FROM T **WHERE** P
T: Tabla de la que se desea borrar.
P: Conjunto de condiciones.

Update

Update es la instrucción del lenguaje SQL que es utilizada para modificar los registros de una tabla. Como para el caso de *Delete*, es necesario especificar por medio de *Where* cuáles son los registros en los que se quiere hacer efectivas las modificaciones. Además, obviamente, se tendrá que especificar cuáles son los nuevos valores de los campos que se desean actualizar. (Álvarez, 2001)

UPDATE T **SET** Campo1=Valor1, Campo2=Valor2, ... CampoN=ValorN **WHERE** P
T: Tabla que se desea actualizar.
CampoN: Atributo que se desea actualizar.
ValorN: Valor nuevo que va a tomar el atributo.
P: Conjunto de restricciones.

1.5 Descripción general del objeto de estudio

Los mecanismos para el acceso y la recuperación de información en bases de datos consisten básicamente en la realización de consultas simples a la base de datos. Una consulta de información ocurre cuando alguien desea conocer sobre algún contenido y este se encuentra almacenado en la base de datos. La misma consta de tres procesos fundamentales: la definición de los criterios de búsqueda, la

petición de la información y la obtención del resultado.

Una consulta puede tener varios criterios que a medida que se incrementen la harán más exhaustiva y la acercará más a la obtención de la información deseada. Los sistemas gestores de base de datos brindan al usuario la posibilidad de definir esos criterios de búsqueda por los cuáles se regirá la petición de información. Una vez definido los parámetros de la búsqueda se procede a la realización de la consulta que emite un resultado. Este resultado corresponde al conjunto de información que se encuentra almacenada en la base de datos y que puede arrojar varios valores, incluso en caso de que no exista la información también se obtiene el conocimiento de la inexistencia de la misma.

Los mecanismos utilizados para la recuperación de la información de la base de datos del Sistema de Gestión de Datos Geológicos en la ONRM son la realización de consultas predefinidas, lo que trae como consecuencia que los especialistas de la entidad no puedan definir los criterios de búsqueda, sino que estos se encuentran previamente elaborados. Esto afecta los procesos que se llevan a cabo en la entidad debido a que si se desea acceder a una información específica, información que incluso hoy no está pensada, hay que acceder de forma manual. La imposibilidad del manejo de los datos provoca que no se puedan realizar consultas específicas para cualquier situación que se presente. Esto hace necesario la elaboración de un editor de consultas que pueda ser integrado al Sistema de Gestión de Datos Geológicos y que permita a los especialistas de la ONRM un pleno acceso a la información contenida en la base de datos.

1.6 Análisis de soluciones existentes

En este epígrafe se analizan algunos de los principales editores de consultas existentes a nivel internacional para PostgreSQL.

1.6.1 pgAdmin III

Principales características:

- ✓ Es una aplicación para el diseño y manejo de bases de datos que utiliza el gestor PostgreSQL.
- ✓ Está licenciada bajo *Open Source* (Código abierto).
- ✓ Se puede utilizar en Linux, Solaris, Windows entre otros.
- ✓ Es capaz de gestionar versiones a partir de PostgreSQL 7.3 ejecutándose en cualquier plataforma.

- ✓ La interfaz gráfica es compatible con todas las características de PostgreSQL y facilita la administración. (PgAdminIII, 2011)

Esta herramienta simplifica la gestión de bases de datos ya que permite desde la escritura de simples consultas SQL a la elaboración de bases de datos complejas. Cada vez que se realiza alguna modificación en un objeto, escribe las sentencias SQL correspondientes, lo que hace que, además de una herramienta útil, sea a la vez didáctica.

1.6.2 EMS SQL Manager para PostgreSQL

Se trata de un eficiente administrador que permite gestionar y editar bases de datos. Algunas de las tareas que se pueden realizar con EMS SQL Manager para PostgreSQL son: añadir nuevas entradas e información, realizar búsquedas sobre los datos almacenados, agregar categorías y manipular objetos. Además, este programa es compatible con todas las versiones de PostgreSQL aparecidas hasta la fecha. Solamente funciona en un ambiente Windows. (EMS, 2011)

1.6.3 Advanced Query Tool (AQT)

AQT está en uso desde 1999 y con decenas de miles de licencias vendidas en más de 70 países, es un producto sólidamente probado. Es un rápido y potente gestor de base de datos con una amplia gama de herramientas de consulta para analistas y desarrolladores. Proporciona una interfaz simple y fácil de usar, con un rico entorno de consultas. Es válido destacar que solamente funciona en un ambiente Windows. (AQT, 2012)

1.6.4 DreamCoder para PostgreSQL

El DreamCoder para PostgreSQL es una herramienta completa para el desarrollo de un servidor de base de datos PostgreSQL. El producto cuenta con una interfaz gráfica la cual facilita las tareas de desarrollo como son: la creación de procedimientos, prueba de procedimientos almacenados, construcción de consultas, exportación e importación de datos, construcción gráfica de consultas y sincronización de datos y estructura, entre otras. Es independiente del Sistema Operativo donde se encuentre instalado el servidor de base de datos PostgreSQL, pero es importante aclarar que el DreamCoder para PostgreSQL solamente funciona en un ambiente Windows. Entre sus principales características se encuentran:

- ✓ Puede exportar todos los scripts de creación de los objetos de la base de datos al igual que los

datos.

- ✓ Cuenta con un moderno query builder el cual le facilitará la construcción de complejas consultas sin requerir mayor conocimiento del lenguaje SQL.
- ✓ Cuenta con una opción para la sincronización de datos, donde la única condición es que las tablas sean físicamente iguales.
- ✓ Cuenta con un gran número de opciones para la manipulación, exportación e importación de datos. (SQLDeveloper, 2011)

1.6.5 phpPgAdmin

phpPgAdmin es una aplicación web, escrita en PHP, para administrar bases de datos PostgreSQL. Provee una manera conveniente a los usuarios para crear bases de datos, tablas, alterarlas y consultar sus datos usando el lenguaje estándar SQL.

Dentro de sus principales características se encuentran:

- ✓ Fácil de instalar y configurar.
- ✓ Permite administrar varios servidores.
- ✓ Soporte para PostgreSQL.
- ✓ Administrar todos los aspectos de: usuarios y grupos, base de datos, esquemas, tablas, índices, restricciones, reglas, privilegios, vistas, secuencias, funciones, objetos avanzados e informes. (phpPgAdmin, 2011)

Una vez finalizado el análisis de las principales soluciones existentes a nivel mundial se concluye que las mismas poseen características avanzadas que permiten realizar una gestión total de la base de datos. A pesar de esto no pueden ser utilizadas para dar respuesta a la situación problemática existente, debido a que para trabajar con ellas es necesario un conocimiento avanzado de base de datos pues estas aplicaciones permiten realizar cualquier tipo de consultas y un error a la hora de realizar una consulta pudiera traer consecuencias negativas, como el borrado de tablas o incluso que se perdieran todos los datos de la base de datos.

En el caso del phpPgAdmin a pesar de ser una aplicación web no cumple con los objetivos que se desean, porque no posee una interfaz gráfica donde los usuarios puedan realizar consultas de manera fácil e interactiva. Lo que se pretende lograr es un editor de consultas simples orientado a los especialistas de la ONRM, donde estos con un mínimo de conocimiento, de forma fácil y rápida puedan realizar cualquier petición de información a la base de datos desde la misma aplicación web (SGDG).

La interfaz gráfica del pgAdmin III, EMS SQL Manager, DreamCoder y Advanced Query Tool ostentan la virtud de ser amigable y facilitan el trabajo, por lo que serán utilizadas como guía para la creación del editor de consultas a la base de datos del SGDG.

1.7 Conclusiones parciales

Con la realización de este capítulo se pudo evidenciar que no es suficiente el acceso que existe a la información contenida en la base de datos de la ONRM, debido a que en ocasiones es necesario consultar alguna información y dicha consulta no se encuentra disponible. De las soluciones existentes estudiadas no existe ninguna que se adapte a las necesidades de la ONRM. Es necesaria la elaboración de un editor de consultas que permita a los especialistas de la ONRM tener un mayor acceso a la información contenida en la base de datos.

CAPÍTULO 2: Metodología, Tecnologías y Herramientas.

2.1 Descripción de la Arquitectura del Sistema

La arquitectura de software de un programa o sistema de cómputo es la estructura o las estructuras del sistema, que incluyen los componentes del software, las propiedades visibles externamente de esos componentes y las relaciones entre ellos. (Bass, 2003) La arquitectura de un sistema puede basarse en uno o varios estilos arquitectónicos. Un estilo arquitectónico es una transformación impuesta al diseño de todo un sistema. El objetivo es establecer una estructura para todos los componentes del sistema. (Pressman, 2005)

Un estilo se define a su vez como "…un concepto descriptivo que define una forma de articulación u organización arquitectónica. El conjunto de los estilos cataloga las formas básicas posibles de estructuras de software, mientras que las formas complejas se articulan mediante la composición de los estilos fundamentales". (Reynoso, 2004) Un patrón arquitectónico al igual que un estilo impone una transformación en el diseño de una arquitectura. (Pressman, 2005)

2.1.1 Estilo Arquitectónico

Para el despliegue del sistema la filosofía utilizada es Cliente-Servidor, ya que es utilizada en el SGDG y debe existir correspondencia puesto que el módulo Editor de Consultas va a ser integrado a dicho sistema.

Cliente-Servidor es utilizado para el desarrollo de sistemas de información en el que las transacciones se dividen en procesos independientes que cooperan entre sí para intercambiar información, servicios o recursos. Se denomina cliente al proceso que inicia el diálogo o solicita los recursos y servidor al proceso que responde a las solicitudes. La utilización de este modelo trae consigo menores costes de operación. (Reynoso, 2004)

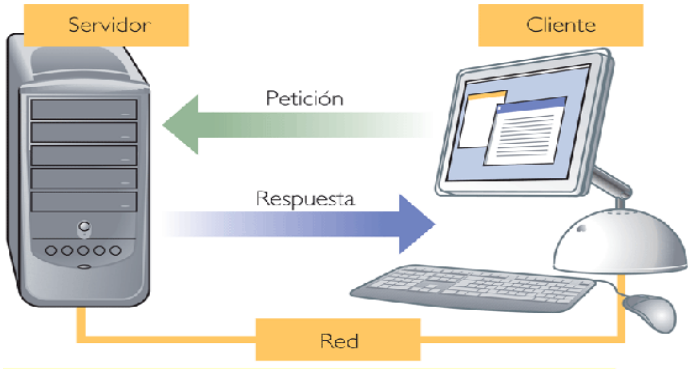


Ilustración 4: Cliente-Servidor

En la ONRM es necesaria la utilización de esta filosofía, debido a que sus especialistas trabajan constantemente con información común y resulta conveniente que esta información no se encuentre repetida en cada uno de los departamentos. Es por esto que en dicha entidad se utiliza un servidor en el cual se almacenan y gestionan todos los datos, que son consultados por los especialistas de las diferentes áreas.

En esta filosofía Cliente-Servidor, se reconoce el uso del estilo arquitectónico de llamada y retorno, el cual enfatiza la escalabilidad del sistema, dentro del cual se aplica el patrón arquitectónico Modelo-Vista-Controlador (MVC). Este patrón es utilizado para establecer la estructura lógica de la aplicación. Se encarga de separar el modelado del dominio, la presentación y las acciones basadas en datos ingresados por el usuario en tres niveles diferentes: (Potencier, y otros, 2008)

- Modelo: Representa la información con la que trabaja la aplicación (Lógica de negocio).
- Vista: Maneja la visualización de la información.
- Controlador: Procesa las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista.

La utilización del MVC brinda un conjunto de ventajas, tales como:

- El soporte de vistas múltiples dado que la vista se halla separada del modelo y no hay dependencia directa del modelo con respecto a la vista.
- La adaptación al cambio dado que el modelo no depende de las vistas, por lo que agregar nuevas opciones de presentación generalmente no afecta al modelo. (Reynoso, 2004)

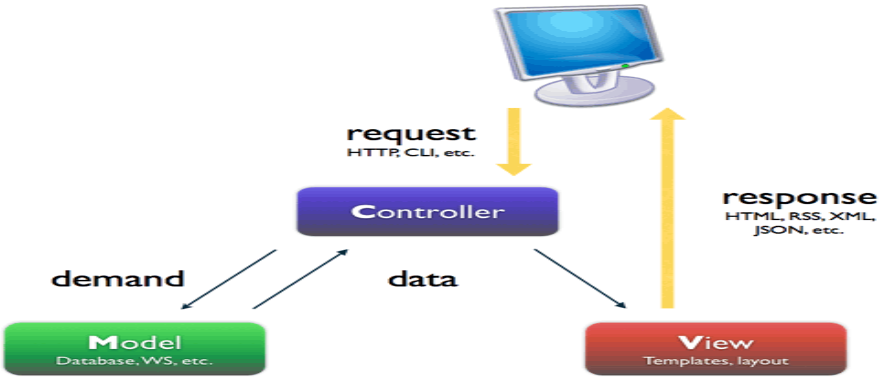


Ilustración 5: Modelo Vista Controlador

2.2 Metodología de desarrollo de software

Las metodologías de desarrollo del software son una combinación de modelos convencionales utilizadas para que el proyecto final tenga la calidad requerida, que se ajuste al tiempo planificado para su construcción y que utilice los menores recursos posibles. (Menéndez, 2011)

El éxito de un sistema depende de numerosas actividades y etapas, donde el impacto de definir la metodología de desarrollo idónea constituye uno de los pasos fundamentales. La metodología de desarrollo de software se define como el conjunto de procedimientos, técnicas, herramientas y soporte documental a la hora de desarrollar un software. Se clasifican en dos tipos: ágiles y pesadas (o robustas). (Menéndez, 2011)

Al realizarse el proyecto SGDG, en el que se integrará el Módulo Editor de Consultas, se seleccionó como metodología de desarrollo de software: Proceso Unificado de Desarrollo (RUP), la cual es una metodología robusta que ha sido aplicada en un gran número de proyectos en la Universidad de las Ciencias Informáticas por lo que se tiene un profundo nivel de experiencia y conocimiento en la misma.

2.2.1 Metodología robusta: Proceso Unificado de Desarrollo (RUP)

El Proceso Unificado Racional (*Rational Unified Process* en inglés) es un proceso de desarrollo de software, lo que se traduce en un conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software. Sin embargo, el Proceso Unificado es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software,

para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto. (Jacobson, y otros, 2000)

Los verdaderos aspectos definitorios de esta metodología se resumen en tres características esenciales: (Pressman, 2007)

- Dirigido por casos de uso: El proceso de desarrollo avanza a través de una serie de flujos de trabajo que parten de los casos de uso.
- Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de desarrollo y los usuarios deben estar de acuerdo. RUP ayuda al arquitecto a centrarse en los objetivos adecuados como la comprensibilidad, la capacidad de adaptación al cambio y la reutilización.
- Iterativo e incremental: Propone la división del trabajo en mini proyectos. Cada mini proyecto constituye una iteración que resulta en un incremento. Las iteraciones responden a pasos en los flujos de trabajo y los incrementos al crecimiento del producto software.

RUP se repite a lo largo de una serie de ciclos que constituyen la vida de un sistema. Cada ciclo constituye una versión del sistema. Cada ciclo consta de cuatro fases: Inicio, Elaboración, Construcción y Transición. Cada fase se subdivide en iteraciones en las cuales se desarrolla un conjunto de disciplinas o flujos de trabajos que están divididos en dos grupos, de Ingeniería y de Apoyo:

De Ingeniería:

- Modelación del negocio.
- Requerimientos.
- Análisis y diseño.
- Implementación.
- Prueba.
- Instalación.

De apoyo:

- Administración del proyecto.
- Administración de configuración y cambios.
- Ambiente.

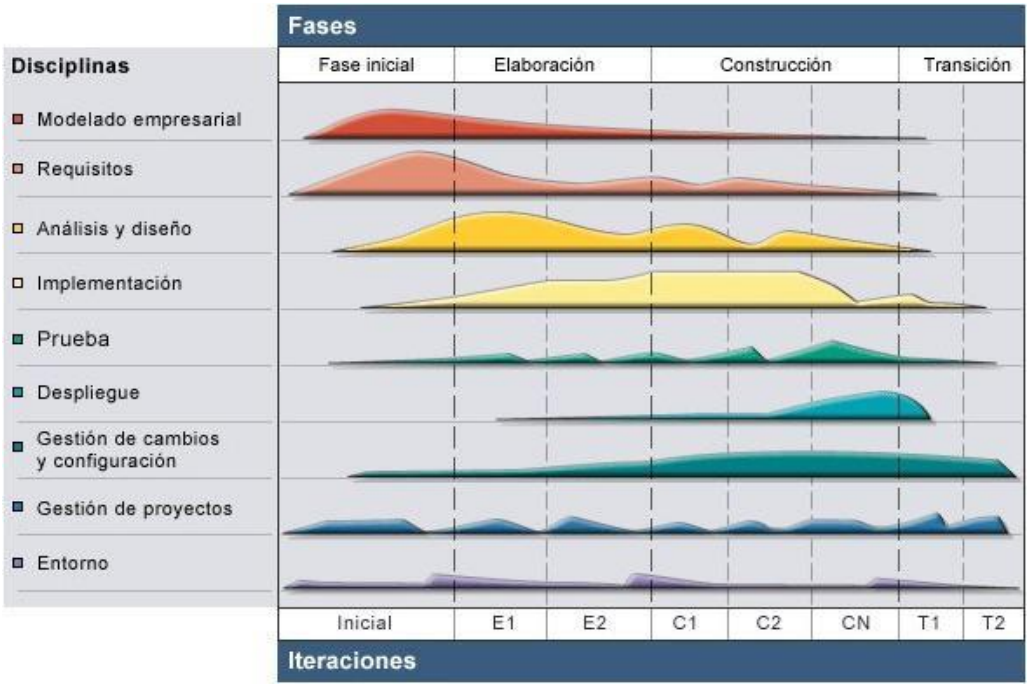


Ilustración 6: RUP en dos dimensiones (Pressman, 2007)

¿Por qué utilizar RUP?

Se seleccionó esta metodología porque el proyecto SGDG, donde será integrado el Módulo Editor de Consultas, es de gran envergadura y bastante extenso, lo que precisa del uso de esta metodología pesada debido a que está guiada por una fuerte planificación y genera una extensa documentación, factor imprescindible que permite que todo el proceso del software quede completamente documentado.

2.3 Lenguaje de Modelado

Cualquier rama de ingeniería o arquitectura ha encontrado útil desde hace mucho tiempo la representación de los diseños de forma gráfica. Desde los inicios de la informática se han estado utilizando distintas formas de representar los diseños de una forma más bien personal o con algún modelo gráfico. La falta de estandarización en la manera de representar gráficamente un modelo impedía que los diseños gráficos realizados se pudieran compartir fácilmente entre distintos diseñadores.

Se necesitaba por tanto un lenguaje no sólo para comunicar las ideas a otros desarrolladores sino también para servir de apoyo en los procesos de análisis de un problema. Con este objetivo se crearon los

Lenguajes de Modelado. La metodología RUP emplea para la especificación, visualización, construcción y documentación de los artefactos que genera durante el proceso de desarrollo del software, el Lenguaje Unificado de Modelado.

2.3.1 Lenguaje Unificado de Modelado (UML 2.1)

El Lenguaje Unificado de Modelado, prescribe un conjunto de notaciones y diagramas estándar para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware y organizaciones del mundo real. (Jacobson, y otros, 2000)

¿Por qué utilizar UML?

Es utilizado UML para la realización del Módulo Editor de Consultas, ya que es el lenguaje de modelado sobre el que se basa la metodología RUP para la creación de sus artefactos. Los modelos generados en cada fase del proceso de desarrollo del software sirven como documentación técnica del Módulo para obtener un mayor entendimiento del mismo y para su futura revisión. Otra de las ventajas que ostenta UML es que organiza el proceso de diseño de forma tal que los clientes y el equipo de desarrollo lo comprendan, lo que le permite al cliente señalar cambios si no se han captado claramente sus necesidades o si ocurre un cambio de opinión.

2.4 Herramientas CASE

Las herramientas CASE se definen como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de desarrollo de un software. (Murillo Alfaro, 1999) Estas herramientas permiten la automatización de los procesos de construcción y aportan como mayor beneficio calidad en el proceso de desarrollo del software. Para el desarrollo de los módulos integrados al SGDG, se seleccionó como herramienta CASE: Visual Paradigm. Con el objetivo de mantener el mismo estándar de diseño, en la confección del Módulo Editor de Consultas se utiliza también dicha herramienta.

2.4.1 Herramientas para el Modelado: Visual Paradigm 8.0

Visual Paradigm propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del

software a través de la representación de todo tipo de diagramas. (Fuster, 2006) Es una herramienta libre y multiplataforma, presenta licencia comercial y gratuita, soporta aplicaciones web, brinda integración con los principales IDEs, permite la realización tanto de ingeniería directa como inversa además de un diseño enfocado al negocio que genera un software de mayor calidad.

¿Por qué utilizar Visual Paradigm?

Se utiliza esta herramienta CASE pues está basada en software libre, es multiplataforma, fácil de instalar y actualizar, tiene compatibilidad entre ediciones, permite la portabilidad y estandarización de la documentación. Además, presenta una interfaz de usuario sencilla que permite realizar los diagramas y artefactos que se generan durante el desarrollo del software.

2.5 Lenguajes de Programación

Un lenguaje de programación es un idioma artificial diseñado para expresar procesos que pueden ser llevados a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. (Rojas, 2007)

Los lenguajes de programación web se pueden clasificar en lenguajes del lado del cliente y del lado del servidor. La diferencia básica que existe entre estos es, que el lenguaje del lado del cliente se ejecuta en su mismo navegador web, y en cambio, el lenguaje del lado del servidor, ejecuta las aplicaciones en el servidor sin tener en cuenta el navegador web del cliente. A continuación se explican los lenguajes del lado del cliente y del servidor que se utilizan en el Módulo Editor de Consultas.

2.5.1 Lenguaje de Marcado de Hipertexto (HTML 5)

HTML será utilizado como lenguaje de programación por el lado del cliente. Es un lenguaje muy sencillo que permite describir documentos hipertextos, se basa en especificar en el texto la estructura lógica del contenido (títulos, párrafos de texto normal, enumeraciones, definiciones, citas, entre otros), así como los diferentes efectos que se quieren dar. Se basa en el uso de etiquetas, que podrán incluir una serie de atributos o parámetros, en su mayoría opcionales. (Kemedy, y otros, 2004) Se ha convertido en un estándar debido a la gran popularidad que ha alcanzado. Puede ser interpretado por cualquier navegador independiente del sistema operativo. (Eguíluz Pérez, 2007)

2.5.2 Hoja de Estilo en Cascada (CSS 3)

Las Hojas de Estilo en Cascada son un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla. Se utilizan para dar estilo a documentos HTML o XML (por extensión en XHTML), separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML y XML. El principal objetivo del uso de CSS es separar la estructura del documento de su presentación y de esta forma aumentar la organización del código y el riesgo de pérdida de uno u otro, además ofrece a los desarrolladores control total sobre el estilo y formato de sus documentos. (w3c, 2003)

¿Por qué utilizar CSS?

Mediante la utilización de CSS se pueden establecer estilos al Módulo Editor de Consultas acorde a los estilos del SGDG. Permite que una misma página contenga varias hojas de estilo. Este lenguaje de programación también posibilita que un documento HTML sea más pequeño y fácil de entender, agilizándole el trabajo a los desarrolladores.

2.5.3 JavaScript

JavaScript será utilizado como lenguaje de programación por el lado del cliente para la creación de las vistas mostradas al usuario, y su principal función estará encaminada a las validaciones dentro de las páginas HTML. Es un lenguaje de programación interpretado que se utiliza principalmente para crear páginas web dinámicas, es multiplataforma, orientado a eventos con manejo de objetos, cuyo código se incluye directamente en el mismo documento HTML. (Brandendaugh, 2000)

¿Por qué utilizar JavaScript?

Al ejecutarse el código JavaScript del lado del cliente posibilita que no se le hagan solicitudes innecesarias al servidor. Es importante utilizarlo para la validación de los datos en los formularios, además posibilita que se creen interfaces de usuarios complejas e interactivas. Es dinámico por lo que responde a eventos en tiempo real y puede combinarse con otras herramientas de desarrollo web, como las hojas de estilo CSS.

2.5.4 Pre-procesador de Hipertexto (PHP 5.4.3)

El lenguaje de programación utilizado por el lado del servidor fue PHP. Es un lenguaje interpretado diseñado especialmente para desarrollo web y que puede ser incrustado dentro de código HTML. Generalmente se ejecuta en un servidor web, tomando el código en PHP como su entrada y creando

páginas web como salida. Puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas de forma gratuita. Es además un lenguaje multiplataforma que permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite. (Charte Ojeda, 2004)

¿Por qué utilizar PHP?

Se utiliza en el desarrollo del SGDG como lenguaje de programación del lado del servidor PHP, por lo tanto también en el Módulo Editor de Consultas, ya que es un lenguaje multiplataforma, orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos. Permite la conexión a la mayoría de motores de bases de datos que se utilizan actualmente y aplica técnicas de Programación Orientada a Objetos (POO). Otras características que permiten elegirlo es que posee una amplia documentación y es libre, por lo que se presenta como una alternativa de fácil acceso para todos.

2.6 Entorno de Desarrollo Integrado (IDE)

Un Entorno de Desarrollo Integrado es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, poder utilizarse para varios. Consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. (Valdés Altamirano, 2008)

2.6.1 NetBeans 7.2

El IDE NetBeans es un entorno de desarrollo integrado disponible para Windows, Mac, Linux y Solaris. El proyecto NetBeans consiste en un IDE de código abierto y una plataforma de aplicaciones que permiten a los desarrolladores crear rápidamente aplicaciones web, empresariales, de escritorio y aplicaciones móviles utilizando la plataforma Java, PHP entre otras. (NetBeans, 2012)

¿Por qué utilizar NetBeans?

Es utilizado como IDE NetBeans, ya que es multiplataforma, libre y gratuito. Brinda la posibilidad de desarrollar aplicaciones web utilizando PHP y ofrece soporte para Symfony. Tiene además, una excelente interfaz con múltiples opciones y un aceptable completamiento de código.

2.7 Marcos de trabajo

En el desarrollo de software, un marco de trabajo simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Proporciona, además, estructura al código fuente, forzando al desarrollador a crear códigos más legibles y fáciles de mantener. Por último, un marco de trabajo facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. (Potencier, y otros, 2008)

2.7.1 Symfony 2.1.7

Symfony es un marco de trabajo diseñado para optimizar el desarrollo de las aplicaciones web, está basado en el patrón clásico del diseño web Modelo Vista Controlador y desarrollado en PHP 5. Es compatible con numerosos gestores de bases de datos como MySQL, PostgreSQL, Oracle y SQL Server. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja, automatizando las tareas más comunes, lo que permite al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Es multiplataforma, fácil de instalar y configurar.

¿Por qué utilizar Symfony?

Se utiliza Symfony como marco de trabajo para PHP, ya que permite dividir el sistema en aplicaciones y módulos independientes, reduciendo de manera considerable el trabajo del programador. Facilita el desarrollo de una aplicación mediante la automatización de patrones, que son utilizados para solucionar tareas comunes, agrupando operaciones complejas en instrucciones sencillas. Este framework ha integrado en uno, las ventajas de este y otros ya existentes, el resultado del mismo es un framework estable, productivo y muy bien documentado. Posee un alto nivel de organización, es de desarrollo ágil y tiene buena seguridad, ya que utiliza el ORM doctrine, el cual impide la inserción de inyecciones SQL a las bases de datos.

2.8 ExtJS 4.1.1

ExtJS es una biblioteca de clases JavaScript que permite construir aplicaciones complejas en Internet. Esta biblioteca incluye: Componentes de Interfaz de Usuario (UI) personalizables, funcionales y reutilizables, modelo de componentes extensibles, una Interfaz de Programación de Aplicaciones (API) fácil de usar y Licencias Open Source (GPL) y comerciales. (Sencha, 2012)

¿Por qué utilizar ExtJS?

Se utiliza ExtJS debido a que facilita la creación de aplicaciones complejas utilizando componentes que ya se encuentran predefinidos y además con esta biblioteca no se necesita validar el código para que funcione bien en la mayoría de los navegadores.

2.9 Servidor Web

Un servidor web es un programa que se ejecuta continuamente en un computador, manteniéndose a la espera de peticiones de ejecución que le hará un cliente o un usuario de Internet. Se encarga de contestar a estas peticiones de forma adecuada, entregando como resultado una página web o información de todo tipo de acuerdo a los comandos solicitados. En el desarrollo del proyecto SGDG se utiliza el servidor web Apache.

2.9.1 Apache 2.2.22

Como servidor Web se utilizó Apache pues corre en varios Sistemas Operativos, lo que lo hace prácticamente universal. Está diseñado para ser un Servidor Web potente y flexible que pueda funcionar en la más amplia variedad de plataformas y entornos. Es una tecnología gratuita de código abierto, altamente configurable de diseño modular, que permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. (Ciberaula, 2010)

¿Por qué utilizar Apache?

Apache es utilizado debido a que es compatible con múltiples sistemas operativos, es una tecnología gratuita y de código abierto. Además es el servidor web por excelencia debido a su robustez, estabilidad y a que es altamente configurable.

2.10 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD) es un sistema de software que permite la definición de bases de datos así como la elección de las estructuras de datos necesarias para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación. (Álvarez, 2007) Son múltiples las opciones de SGBD que existen en el mercado del software, entre las que poseen licencia libre se encuentran: PostgreSQL, SQLite, Firebird, Apache Derby y MariaDB.

2.10.1 PostgreSQL 9.1

Es un SGBD objeto-relacional, distribuido bajo Licencia de Software Libre que permite el uso del código fuente en software no libre (BSD) y con su código fuente disponible libremente. PostgreSql utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando. Se ejecuta en la mayoría de los Sistemas Operativos más utilizados en el mundo, incluyendo Linux, varias versiones de Unix y por supuesto Windows. (Claudín, 2011)

¿Por qué utilizar PostgreSQL?

Para el desarrollo del SGDG se utilizó PostgreSQL 9.1, por lo que el Módulo Editor de Consultas hace uso de la misma versión. PostgreSQL posee gran escalabilidad, lo que le permite atender un mayor número de peticiones concurrentes. Es multiplataforma y está licenciado bajo BSD por lo que se tiene libertad de usar, modificar y distribuir en productos comerciales o no comerciales sin costo alguno. Además utiliza PostGIS que es una extensión al sistema de base de datos objeto-relacional PostgreSQL que posibilita el trabajo con datos geoespaciales.

2.11 Conclusiones parciales

Una vez caracterizada la metodología, herramientas y tecnologías a utilizar, se garantiza que sus características satisfacen las necesidades existentes para el desarrollo del Editor de Consultas. Estas tecnologías libres posibilitan la construcción de sistemas de manera gratuita, aspecto de vital importancia para el desarrollo del software en el país. Su selección está basada teniendo como primicia las tendencias actuales de la programación web y están enfocadas al código abierto lo que permitirá que la aplicación sea fácil de desarrollar y mantener.

CAPÍTULO 3: Características del Sistema

3.1 Modelo de Dominio

El Modelo de Negocio de RUP incluye toda la organización, sus relaciones y sus procesos. Sin embargo, el Modelo de Dominio se centra en una parte del negocio, la relacionada con el ámbito del proyecto. En este contexto el término dominio representa una parte del negocio. El Modelo de Dominio ayuda a comprender los conceptos que utilizan los usuarios, los conceptos con los que trabajan y con los que deberá trabajar la aplicación. El proceso para su elaboración tiene tres pasos. En primer lugar identificar las Clases Conceptuales, después dibujarlas en un Diagrama de Clases y finalmente añadir Relaciones y Atributos. (Pressman, 2005)

En resumen, se llama Modelo de Dominio a la representación visual de los conceptos u objetos de interés, sus características y las relaciones que existen entre ellos. Es el mecanismo principal para lograr una mejor comprensión del problema y para establecer conceptos comunes. Es un interlocutor entre clientes y desarrolladores.

3.2 Diagrama de Clases del Dominio

A continuación se muestra el diagrama de dominio, que ofrece una mayor comprensión de los términos utilizados en la elaboración del Editor de Consultas y que se utilizará como guía para el diseño del sistema.

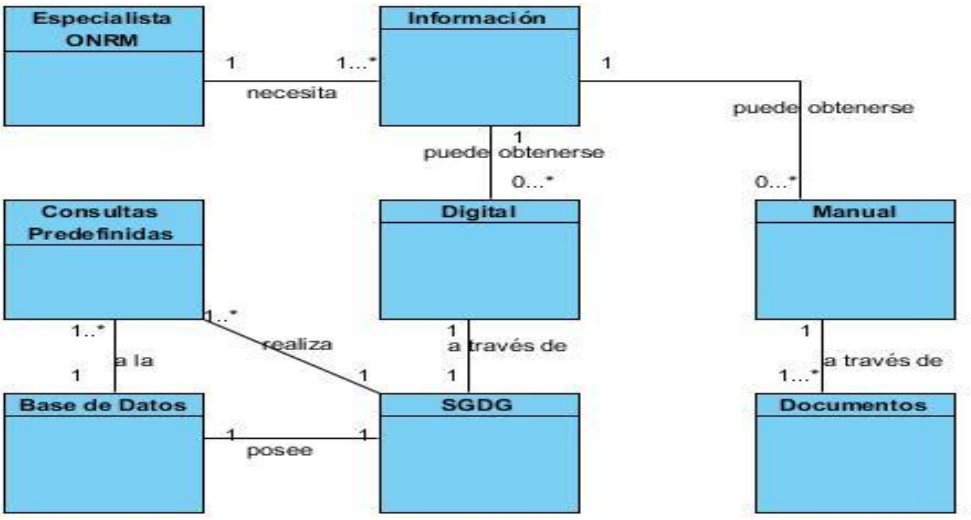


Ilustración 7: Diagrama de Clases del Dominio

3.3 Definición de Clases del Modelo de Dominio

Especialista de la ONRM

Los especialistas de la Oficina Nacional de Recursos Minerales pueden necesitar cualquier información para dar respuesta a una determinada situación.

Información

La información es lo que necesitan los especialistas de la Oficina Nacional de Recursos Minerales para dar respuesta a las situaciones que se puedan plantear en la entidad.

Manual

Es una forma que tienen los especialistas de obtener la información.

Documentos

Los documentos son los materiales analizados cuando la información hay que obtenerla manualmente.

Digital

Es una forma que tienen los especialistas de obtener la información.

SGDG

El Sistema de Gestión de Datos Geológicos es la aplicación con la que interactúan los especialistas de la ONRM cuando la información puede obtenerse de manera digital.

Base de Datos

Es la base de datos que posee el SGDG y donde se encuentra la información digital.

Consultas Predefinidas

Es la forma fundamental que existe en la Oficina Nacional de Recursos Minerales de acceder a la información de la base de datos.

3.4 Breve Descripción del Diagrama

Los especialistas de la ONRM para dar respuestas a las necesidades de los clientes necesitan consultar la información con que cuenta la entidad. Esta información puede obtenerse de dos formas: de manera manual mediante el análisis de documentos o bien de manera digital utilizando el SGDG, que realiza consultas predefinidas a la base de datos. El principal problema existente es que cuando la información

tiene que ser recuperada manualmente el proceso resulta engorroso y requiere de gran cantidad de tiempo.

3.5 Especificación de los Requisitos de Software

Conocidos los principales conceptos asociados al dominio del problema se procede a la modelación del sistema que se desea, para ello primeramente se seleccionan las técnicas para la captura de requisitos. Se identifican los Requisitos Funcionales (RF) y No Funcionales (RNF), además se modelan los RF en representaciones de Casos de Uso del sistema (CUS). “Un proyecto no puede ser exitoso sin una especificación correcta y exhaustiva de los requerimientos.” (Jacobson, y otros, 2000)

3.5.1 Técnicas de Captura de Requisitos

El proceso de identificación de requisitos de un software es una actividad que se lleva a cabo desde el inicio del desarrollo del sistema. En esta etapa de desarrollo del software los analistas extraen los datos que son necesarios para conocer las funcionalidades que implementará el sistema, para ello es necesario la investigación de diferentes fuentes de información. Existen técnicas que permiten realizar el proceso de captura de requisitos de una forma eficiente y segura. Algunas son: entrevista, introspección, cuestionarios, listas de verificación, grabaciones de video y de audio, tormenta de ideas y análisis de la documentación.

Las técnicas utilizadas para la captura de requisitos del Módulo Editor de Consultas son: la tormenta de ideas y la introspección. La tormenta de ideas es una de las técnicas más usadas en este campo, evidenciándose en las reuniones realizadas entre los miembros del proyecto Sistema de Gestión de Datos Geológicos y el autor. Esto posibilitó que los miembros del proyecto expresaran sus ideas sobre las principales características que debía tener el sistema. Por lo que se puede decir que la utilización de esta técnica logra mejorar la calidad del software.

Para la aplicación de la introspección, fue necesario que el autor se pusiera en el lugar del interesado tratando de imaginar cómo desearía éste la aplicación de software. Basado en estas suposiciones se realizó un resumen de las funcionalidades que debería tener la aplicación. El problema radica en que es posible que existan funcionalidades que el interesado no necesita o que incluso no sabe que necesitará en un futuro. (Torres, y otros, 2006)

3.5.2 Requisitos Funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. Permiten expresar una especificación detallada de las responsabilidades del sistema en cuestión. Se mantienen invariables, sin importarle con que propiedades o cualidades se relacionen. (Somerville, 2005) Los RF del Módulo Editor de Consultas son los siguientes:

RF1. Importar una consulta: Permitirá que se puedan cargar consultas con el objetivo de ejecutarlas o poder modificarlas.

RF2. Guardar las consultas realizadas: Permitirá guardar las consultas del lado del cliente para que en el futuro puedan ser importadas.

RF3. Limpiar editor de consultas: Si el usuario se da cuenta que está trabajando de manera incorrecta, esta funcionalidad le permitirá borrar todo lo que se encuentre en el panel donde se realiza la consulta.

RF4. Ejecutar la consulta realizada o importada: Posibilitará que la consulta realizada o importada devuelva los datos deseados de la base de datos, el tiempo que demoró ejecutarla y la cantidad de filas recuperadas. En caso que la consulta esté incorrecta debe mostrar un mensaje con la línea donde existe el error.

RF5. Modificar consultas importadas: Permitirá modificar una consulta previamente importada.

RF6. Exportar los datos obtenidos a formato PDF: Permitirá exportar los datos recuperados de la base de datos.

RF7. Listar las tablas de la base de datos: El editor mostrará todas las tablas de la base de datos para que los especialistas puedan trabajar con ellas y realizar las consultas.

RF8. Arrastrar las tablas listadas: Los especialistas tendrán la posibilidad en el editor gráfico de poder arrastrar las tablas a un panel, donde se podrá ir construyendo la consulta gráficamente.

RF9. Crear una nueva consulta: Los especialistas tendrán la posibilidad de crear consultas de forma manual o mediante el editor gráfico.

RF10. Relacionar las tablas arrastradas: Permitirá a los especialistas de la ONRM poder relacionar las tablas, lo que permitirá ir construyendo la consulta gráficamente.

RF11. Eliminar relación entre tablas: Se le debe permitir a los especialistas de la ONRM poder eliminar las relaciones entre las tablas.

RF12. Definir atributos a devolver: Permitirá seleccionar los atributos a devolver.

RF13 Definir predicado de la consulta: Permitirá definir el predicado que tendrá la consulta, este puede ser DISTINCT o ALL.

RF14. Definir función de un atributo: Permitirá que los especialistas puedan determinar la función que se le aplicará a un atributo (MAX, MIN, SUM, AVE, COUNT) en la consulta.

RF15. Eliminar atributos a devolver: Permitirá eliminar los atributos a devolver.

RF16. Definir Alias de un atributo: Permitirá que los especialistas puedan determinar con que alias será devuelto un atributo en la consulta.

RF17. Ordenar las filas de la consulta realizada: Permitirá ordenar las filas devueltas de acuerdo al orden que se desee, este puede ser descendente o ascendente.

RF19. Eliminar las tablas arrastradas: No es para eliminar tablas de la base de datos, sino que permitirá eliminar tablas del panel donde se construye la consulta en el editor gráfico. Esta funcionalidad deberá eliminar todas las dependencias de esta tabla, dígase, relación con otras tablas y atributos que estén seleccionados para ser devueltos.

RF20. Definir si se desea agrupar: Permitirá a los especialistas decidir si desean que la información obtenida de la base de datos esté agrupada por un atributo.

3.5.3 Requisitos No Funcionales

Los requisitos no funcionales especifican las propiedades del sistema. Determinan el rendimiento del mismo, disponibilidad, la protección, mantenimiento, interfaces de usuario, portabilidad así como otras cualidades. Son aquellos que no se refieren directamente a las funciones específicas que proporciona el

sistema. (Somerville, 2005) El Módulo que se propone presenta los siguientes requisitos no funcionales:

Requisitos de usabilidad

- El sistema debe poder ser usado por cualquier persona que tenga conocimientos básicos del lenguaje de consultas SQL o al menos de base de datos, para que puedan entender los términos que se manejan en el Editor de Consultas.
- La información deberá estar disponible en todo momento, limitada solamente por las restricciones de acuerdo a las políticas de seguridad definidas.

Requisitos de fiabilidad

- El sistema debe estar disponible todo el tiempo para sus usuarios, descontando el tiempo que se encuentre en mantenimiento y la ocurrencia de alguna falla externa.

Requisitos de confiabilidad

- Cada usuario debe tener los permisos necesarios para realizar operaciones en el módulo.

Requisitos de eficiencia

- El tiempo de respuesta por transacción de las peticiones realizadas al módulo estará en el rango de 2 a 5 segundos, en dependencia de la cantidad de información a procesar.

Requisitos de soporte

- El período de soporte así como las restricciones asociadas se manejarán entre el equipo de desarrollo y los clientes.

Restricciones de diseño

El producto de software debe diseñarse teniendo en cuenta:

- La arquitectura Modelo - Vista – Controlador en la lógica y gestión de los datos en la interfaz.
- Cliente – Servidor como arquitectura de despliegue.
- El lenguaje de programación a utilizar en la implementación de los módulos será PHP, con el framework de desarrollo Symfony2.

Requisitos de interfaz

- El sistema debe tener indicadores que permitan conocer al usuario las acciones que debe realizar, por ejemplo botones con íconos sugerentes y alternativa textual.

- El sistema debe permitir al usuario transitar de una tarea a otra sin necesidad de obligarlo a realizar acciones innecesarias o no deseadas, por ejemplo para llegar de una tarea a otra el usuario no debe dar más de 3 clics.

Requisitos de hardware

PC Servidor

- RAM 1024 MB.
- Velocidad de procesamiento del microprocesador 1GHz o superior.

PC Cliente

- RAM 512 MB, se recomienda 1024 MB.
- Velocidad de procesamiento del microprocesador 1GHz o superior.

Requisitos de software

PC Servidor

- Instalación del servidor web Apache 2.2.22.
- Instalación del servidor de base de datos PostgreSql 9.1.
- Instalación de PHP 5.
- Configuración de PHP con las extensiones php5-pgsql, php5-pdo, php5-pdo-pgsql.

PC Cliente

- Navegador web de internet con soporte para HTML 5. Recomendable Firefox 10+ o Google Chrome 14+.

Requisitos Legales, de Derecho de Autor y otros.

- El sistema debe ajustarse y regirse por la ley, decretos leyes, decretos, resoluciones y manuales (órdenes) establecidos, que norman los procesos que serán automatizados.
- Como producto, PNICG-SGDG se distribuye amparado bajo las normativas legales establecidas en el registro comercial emitido por las entidades jurídicas de la Universidad de las Ciencias Informáticas.

3.6 Descripción del Sistema Propuesto

3.6.1 Actores del Sistema

Los actores representan papeles que las personas o dispositivos juegan como impulsores del sistema. Un actor es algo que se comunica con el sistema o producto y que es externo a un sistema en sí mismo. (Pressman, 2005)

Un actor es una agrupación uniforme de personas, sistemas o máquinas que interactúan con el sistema que se está construyendo de la misma forma. (Santiago, 2002)

En resumen, el actor del sistema es una entidad externa que de alguna manera interactúa con este. Por lo regular estimula el sistema con eventos de entrada o recibe algo de él. En este caso particular el actor del sistema será el especialista de la ONRM.

Actor	**Descripción**
Especialista de la ONRM	Los especialistas de la ONRM son las personas que utilizarán el Editor de Consultas con el fin de poder acceder a cualquier información de la Base de Datos.

Tabla 1: Descripción de los actores del sistema

3.6.2 Casos de Uso del Sistema

Un caso de uso proporciona a los desarrolladores una visión de lo que quieren los usuarios. Se enfoca en lo que hace el sistema más que en la forma como lo hace. Además en un caso de uso, un actor que utiliza el sistema inicia un evento que desencadena una serie de interacciones relacionadas en el sistema. (Kendall, 2005) Un caso de uso describe un proceso que deberá convertirse en una funcionalidad.

3.6.3 Diagrama de Casos de Uso del Sistema

Un diagrama de casos de uso del sistema representa gráficamente las funcionalidades principales del sistema y su interacción con los actores.

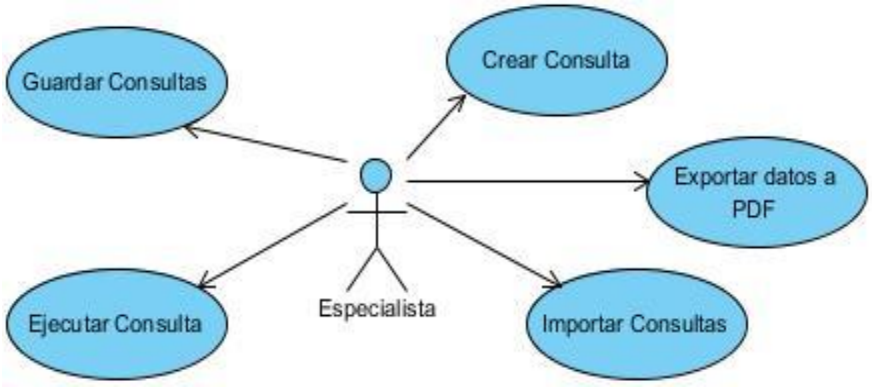


Ilustración 8: Diagrama de CU del Sistema

3.6.4 Descripción textual de los Casos de Uso del Sistema

Para lograr una mejor comprensión de cada caso de uso del sistema, se realiza la descripción textual de los mismos. A continuación se presenta la descripción textual para el caso de uso: Crear Consulta.

Descripción textual del Caso de Uso Crear Consulta

Caso de Uso:	Crear Consulta.
Actores:	Especialista
Propósito:	Este caso de uso se lleva a cabo con el objetivo de realizar consultas que puedan extraer información de interés de la base de datos.
Resumen:	Este caso de uso se inicia cuando el especialista decide crear una consulta y termina una vez que el especialista decide que ya la consulta está creada.
Precondiciones:	El usuario tiene que acceder al Editor de Consultas.
Referencias:	RF 9, RF 7, RF 8, RF 10, RF 11, RF 12, RF 13, RF 14, RF 15, RF 16, RF 17, RF 18, RF 19, RF 20
Prioridad:	Crítico

Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
1. El especialista decide crear una consulta y accede al Editor de Consultas.	**2.** El sistema muestra el Editor de Consultas donde los especialistas pueden crear la consulta de forma manual o crear la consulta de forma gráfica. • En caso de decidir realizar la consulta de forma gráfica ver Alternativa 1 “Realizar Consulta de Forma Gráfica”. • En caso de decidir realizar la consulta de forma manual ver Alternativa 2 “Realizar Consulta de Forma Manual”.
Flujos Alternos	
Alternativa 1 “Realizar Consulta de Forma Gráfica”	
Acción del Actor	**Respuesta del Sistema**
1. El especialista decide crear una consulta gráficamente.	**2.** El Sistema muestra el “Constructor Gráfico de Consultas” que cuenta con 3 secciones: Una a la izquierda con el listado de los schemas y las tablas **(A)** que posee cada uno de ellos en la base de datos, una a la derecha que es para donde se arrastraran las tablas **(B)** para ir construyendo la consulta y una en la parte inferior donde se definirán los atributos **(C)** que devolverá la consulta así como los criterios que tendrá esta.

3. El especialista arrastra las tablas que desee hacia el panel derecho, donde podrá realizar relaciones entre ellas, estas relaciones se mostrarán en la sección “Unión”, ubicada en la parte inferior del Editor. Para construir la consulta el especialista también puede arrastrar los atributos de salida a la sección “Selección”, allí podrá definir parámetros que le permitirán realizar una búsqueda con el nivel de detalle que decida. Además en la sección “Criterio”, podrá definir otros criterios para la consulta.	**4.** El sistema va actualizando la consulta en la ventana “Editor SQL” a medida que se vayan realizando los cambios.
5. El especialista determina que la consulta está terminada puede observarla en la ventana “Editor SQL”, ejecutarla para obtener una respuesta, exportarla PDF o guardarla.	
Alternativa 2 “Realizar Consulta de Forma Manual”	
Acción del Actor	**Respuesta del Sistema**
1. El especialista decide crear una consulta manualmente y selecciona la ventana “Editor SQL”.	**2.** El sistema muestra la ventana “Editor SQL” **(D)** que posee un panel que permite al especialista ir construyendo la consulta de forma manual.
3. El especialista determina que ya concluyó su consulta y puede ejecutarla, exportarla a PDF o guardarla.	
Poscondiciones:	Una consulta sql que pueda ser guardada, ejecutada para obtener información de interés de la base de datos.

Tabla 2: Descripción textual del Caso de Uso Crear Consulta

3.7 Conclusiones parciales

Una vez terminado el capítulo se puede concluir que los diagramas elaborados facilitan un mejor entendimiento del dominio del problema y muestran cómo funciona el sistema. Las técnicas de captura de requisitos utilizadas posibilitaron que este proceso se desarrollara satisfactoriamente, lo que garantiza que el Módulo Editor de Consultas esté acorde a las necesidades de la ONRM. La realización de este capítulo permitió la creación de las bases para realizar el diseño y la implementación del sistema.

CAPÍTULO 4: Construcción y validación de la solución propuesta

4.1 Análisis y Diseño

Como se explicó anteriormente la metodología utilizada para esta investigación es RUP, la cual presenta entre sus características que es adaptable y configurable, por lo que no se está obligado a hacer uso de todas las actividades que define, sino que se pueden realizar solo aquellas que se consideren necesarias para garantizar el desarrollo del software. El flujo de trabajo que corresponde desarrollar es Análisis y Diseño, el cual tiene como objetivo principal traducir los requisitos de software a un diseño que describa cómo realizar la implementación del sistema. El Análisis posibilita refinar y estructurar los requisitos para obtener una comprensión más detallada de los mismos.

El equipo de desarrollo puede prescindir del Análisis cuando los requisitos son sencillos y conocidos, si se puede identificar de manera simple la forma que va a adoptar el sistema y se cuenta con la comprensión correcta de los requisitos permitiendo la construcción de un software que los represente lo más directamente posible. (Jacobson, y otros, 2000) Por las condiciones explicadas anteriormente, en la presente investigación se decidió prescindir del modelo de análisis. Otra de las razones es que se desea evitar el coste en tiempo que implica el Análisis. De esta manera a continuación se presenta el desarrollo del Diseño como parte del flujo de trabajo Análisis y Diseño propuesto por RUP.

4.2 Modelo de Diseño

El modelo del diseño es un modelo de objetos que describe la realización física de los casos de uso centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además, sirve de abstracción de la implementación del sistema y es, de ese modo, utilizada como una entrada fundamental de las actividades de implementación. (Jacobson, y otros, 2000)

En el modelo de diseño, los casos de uso son realizados por las clases de diseño y sus objetos. Se representa por un sistema de diseño que denota el subsistema de nivel más alto del modelo. La utilización de otro subsistema es, entonces, una forma de organización del modelo de diseño en porciones más manejables.

4.2.1 Patrones de Diseño

Un patrón de diseño provee un esquema para refinar los subsistemas o componentes de un sistema de software, o las relaciones entre ellos. Describe la estructura comúnmente recurrente de los componentes en comunicación, que resuelve un problema general de diseño en un contexto particular. (Buschmann, 1996) Los Patrones de Diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción o interfaces. Cada patrón describe un problema que ocurre una y otra vez en nuestro ambiente, y luego describe el núcleo de la solución a ese problema, de tal manera que puede usar esa solución sin hacer jamás la misma cosa dos veces.

Para el desarrollo de la solución se aplicaron diferentes patrones de diseño, fundamentalmente los patrones GRASP y GoF, con el objetivo de facilitar el mantenimiento del software y contribuir a la realización de un producto reutilizable y escalable.

Patrones GRASP

Se ha hecho uso en el diseño del sistema de los patrones GRASP para la asignación de responsabilidades, ya que estos describen los principios fundamentales de diseño de objetos para dicha actividad. Los patrones GRASP constituyen además un apoyo para la enseñanza y entendimiento del diseño de objetos, ejerciendo el razonamiento de una forma sistemática, racional y explicable.

El framework Symfony implementa varios patrones GRASP, estos se pueden apreciar en las diferentes interacciones de los componentes internos del framework. A continuación se mencionan los patrones GRASP utilizados:

- Creador: Propone que una clase B cree instancias de una clase A dado determinadas condiciones. (Larman, 1999) El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Una de las características más poderosas del framework symfony2 es la utilización del contenedor de inyecciones de dependencias, que es un objeto que sabe cómo crear los objetos de una aplicación. Para ello, conoce todas las relaciones entre las clases y la configuración necesaria para instanciar correctamente cada clase. Este patrón también se evidencia en la clase app.js porque se encarga de inicializar todas las clases del sistema, interfaz y lógica de negocio de crear consulta.

- Controlador: Asignar la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase que represente el "sistema" global. (Larman, 1999) Este patrón define quién debe encargarse de atender un evento del sistema. El controlador es una parte vital de todo proyecto web ya que es el encargado de procesar las diferentes peticiones hechas por el cliente. Este patrón se puede ver representado en la clase DefaultController que es la encargada de controlar en la aplicación el manejo de los eventos.
- Experto: Este patrón indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. El framework symfony lo utiliza en las clases entidades ya que estas son las encargadas de acceder exclusivamente a los datos del sistema. Se evidencia el uso de este patrón en la aplicación, en el archivo class.js, específicamente en la clase query que es experta en todo lo referente a la construcción de una consulta.
- Bajo Acoplamiento: Este patrón se basa en tener las clases lo menos relacionadas posibles entre sí que se pueda, para que de tal forma que si se produce una modificación en una de ellas, se tenga la menor repercusión posible en el resto de las clases, esto potencia la reutilización. Un buen ejemplo de este patrón es la clase ExceptionListener que se puede reutilizar en otros proyectos y no tiene dependencia de clases. También se evidencia la utilización del bajo acoplamiento en la clase utiles.php, la cual puede ser extraída de esta aplicación y utilizada sin problemas.

Patrones GoF

Otro conjunto de patrones de diseño bien conocidos son los patrones conocidos como grupo de los cuatro GoF, y se dividen en 3 tipos fundamentales: los creacionales que abstraen el proceso de creación de instancias, los estructurales que se ocupan de cómo clases y objetos son utilizados para componer estructuras de mayor tamaño y los de comportamiento que atañen a los algoritmos y a la asignación de responsabilidades entre objetos. De los patrones Gof se utilizó el Observador que pertenece a los de comportamiento.

- Observador: Define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él. El framework symfony lo utiliza para la captura de errores ya que el núcleo del sistema, captura los errores y notifica a todas las clases que estén "escuchando" dicho error. En la

aplicación este patrón se evidencia en la clase ExceptionLinester que es la encargada de vigilar los errores que puedan producirse.

4.2.2 Diagrama de Clases del Diseño por Subsistema

Los subsistemas de diseño son una forma de organizar los artefactos del modelo de diseño en piezas más manejables. El diagrama de clases del diseño presenta las clases junto con sus atributos, operaciones, interfaces y relaciones. También presenta el agrupamiento de clases en paquetes y las relaciones entre ellos. A continuación se muestra el diagrama de clases del diseño del CU Crear Consulta.

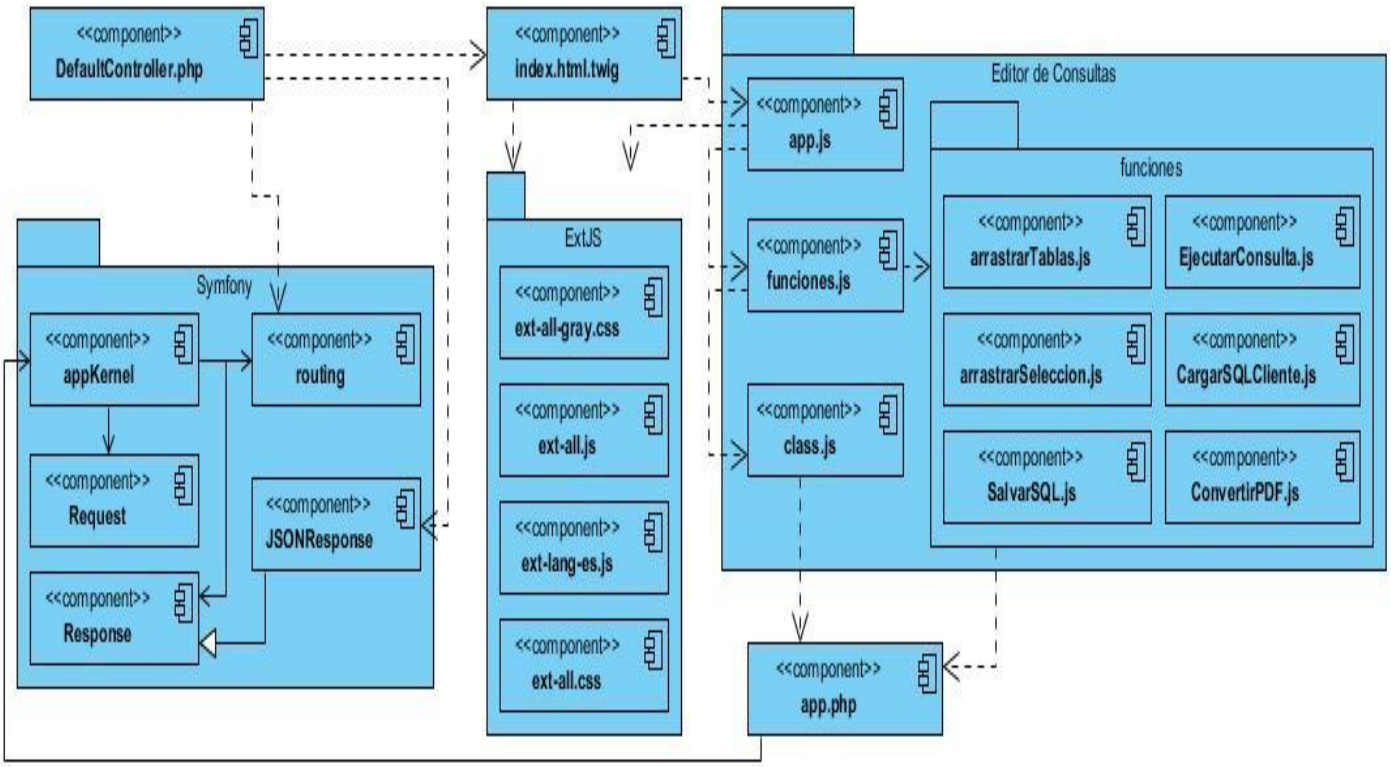


Ilustración 9: Diagrama de Clases del Diseño del CU Crear Consulta

4.3 Modelo de Despliegue

Los diagramas de despliegue son los complementos de los diagramas de componentes que unidos, proveen la vista de implementación del sistema. Describen la topología del sistema, la estructura de los elementos de hardware y el software que ejecuta cada uno de ellos. Los diagramas de despliegue representan a los nodos y sus relaciones. (Daniele, 2007)

Ilustración 10: Diagrama de Despliegue

4.4 Implementación

El Modelo de Implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes. El Modelo de Implementación describe también cómo se organizan los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados y cómo dependen los componentes unos de otros. (Jacobson, y otros, 2000)

4.4.1 Diagrama de Componentes

Un diagrama de componentes muestra las dependencias lógicas entre componentes de software, se relacionan con los diagramas de clases ya que un componente normalmente se corresponde con una o más clases, interfaces o colaboraciones; pero un diagrama de Componentes tiene un nivel más alto de abstracción que un diagrama de clases, usualmente un componente se implementa por una o más clases (u objetos) en tiempo de ejecución.

Componente: un componente puede ser considerado como una unidad autónoma dentro de un sistema o subsistema, tiene una o más interfaces proporcionadas o requeridas, y sus interioridades permanecen ocultas e inaccesibles. Un componente se va modelando a través de todo el ciclo de desarrollo y sucesivamente se va refinando hasta llegar a su implantación y creación de su módulo ejecutable. (Ramírez, 2009)

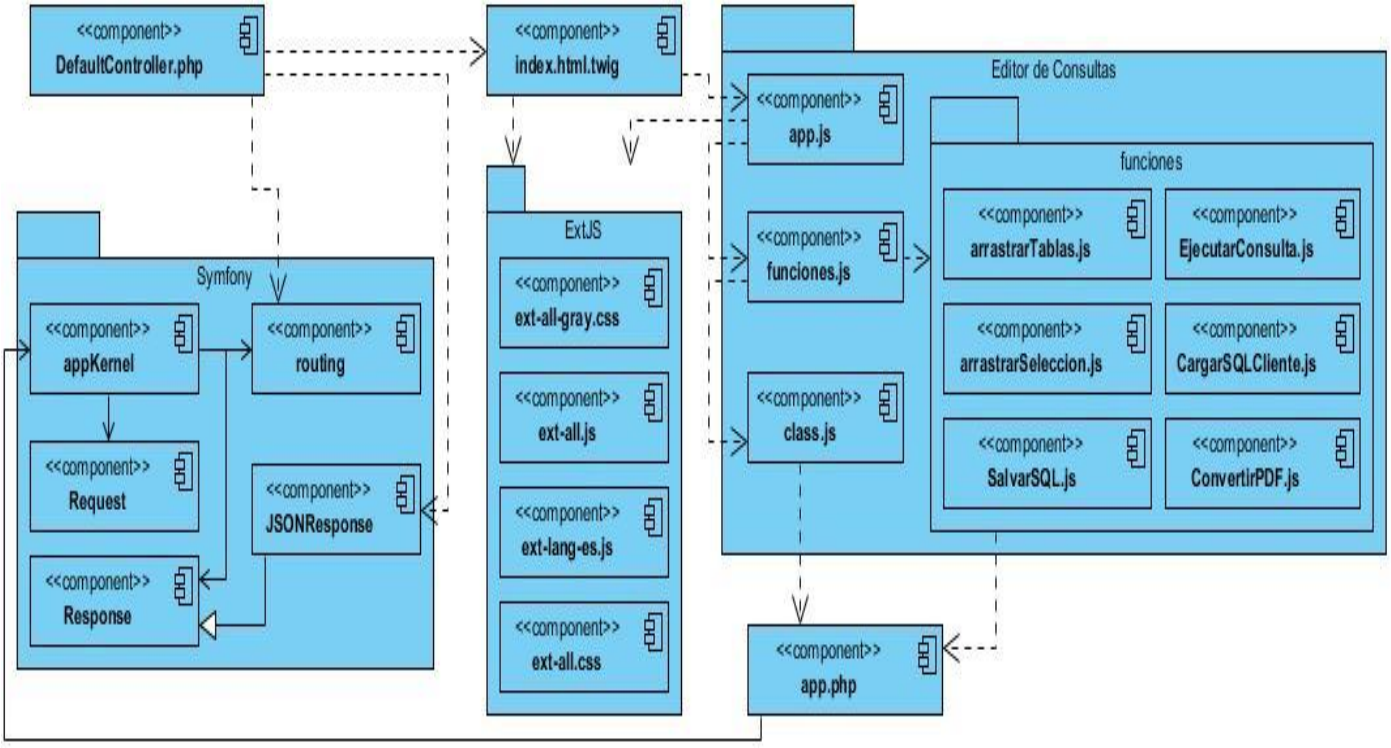


Ilustración 11: Diagrama de Componentes

4.5 Prueba

El flujo de trabajo Prueba persigue como objetivo probar el sistema desarrollado, verificando que cumple con los requisitos definidos y que se encuentra libre de errores. En este flujo se prueba el resultado de la implementación, incluyendo construcciones internas como intermediarias, así como la versión final del sistema que será entregada a los usuarios. (Jacobson, y otros, 2000)

Realizar pruebas a un software es imprescindible para verificar la calidad y el adecuado funcionamiento del mismo. La prueba es un proceso de ejecución de un programa con la intención de comprobar que el producto satisface los requisitos y se comporta como se desea. Para que las pruebas tengan éxito es necesario realizar casos de pruebas que tengan probabilidad de descubrir errores en el sistema y utilizar técnicas que nos guíen el proceso de la prueba.

4.5.1 Tipos de Pruebas

Las pruebas poseen una estructura en la que se define el objetivo de la prueba a realizar, la descripción detallada de la prueba y la técnica a utilizar. RUP define cuatro niveles de prueba:

- Pruebas de unidad: se realizan durante la fase de construcción, específicamente en el flujo de trabajo de implementación; las cuales se basan en probar los componentes implementados como unidades individuales. Las pruebas de unidad están divididas en dos grupos, pruebas de caja blanca y pruebas de caja negra.
- Pruebas de integración: se llevan a cabo durante la fase de construcción, las mismas involucran a un número creciente de módulos y terminan probando el sistema como conjunto. Estas pruebas se pueden plantear desde un punto de vista estructural o funcional. Verifican que los componentes interaccionan entre sí de un modo apropiado después de haber sido integrados en el sistema. Se toman como casos de prueba los casos de uso del diseño. Para ello se utiliza el diagrama de secuencia correspondiente y se diseñan combinaciones de entrada y salida del sistema que lleven a distintas utilizaciones de las clases y en consecuencia de los componentes, que participan en el diagrama.
- Pruebas de sistema: prueban que el sistema funciona globalmente de forma correcta. Cada prueba del sistema prueba combinaciones de casos de uso bajo condiciones diferentes. Se prueba el sistema como un todo probando casos de uso unos detrás de otros y si es posible, en paralelo. En este nivel existen una gran variedad de pruebas que se utilizan según el interés que se tenga respecto al funcionamiento del software.
- Pruebas de aceptación: se realizan para permitir que el cliente valide y verifique todos los requisitos pactados. Estas pruebas las realiza el usuario final en lugar del responsable del desarrollo del sistema. El cliente es quien impone los requisitos pues quien mejor que él para dar fe de su satisfacción. (López Quesada, 2005)

Para validar el correcto funcionamiento del sistema implementado se seleccionaron las pruebas unitarias, y dentro de esta específicamente se eligió el método de prueba de caja negra, para certificar que el sistema responde adecuadamente a los diferentes requisitos del diseño.

4.5.2 Pruebas de Caja Negra

Las pruebas de caja negra, también denominadas pruebas de comportamiento, se centran en los requisitos funcionales del software, o sea, la prueba de caja negra le permite al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. La prueba de caja negra intenta encontrar errores de las siguientes categorías: funciones

incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a base de datos externas, errores de rendimiento y errores de inicialización y de finalización.

Para desarrollar la prueba de caja negra se usó la técnica de la Partición de Equivalencia la cual divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. Dentro del método de Caja Negra la técnica de la Partición de Equivalencia es una de las más efectivas, pues permite examinar los valores válidos e inválidos de las entradas existentes en el software, descubre de forma inmediata una clase de errores que, de otro modo, requieren la ejecución de muchos casos antes de detectar el error genérico. La partición equivalente se dirige a la definición de casos de pruebas que descubran clases de errores, reduciendo así el número de clases de prueba que hay que desarrollar.

4.5.3 Casos de Prueba

Los casos de prueba son el diseño de un conjunto de variables o condición de entrada para demostrar que los requisitos de una aplicación se cumplen parcial o completamente. Estos casos de prueba normalmente se describen por cada caso de uso del sistema y contienen un identificador, la descripción del caso de uso y variables asociada ha dicho caso de uso. A continuación se muestra el Diseño de Casos de Prueba para el Caso de Uso Crear Consulta.

Casos de Prueba del Caso de Uso Crear Consulta

Descripción General: Este caso de uso se inicia cuando el especialista decide crear una consulta y termina una vez que el especialista decide que ya la consulta está creada.

Condiciones de Ejecución: El usuario tiene que acceder al editor de consultas y decidir realizar una consulta.

Secciones a probar en el Caso de Uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central

SC1: Crear Consulta De forma gráfica.	EC 1.1: Arrastrar tablas al panel derecho.	Una vez accedido al editor de consultas el sistema muestra las tablas de la base de datos en el panel izquierdo del editor, estas se encuentran separadas por schemas. Cuando el usuario arrastra una tabla el sistema la muestra en el panel derecho con la lista de atributos que posee.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar la tabla deseada al panel derecho.
	EC 1.2: Arrastrar tablas a un lugar que no sea el panel derecho.	El sistema no realiza ningún cambio.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar la tabla deseada a un lugar que no sea el panel derecho.

	EC 1.3: Arrastrar atributos de salida a la sección “Selección”.	Los atributos de las tablas arrastradas al panel derecho pueden ser arrastrados a la sección “Selección” para ser devueltos en la consulta. El sistema muestra el atributo arrastrado en dicha sección.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar la tabla deseada al panel derecho. Arrastrar el atributo deseado a un lugar que no sea la sección “Selección”.
	EC 1.4: Arrastrar atributos de salida a un lugar que no sea la sección “Selección”.	El sistema no realiza ningún cambio.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar la tabla deseada al panel derecho. Arrastrar el atributo deseado a la sección “Selección”.

	EC 1.5: Definir parámetros de atributos de salida.	El sistema permite que se le definan a los atributos de salida: función, alias, si se desea agrupar por este atributo, si se desea ordenar por este atributo y el predicado.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar la tabla deseada al panel derecho. Arrastrar el atributo deseado a la sección “Selección”. Definir parámetros de los atributos de salida.
	EC 1.6: Eliminar atributos de salida.	El sistema brinda la posibilidad de eliminar atributos de salida con todos los parámetros que este tenga definidos.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar la tabla deseada al panel derecho. Arrastrar el atributo deseado a la sección “Selección”. Clic en el ícono que representa la acción eliminar.

	EC 1.7: Realizar unión de tablas.	Las tablas arrastradas al panel derecho se pueden relacionar entre ellas a través de atributos con el mismo tipo de datos. El sistema muestra una línea entre los atributos relacionados y crea la unión de las tablas en la sección “Unión”.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar las tablas deseadas al panel derecho. Relacionar tablas.
	EC 1.8: Realizar unión de tablas por atributos que no posean el mismo tipo de datos.	El sistema muestra un mensaje indicando que no es posible unir las tablas porque los atributos no poseen el mismo tipo de datos.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar las tablas deseadas al panel derecho. Relacionar tablas.

	EC 1.9: Eliminar unión de tablas.	El sistema brinda la posibilidad de eliminar las uniones existentes entre las tablas.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar las tablas deseadas al panel derecho. Relacionar tablas. Clic en la sección “Unión” Seleccionar la Unión a eliminar. Clic en el botón “Eliminar”.

	EC 1.10: Adicionar criterios de consultas.	En la sección “Criterios” el especialista tiene la posibilidad de adicionar nuevos criterios de comparación a la consulta. El sistema brinda la posibilidad de seleccionar el atributo, el operador, el valor con el que se quiere comparar y el conector para permitir unir más de un criterio.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar las tablas deseadas al panel derecho. Clic en la sección “Criterios” Se oprime el botón “Adicionar”. Se definen los parámetros de este criterio.

	EC 1.11: Eliminar criterios de consultas.	El sistema brinda la posibilidad de eliminar criterios de consultas previamente creados.	Clic en Editor de Consultas. Seleccionar schemas. Arrastrar las tablas deseadas al panel derecho. Clic en la sección "Criterios" Seleccionar el criterio a eliminar. Clic en el botón "Eliminar".
SC 2: Crear Consulta de forma manual.	EC 2.1: Crear Consulta de forma manual.	El sistema brinda la posibilidad al especialista de teclear una consulta.	Clic en Editor de Consultas. Seleccionar la ventana "Editor SQL".

Tabla 3: Escenarios del CUS Crear Consulta

Descripción de las variables:

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	predicado	combo	Si	Permite seleccionar el tipo de predicado.
2	atributos	array	Si	Variable que almacena los atributos de salidas.

3	function	combo	Si	Permite seleccionar la función.
4	alias	textfield	Si	Permite seleccionar el alias de un atributo.
5	groupby	checkbox	Si	Permite seleccionar si se desea hacer Group By por ese atributo.
6	orderby	combo	Si	Permite seleccionar si se desea hacer Order By por ese atributo.
7	tablas	array	No	Variable que almacena las tablas a devolver.
8	atributo	combo	Si	Permite seleccionar un atributo para definirle criterio.
9	operador	combo	Si	Permite definir el operador para el criterio.
10	valor	textfield	Si	Permite definir el valor con el que se va a comparar en el criterio.
11	conector	combo	Si	Permite definir el conector a utilizar para unir los criterios.
12	sqlquery	textfield	No	Permite crear una consulta de forma manual

Tabla 4: Variables CUS Crear Consulta

Matriz de Datos:

Id del escenario	Escenarios	predicados	atributos	function	alias	groupby	orderby	tablas	atributo	operador	valor	conector	sqlquery	Respuesta del sistema	Resultado de la Prueba

1.1	Arrastrar tablas al panel derecho.	–	–	–	–	–	–		–	–	–	–		El sistema añade la tabla a “tablas”, muestra la tabla en el panel derecho y actualiza la consulta en el “sqlquery”.	Satisfactorio.
1.2	Arrastrar tablas a un lugar que no sea el panel derecho	–	–	–	–	–	–	–	–	–	–	–	–	El sistema no realiza ningún cambio.	Satisfactorio
1.3	Arrastrar atributos a la sección “Selección”	–		–	–	–	–	–	–	–	–	–		El sistema muestra el atributo arrastrado, lo inserta en “atributos” y actualiza la consulta en el “sqlquery”	Satisfactorio
1.4	Arrastrar atributos a un lugar que no sea la sección “Selección”	–	–	–	–	–	–	–	–	–	–	–	–	El sistema no realiza ningún cambio.	Satisfactorio
1.5	Definir parámetros de atributos de salida	V(DISTINCT)	–	V(COUNT)	V(PERSONAS)	V(TRUE)	V(ASCENDENTE)	–	–	–	–	–	–	El sistema captura los parámetros y actualiza la consulta en el “sqlquery”	Satisfactorio
1.6	Eliminar atributos de Salida	–		–	–	–	–	–	–	–	–	–		El sistema elimina de “atributos” el seleccionado con todos sus parámetros y actualiza el “sqlquery”.	Satisfactorio

1.7	Realizar unión de tablas	–	–	–	–	–	–	–	–	–	–	–	El sistema muestra la unión y actualiza la consulta en el “sqlquery”	Satisfactorio
1.8	Realiza unión de tablas por atributos que no posean el mismo tipo de datos	–	–	–	–	–	–	–	–	–	–	–	El sistema muesta el mensaje ”No se puede realizar la unión”	Satisfactorio
1.9	Eliminar unión de tablas	–	–	–	–	–	–	–	–	–	–	–	El sistema elimina la unión y actualiza la consulta en el “sqlquery”.	Satisfactorio
1.10	Adicionar criterios de consulta.	–	–	–	–	–	–	–	V(tpermiso_seg.idpermiso)	V(=)	V(5)	V(OR)	El sistema captura el criterio y actualiza la consulta en el “sqlquery”	Satisfactorio
1.11	Eliminar criterios de consulta	–	–	–	–	–	–	–	–	–	–	–	El sistema elimina los criterios de consulta y actualiza la consulta en el “sqlquery”.	Satisfactorio

2.1	Crear consulta de forma manual	–	–	–	–	–	–	–	–	–	–	–	V(SELECT * FROM tpermiso_seg)	El sistema muestra la consulta creada en el “sqlquery”.	Satisfactorio

Tabla 5: Matriz de Datos CUS Crear Consulta

4.5.4 Resultado de las Pruebas

Una vez realizados los diseños de casos de prueba, estos fueron aplicados al sistema con el objetivo de comprobar que el software desarrollado cumple con las especificaciones definidas y está apto para ser entregado. El Módulo Editor de Consultas fue sometido a una primera iteración de pruebas, mediante la cual se detectaron una serie de no conformidades que fueron corregidas. Luego de darle solución a los problemas encontrados, se realizó una segunda iteración, en la cual se obtuvieron resultados satisfactorios. Por lo anterior no fue necesario efectuar una tercera iteración, demostrando así que el sistema está en condiciones de ser desplegado en la ONRM. En la Ilustración se muestran los resultados obtenidos en las dos iteraciones de pruebas desarrolladas.

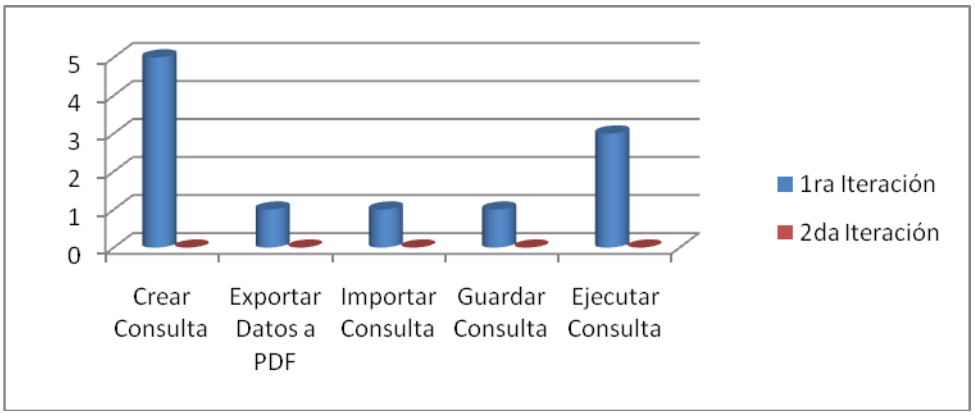


Ilustración 12: Resultado de las iteraciones de pruebas

4.6 Conclusiones parciales

La realización de los flujos de trabajo Análisis y Diseño, Implementación y Prueba hicieron posible que se generaran los artefactos correspondientes, esto permite que el editor obtenido posea gran escalabilidad. Los modelos de diseño permitieron representar la estructura del Editor de Consultas. Mediante el modelo de implementación se especificaron los distintos componentes utilizados para desarrollar la aplicación y la relación existente entre cada uno de ellos. El modelo de despliegue ofreció la posibilidad representar la distribución física que tendrá el sistema. La determinación del método de Caja Negra para realización de las pruebas, específicamente aplicando la técnica de partición equivalente, arrojó como resultado que el sistema satisface los requisitos identificados.

CONCLUSIONES

Al culminar la investigación se arribaron a las siguientes conclusiones:

- El desarrollo del Módulo Editor de Consultas influirá de forma positiva en la capacidad de respuesta de la ONRM ya que posibilitará que los especialistas puedan acceder a la información que deseen de la base de datos.
- La utilización de tecnologías libres hizo posible la construcción del sistema de manera gratuita, aspecto de vital importancia para el desarrollo del software en el país.
- Los artefactos generados en el proceso de desarrollo del editor de consultas servirán como guía para obtener un mejor entendimiento de su estructura, facilitando la realización de posibles modificaciones o la agregación de nuevas funcionalidades al mismo.
- Los casos de prueba definidos y aplicados al módulo garantizaron mejorar la calidad del Editor de Consultas.

RECOMENDACIONES

Con el objetivo de facilitar el trabajo y mejorar el módulo editor de consultas se recomienda:

- Realizar la ayuda del Editor de Consultas para facilitar el trabajo con el mismo.
- Añadir una funcionalidad que permita poder mostrar los datos sobre una cartografía en caso de que la respuesta a la consulta realizada sean datos georeferenciados.

REFERENCIAS BIBLIOGRÁFICAS Y BIBLIOGRAFÍA

Alarcón De Quesada, Ricardo. 1995. Ley No. 76 de Minas. La Habana : s.n., 1995.
Álvarez, Rubén. 2001. desarrolloweb.com. [En línea] 2001. [Citado el: 10 de Diciembre de 2012.] http://www.desarrolloweb.com/articulos/266.php.
Álvarez, Sara. 2007. desarrolloweb.com. [En línea] 2007. [Citado el: 06 de Febrero de 2013.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.
AQT. 2012. [En línea] 2012. [Citado el: 12 de Diciembre de 2012.] http://www.querytool.com/?gclid=CK-mhYW56bQCFQ-f4Aod3lMAcQ.
Barajas, Matilde Romero. 2009. El servicio de consulta dinámica. *GestioPolis.* [En línea] 2009. [Citado el: 10 de Enero de 2013.] www.gestiopolis.com/administracion-estrategia/servicio-de-consulta-olap-base-de-datos.html.
Bass, Len. 2003. *Software Architecture in Practice.* s.l. : Addison Wesley, 2003.
Booch, Grady. 1996. *Análisis y Diseño Orientado a Objetos con Aplicaciones.* Madrid, España : 2da ed.: Pearson Prentice Hall S.A., 1996. 9684443528.
Brandendaugh, Jerry. 2000. *Programación de aplicaciones Web con JavaScript .* Madrid : EDlClONES ANAYA MULTIMEDIA, 2000, 2000. 84-41 5-1 070-9.
Buschmann, F. 1996. Pattern – Oriented Software Architecture. A System of Patterns. . Inglaterra : s.n., 1996.
Casares, Claudio. 2013. SQL Tutorial. [En línea] 23 de Enero de 2013. http://www.cyta.com.ar/biblioteca/bddoc/bdlibros/sql_tutorial.html.
Castaño, Miguel. 1999. *Fundamentos y Modelos de Base de Datos.* España : RA-MA, 1999.
Charte Ojeda, Francisco. 2004. *PHP 5.* 2004. 8441517703.
Ciberaula. 2010. Ciberaula Linux. [En línea] 2010. [Citado el: 05 de Febrero de 2013.] http://linux.ciberaula.com/articulo/linux_apache_intro.
Claudín, Rafael María. 2011. PostgreSQL. [En línea] 2011. [Citado el: 06 de Febrero de 2013.] http://www.postgresql.org.es/sobre_postgresql.
Coronado, Salvador Pozo. 2005. Con Clase. [En línea] Mayo de 2005. [Citado el: 11 de Enero de 2013.] http://www.conclase.net.
Daniele, Ing. Marcela. 2007. El Arte de Modelar. 2007.
Date, J.C. 2003. *Introducción a los Sistemas de Base de Datos.* La Habana : Félix Varela, 2003. Segunda Parte.
DefiniciónDe. 2008. Definición De. [En línea] 2008. [Citado el: 10 de Diciembre de 2012.] http://definicion.de/.
Eguíluz Pérez, Javier. 2007. *Introducción a XHTML.* 2007.
EMS. 2011. [En línea] 2011. [Citado el: 10 de Diciembre de 2012.] http://descargar.mp3.es/lv/group/view/kl46928/EMS_SQL_Manager_for_PostgreSQL_Lite.htm.
Fuster, Gonzalo. 2006. *Evaluación comparativa de herramientas CASE para UML desde el punto de vista notacional.* Madrid : s.n., 2006.
García, Lic. Rosa María Mato. 1999. *Diseño de Base de Datos.* La Habana : s.n., 1999.

Hansen, Gary W. 2000. *Diseño y administración de Base de Datos.* 2000.
Ibaceta, José Bengoechea. 2012. *Microsoft Access : diseño de aplicaciones sencillas de bases de datos.* s.l. : Ideas Propias, 2012. 978-84-9839-226-5.
Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* s.l. : Pearson Educación , 2000.
Kemedy, Chuck y Bill. 2004. *HTML La Guía Completa México.* México : McGRAW-HILL INTERAMERICANA EDITORES, S.A. de C.V., 2004.
Kendall, K. 2005. Análisis y diseño de sistemas. México: Atlacomulco : s.n., 2005.
Larman, Craig. 1999. *UML y Patrones.* México : s.n., 1999. 970-17-0261-1..
López Quesada, Juan Antonio. 2005. *Pruebas del software.* 2005.
López, José E. Gallardo y Ruiz, Carmen M. García. 2010. *Diseño modular.* Málaga : Facultad de Ciencias Matemáticas, 2010, 2010.
Matos, Rosa María. 2001. *Sistemas de Base de Datos.* La Habana : Pueblo y Educación, 2001.
Menéndez, Rafael. 2011. Universidad de Murcia. [En línea] 2011. [Citado el: 30 de Febrero de 2013.] http://www.um.es/docencia/barzana/IAGP/IAGP2-Metodologias-de-desarrollo.html..
Monte, Eduardo Mora, Staff, VV y Pantaleón, Marta E. Zorrilla. 2003. *Iniciación a las Bases de Datos con Access 2002.* Madrid : s.n., 2003. 84-7978-592-6.
MUKHAN, KEVIN T. y JOHN CARNNELL, L. 2002. *Fundamentos Base Datos con Java.* 2002.
Murillo Alfaro, Félix. 1999. *Herramientas CASE.* 1999.
NetBeans. 2012. NetBeans. [En línea] 2012. [Citado el: 04 de Febrero de 2013.] http://netbeans.org/community/releases/70/.
Onorat, Fernando. 2009. Base de Datos. Fundamentos de Base de Datos y algo más... [En línea] 2009. [Citado el: 12 de Enero de 2013.] http://uvfdatabases.wordpress.com/2009/02/07/tipos-de-usuarios-de-la-base-de-datos/.
Parsons, June Jamrich. 2008. *Conceptos de Computación: Nuevas Perspectivas.* 2008.
PgAdminIII. 2011. Guia Documentada para Ubuntu. [En línea] 2011. [Citado el: 10 de Diciembre de 2012.] http://www.guia-ubuntu.com/index.php?title=PgAdmin_III.
phpPgAdmin. 2011. phpPgAdmin. [En línea] 2011. [Citado el: 10 de Enero de 2013.] http://phppgadmin.sourceforge.net/doku.php.
Potencier, Fabien y Zaninotto, François. 2008. *Symfony la guía definitiva.* s.l. : Apress, 2008. ISBN-13:978-1590597866.
Pressman, Roger. 2007. *Ingeniería de Software: un enfoque práctico.* Nueva York : McGraw-Hill, 2007.
—. 2005. *Ingeniería del Software. Un enfoque práctico.* 2005.
RAE. 2012. Real Academia Española. *Diccionario de la Lengua Española - Vigésima Segunda Edición.* [En línea] 2012. [Citado el: 24 de Noviembre de 2012.] http://buscon.rae.es/.
Ramírez, Lic. Elisa Arizaca. 2009. *Análisis y diseño de sistemas II. .* Bolivia : s.n., 2009.
Reynoso, Carlos. 2004. *Introducción a la Arquitectura de Software.* 2004.
Reynoso, Carlos y Kicillof. 2004. *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft Version 1.0.* Buenos Aires : s.n., 2004.
Riquelme, MSc. Silvia López y Morales, MSc. Mirtha Cepero. 2010. Facultad de Economía de la

Universidad de la Habana. [En línea] Marzo de 2010. [Citado el: 11 de Diciembre de 2012.] http://www.fec.uh.cu/CUGIO/1%20acciones/Contenidos/BDR.pdf.
Rob, Peter: Coronel, Carlos. 2004. *Sistemas de Bases de Datos.* 2004. 970-686-286-2.
Rojas, José. 2007. slideshare. [En línea] 2007. [Citado el: 01 de Febrero de 2013.] http://www.slideshare.net/jrojas/tema1-lenguajes-de-programacion.
Rosario, Jimmy. 2005. Observatorio para la cibersociedad. [En línea] 2005. [Citado el: 02 de Diciembre de 2012.] http://www.cibersociedad.net/archivo/articulo.php?art=218.
RUP, Ayuda del. 2006. 2006.
Santiago, C. 2002. Casos de Uso: Un método Práctico para Explorar Requisitos. 2002.
Sencha. 2012. Sencha. [En línea] 2012. [Citado el: 05 de Febrero de 2013.] http://www.sencha.com/.
Somerville, Ian. 2005. *Ingeniería de Software.* Madrid : Pearson Educación, 2005.
SQLDeveloper. 2011. SQLDeveloper.net. *DreamCoder for PostgreSQL.* [En línea] 2011. [Citado el: 12 de Diciembre de 2012.] http://www.sqldeveloper.net/herramientas-base-datos/postgresql/faq.html.
Torres, Ing. Mariela y Paz, Ing. Karim. 2006. Tamaño de una muestra para una investigación de mercado. *Boletín Electrónico Ingeniería Primero.* Guatemala : s.n., 2006. 2.
Valdés Altamirano, Alfonso. 2008. IGNETWORK. [En línea] 23 de Agosto de 2008. [Citado el: 04 de Febrero de 2013.] http://foro.ignetwork.net/showthread.php?15188-IDE-Entorno-integrado-de-desarrollo-%28Concepto-importante%29..
w3c. 2003. World Wide Web Consortium. [En línea] 2003. [Citado el: 02 de Febrero de 2013.] http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo.

GLOSARIO DE TÉRMINOS

Aplicación Web: En la ingeniería de software se denomina aplicación web a aquellas aplicaciones que los usuarios pueden utilizar accediendo a un servidor web a través de Internet o de una intranet mediante un navegador.

GOF: *Gang of Four* es el nombre con el que se conoce comúnmente a los autores del libro *Design Patterns*, referencia en el campo del diseño orientado a objetos. La “Banda de los cuatro”, se compone de los siguientes autores: Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides.

GRASP: Acrónimo de General *Responsability Assignment Software Patterns* (Patrones de Software para la asignación General de Responsabilidad).

HTTP: *Hypertext Transfer Protocol* (Protocolo de Transferencia de Hipertexto) o HTTP es el protocolo usado en cada transacción de la *World Wide Web*.