**UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS**

**FACULTAD 3**

# “Implementación del módulo Diseñador de Consultas para el Generador de Reportes Dinámicos”

Trabajo de Diploma para optar por el título de Ingeniero en

Ciencias Informáticas

**Autor:** Alex González Paz.

**Tutor:** Ing. Yunior Miguel Almaguer Bajuelo.

**Co-Tutora**: MSc. Laura C. Toledo Diez

Ciudad de La Habana, Cuba
Junio, 2009

Lo que no te mata te hace más fuerte (Nietzsche).

# Declaración de Autoría

Declara ser autor del presente trabajo de diploma y reconocer a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Autor: Alex González Paz.

_____

Tutor: Ing. Yunior Miguel Almaguer Bajuelo

_____

# Dedicatoria

A mi madre y a mi padre; juntos porque sería injusto si hablara primero de uno; que me han guiado, aconsejado y se han preocupado por mí, en cada momento de la vida.

A mí querida hermanita que es muy buena conmigo.

A mi familia de La Habana, mi tía querida, mis primas y mi tío.

A mis abuelos por preocuparse tanto por mí.

A todos mis familiares por su apoyo incondicional.

A mis amistades.

A los profesores que contribuyeron a mi formación profesional.

# Agradecimientos

A mis familiares por estar siempre conmigo en los buenos y malos momentos de la vida y depositar su confianza en mí.

A nuestro Comandante en Jefe Fidel por crear esta gran universidad y darnos la oportunidad de formarnos en ella como profesionales y con ello contribuir a la Revolución.

A la Revolución, que ha hecho mi sueño realidad.

A los amigos que he tenido en las diferentes etapas por las que he transitado en la vida y me han apoyado.

A mi tutor por apoyarme cuando lo necesite y guiarme en la realización del presente trabajo.

A mi cotutora por apoyarme en todo momento y guiarme en la adecuada realización del presente trabajo.

A todos los profesores que he tenido a lo largo de mi vida como estudiante, destacando al profesor Pedro Piñeiro que me enseño a ver lo difícil: fácil y sencillo, sin lo que no hubiera sido posible adquirir los conocimientos que hoy poseo.

A todo aquel que de una forma u otra ha contribuido con la realización de este sueño.

Muchas gracias.

# Resumen

Con la realización de este trabajo se pretende llevar a cabo la implementación del Módulo Diseñador de Consultas Dinámicas, utilizando tecnologías web, de forma tal que se pueda proveer a los usuarios de una interfaz amigable y fácil para el diseño de consultas de manera dinámica, evitando que este proceso se realice manualmente en el gestor de base de datos.

Dicho módulo, es uno de varios, del Sistema Informático Generador de Reportes Dinámicos, los cuales son desarrollados en estos momentos como parte del Proyecto "Paquete de Ayuda a la Toma de Decisiones y Sistemas Inteligentes" (PATDSI) en la Universidad de las Ciencias Informáticas (UCI). Este Sistema basa su arquitectura en una ya elaborada por el grupo de arquitectura de la línea de herramientas integrales del Centro de Tecnologías y Almacenamiento de Datos CENTALAD.

**Palabras Claves**: aplicaciones web, diseñadores de consultas, arquitectura, PHP, Java Script, clases, gestores de base de datos, implementación, prueba.

# Contenido

# Introducción.

Cada vez la cantidad de datos que se necesita manipular para extraer información útil a los procesos de negocio de las empresas es mayor. Pero en la medida que los volúmenes de información se han ido haciendo mayor, las Tecnologías de la Información y las Comunicaciones (TIC) han ido desarrollándose y en la actualidad facilitan cualquier tipo de gestión de información en cualquier sector de la sociedad. Son las TIC precisamente las que han llevado a perfeccionar software especializado en el almacenamiento y manipulación de los datos, entre estos software están los Sistemas de Gestión de Bases de Datos (SGBD).

Una colección de datos, normalmente denominada base de datos, contiene información relevante para una empresa y un SGBD [18]: es un tipo de software dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. El propósito general de los sistemas de gestión de base de datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante.

Cada SGBD tiene implementado mecanismos que permiten a los usuarios, recuperar la información de interés, de los datos. Para un usuario utilizar estos mecanismos de consultoría, que emplea el gestor, necesita tener algún conocimiento sobre los mismos.

Existen además otras herramientas para crear informes de una amplia variedad de datos que son producidos por un sistema de información. Estos extraen datos de los archivos o de bases de datos y crean reporte de acuerdo a muchos formatos. Estas herramientas son llamadas: **Generadores de Reportes.**

Una funcionalidad importante en los generadores de reportes la proporcionan los diseñadores de consulta (Query Builder) pues estos le facilitan al usuario consultar y recuperar información de una Base de Datos sin tener muchos conocimientos del gestor de Base de Datos y del lenguaje de consultoría que el mismo emplea.

Los Diseñadores de Consultas han alcanzado un nivel de desarrollo significativo, cuyo alcance va más allá de brindar facilidades a los usuarios en el diseño de consultas dinámicas: disminuyendo el rango de errores, ya que son basados en tecnologías probadas; optimizando el tiempo empleado en su construcción; permitiendo que distintas personas vean datos diferentes o los mismos de forma diferente. Los mismos poseen una interfaz gráfica que facilita que se incluyan uniones de consultas y sub-consultas que permiten guardar y clasificar como privadas o públicas, para volver a utilizar o personalizar cumpliendo los requisitos de seguridad. Es por ello que la elaboración de estas herramientas ha tenido un auge en los últimos tiempos, debido al avance tecnológico y a las necesidades del cliente, dando como resultado la elaboración de diseñadores de consultas como: Active Query Builder, Data Query Builder y el Visual Database Tools.

Todos estos diseñadores son de alto coste adquisitivo, no son multiplataforma a excepción del Diseñador de Consultas de Pentaho Report Designer y no están basados en tecnologías Web, las cuales dan la posibilidad de migrar de sistema operativo o cambiar el hardware libremente sin afectar el funcionamiento de la aplicación, solo requieren de un navegador Web para su utilización y facilitan el trabajo a distancia ya que se puede trabajar desde cualquier computadora con conexión a Internet y al residir en los servidores del proveedor usan la memoria de ellos, dejando espacio libre para correr múltiples aplicaciones al mismo tiempo sin incurrir en deterioros de tiempo del lado del cliente o frustrantes deterioros en el rendimiento.

Actualmente en la Universidad de las Ciencias Informáticas se desarrolla el Proyecto Generador de Reportes Dinámicos, que ya cuenta con algunos módulos implementados, como son: el visor de reportes, diseñador de modelo, diseñador de reportes, administrador de reportes, diseñador de plantillas; pero carece de un módulo que permita facilitar el diseño de las consultas de forma dinámicas, las cuales serán utilizadas por el diseñador de reportes, pues las mismas se diseñan actualmente de forma manual en el gestor de bases de datos.

Teniendo en cuenta todo lo analizado anteriormente y la situación problémica se propone como **problema científico** a resolver en este trabajo, ¿Cómo diseñar de forma dinámica las consultas que serán utilizadas por el “Generador de Reportes Dinámicos”?

Como objetivo general de la investigación para darle solución al problema antes mencionado se tiene:

Desarrollar partiendo de los artefactos obtenidos durante el análisis, diseño y arquitectura una aplicación Web que facilite el diseño de consultas dinámicas.

Idea a Defender

Si se implementa el Módulo Diseñador de Consultas, empleando las Tecnologías Web se facilitará a los usuarios el diseño de las mismas para el Generador de Reportes Dinámicos.

Se define como Objeto de Estudio: El proceso de desarrollo de software.

El Campo de Acción se enfoca en: Implementación del Módulo Diseñador de Consultas para el Sistema Informático Generador de Reportes Dinámicos.

Para dar cumplimiento al objetivo, se definen las siguientes Tareas de Investigación:

**Tareas de la Investigación**:

1. Estudio de sistemas existentes para el diseño de consultas dinámicas.
2. Análisis de tecnologías y herramientas de desarrollo de software utilizadas en la actualidad.
3. Selección de las herramientas y el lenguaje de implementación a utilizar.
4. Estudio de los estándares de codificación y de comunicación establecidos.
5. Obtención del modelo de Implementación.
6. Implementación del módulo utilizando los patrones de diseño establecidos.
7. Diseño de una interfaz gráfica orientada al usuario, agradable y fácil de operar.

Para dar cumplimiento a las tareas se emplearon los siguientes métodos científicos en el proceso investigativo:

**Métodos teóricos:**

**Analítico - Síntesis:** Se utilizó para el procesamiento de la información y análisis de los documentos más importantes acerca del proceso de diseño de consultas y para el arribo de las conclusiones de la investigación.

**Histórico - Lógico:** Para determinar las tendencias actuales de desarrollo de aplicaciones web.

**Hipotético - Deductivo:** Se utilizó para a partir del problema concreto y objetivo plantear la idea a defender.

**Empíricos**

**Observación:** Se utilizó para realizar una valoración a partir de la percepción en la etapa exploratoria de la realidad estudiada para determinar cómo ocurre realmente el proceso de diseño de consultas y cuáles son los principales elementos en los que se debe trabajar.

La investigación realizada se estructurará en cuatro capítulos como sigue:

**Capítulo 1** *Fundamentación Teórica***:** en este capítulo se muestran conceptos generales de los Diseñadores de Consultas. Además, se reflejan conceptos generales y básicos relacionados con el dominio del problema que permiten comprender el diseño de la consulta, tendencias y tecnologías actuales a considerar**.** Se realiza un análisis sobre las principales tecnologías, metodologías de desarrollo, lenguajes de programación y Sistemas Gestores de Bases de Datos que más se emplean en la actualidad.

**Capítulo 2** *Elementos de la arquitectura***:** Se realiza una argumentación de los requisitos no funcionales, un análisis de los patrones o estilos arquitectónicos presentes en la propuesta de solución y se muestran las vistas de despliegue e implementación.

**Capítulo 3** *Descripción y análisis de la solución propuesta*: Se realiza un análisis de la integración con otros módulos, las descripciones de la clases y métodos más importantes, así como una descripción de la tabla que se emplea para obtener los modelos y salvar las consultas.

**Capítulo 4** *Validación de la solución propuesta*: En este capítulo se plasma la realización de las pruebas pertinentes al software.

# Capítulo 1: Fundamentación Teórica.

## 1.1 Introducción

En este capítulo se tratarán los principales conceptos relacionados con el problema a desarrollar, se describirá el entorno del objeto de estudio, y se realizará un estudio de las diferentes tecnologías existentes para decidir cuáles se deben emplear en la confección de la solución propuesta.

## 1.2 Descripción del dominio del problema

### Descripción del objeto de estudio

Los generadores de reportes utilizan los diseñadores de consulta como una herramienta que facilita al usuario, a través de una interfaz sugerente, la posibilidad de diseñar sus propias consultas, ya que nadie sabe más que el mismo usuario, de los datos que mostrará el reporte. Además tienen la ventaja de poder ser utilizados por aquellos usuarios que no tienen conocimientos de un lenguaje de consultoría de datos.

### Descripción del objeto a automatizar

Con el diseñador de consultas propuesto se pretende que el usuario pueda editar sus propias consultas a través de una interfaz web, para lo cual el mismo consta de dos formas de construcción de la consulta, una a través de un diseñador en la cual se pueden definir varios workflow, desplazando las tablas que se emplearán hacia él, las cuales se mostraran en un árbol donde se representan los modelos referentes a las beses de datos con sus respectivas tablas y consultas, se podrá observar los atributos que tiene cada una de ellas y realizar todo el proceso de consultoría a clic cuando se realizan las consultas, se va construyendo un árbol que se muestra en la parte izquierda de la aplicación y en el que se observan las principales sentencias SQL por las que está construida una consulta de forma ordenada y otra forma de construcción de la consulta es a través de un editor de consultas que sería utilizado por los llamados usuarios avanzados, los que tienen conocimientos del lenguaje de consulta que se esté empleando. Al cambiar la vista del diseñador al editor este

se actualizará automáticamente mostrando el código en lenguaje SQL generado por el diseño, de igual forma al editar la consulta se actualizará el modelo correspondiente a la misma. Si es editada de forma correcta permite la ejecución de la misma mostrando la tabla devuelta por ella a través de una ventana.

## 1.3 Análisis de herramientas de diseño de consultas existentes.

Para lograr una visión general del estado del arte se realizó un análisis de algunas herramientas de diseño de consultas como son: el Query Builder del Pentaho Report Designer 1.7.0, el Active Query Builder .NET Edition y el Visual Database Tools

### Diseñador de consultas del Pentaho "Report Designer 1.7.0".

El diseñador de consultas del Pentaho (Report Designer 1.7.0) [1]: es una herramienta que forma parte del diseñador de reportes (Pentaho Report Designer) incorporando la funcionalidad de diseño de consultas complejas con operadores relacionales y subconsultas a través de una interfaz de escritorio amigable y fácil de manejar por el usuario; permitiendo su diseño de dos formas: una en la cual se pueden definir varios marcos de trabajo, desplazando las tablas que se emplearán hacia él ,se podrá observar los atributos que tiene cada una de ellas y realizar todo el proceso de consultoría a clic ,cuando se realizan las consultas, se va construyendo un árbol que se muestra en la parte derecha de la aplicación y en el que se observan las principales sentencias SQL por las que está construida una consulta y otra nos brinda un editor de consultas que sería utilizado por los llamados usuarios avanzados, los que tienen conocimientos del lenguaje de consulta que se esté empleando. (Ver anexo1 y 2)

### Diseñador de consultas "Active SQL Query Builder".

El (Active SQL Query Builder)[2]: es un componente visual de construcción de consultas que se integra al Visual Studio 2005 o Visual Studio 2008 permitiendo a los usuarios finales construir consultas SQL, así como representarlas visualmente. Presenta dos formas de construcción de consultas: una gráfica y otra a través del editor, y permite la combinación de

la construcción de la consulta dinámica con el editor de consultas. Su interfaz es similar al MS-Access[1], pero mucho más poderoso, comparado con otros componentes similares, lo que le permite al usuario definir la agrupación de los campos y construir criterios de un modo simple y directo. Cada unión y subconsulta tienen su propia área de trabajado donde podrá ser construida visualmente. Se puede acceder a cada unión y subconsulta con solo dar un clic en el árbol de estructura. Permite darle un alias familiar a las tablas que tengan nombres pocos sugerentes Soporta Oracle, SQL Server, MS Access, MySQL, PostgreSQL.

Funciona en modo autónomo, almacenando metadato en archivos XML ya que en muchos casos, la conexión directa a la base de datos es indeseable o aún imposible desde el punto de vista de seguridad o funcionamiento, esto también hace la carga metadato casi inmediata independientemente del número de objetos que estén siendo cargados. (Ver anexo3)

## Diseñador de consultas "Visual Database Tools"

La herramienta Visual Database Tools se incluye en diversas ediciones de Visual Studio y posee un diseñador de consultas y vistas que sirve de ayuda para crear y mantener los componentes de recuperación y manipulación de datos de la aplicación. El diseñador está formado por cuatro paneles: el panel Diagrama, el panel Criterios, el panel SQL y el panel Resultados. [3]

**En el panel Diagrama** se muestran las tablas. Cada rectángulo representa una tabla y muestra las columnas de datos disponibles. Las combinaciones se indican mediante líneas entre los rectángulos.

**El panel Criterios** permite definir las opciones de consulta como: columnas de datos a mostrar, cómo se van a ordenar los resultados y qué filas se van a seleccionar; para ello deben especificarse las opciones en una cuadrícula con forma de hoja de cálculo.

[1] Es un programa Sistema de gestión de base de datos relacional creado y modificado por Microsoft para uso personal de pequeñas organizaciones.

**En el panel SQL** se muestra la instrucción SQL para la consulta o vista, la que puede ser editada y resulta especialmente útil para escribir instrucciones SQL que no se pueden crear mediante los paneles Diagrama y Cuadrícula.

**En el panel Resultados** se muestra una cuadrícula con datos recuperados por la consulta o vista y en el panel del Diseñador de consultas y vistas, se muestran los resultados de la consulta SELECT ejecutada más recientemente. El usuario puede modificar la base de datos mediante la edición de los valores de las celdas de la cuadrícula y puede agregar filas o eliminarlas. (Ver anexo4)

## 1.4 Comparación de los Diseñadores de Consultas existentes con la aplicación a desarrollar.

Los diseñadores de consultas estudiados anteriormente presentan algunas desventajas para su empleo ya que tienen un alto coste adquisitivo, no son basados en tecnologías web, no son multiplataforma exceptuando el diseñador de consultas del Pentaho “Report Designer 1.7.0” y en su mayoría están orientados a un gestor de bases de datos específico.

La aplicación a desarrollar es multiplataforma y posee gran flexibilidad permitiendo al usuario el diseño de complejas consultas abstrayéndose de los conocimientos relacionados con los gestores de bases de datos. Permite especificar condiciones, agrupar por campos, entre otras funcionalidades. El usuario solo trabajará con las entidades de la base de datos visualmente, lo que permite crear consultas de alto nivel de complejidad sin dominar un lenguaje de consultoría de datos. La herramienta también brinda la posibilidad a usuarios avanzados de crear sus propias consultas mediante el editor de consultas. Puede ser aplicada a varios gestores de base de datos como: MySQL, PostGreSQL, SQLServer, Oracle. Está basada en tecnologías web y no representa costo de adquisición.

## 1.5 Metodologías de desarrollo

### Rational Unified Process (RUP)

El Proceso Unificado de Desarrollo, fue creado por el mismo grupo de expertos que crearon UML, Ivar Jacobson, Grady Booch y James Rumbaugh en el año 1998. El objetivo que se perseguía con esta metodología era producir software de alta calidad, es decir, que cumpla con los requerimientos de los usuarios dentro de una planificación y presupuesto establecido. Como se expresaba anteriormente, esta metodología concibió desde sus inicios el uso de UML como lenguaje de modelado.

Es un proceso dirigido por casos de uso, este avanza a través de una serie de flujos de trabajo (requisitos, análisis, diseño, implementación, prueba) que parten de los casos de uso; está centrado en la arquitectura y es iterativo e incremental. Además cubre el ciclo de vida de desarrollo de un proyecto y toma en cuenta las mejores prácticas a utilizar en el modelo de desarrollo de *software*.

A continuación se muestran estas prácticas.

- Desarrollo de software en forma iterativa.
- Manejo de requerimientos.
- Utiliza arquitectura basada en componentes.
- Modela el software visualmente
- Verifica la calidad del software.
- Controla los cambios.

Para apoyar el trabajo con esta metodología ha sido desarrollada por la Compañía norteamericana Rational Corporation la herramienta CASE (Computer Assisted Software Engineering) Rational Rose en el año 2000. Esta herramienta integra todos los elementos que propone la metodología para cubrir el ciclo de vida de un proyecto. [4]

## Proceso Unificado Abierto (OpenUP).

OpenUp [5]: es un proceso de desarrollo de software, creado por la fundación eclipse, mínimo y suficiente, lo que significa que solo el contenido fundamental y necesario es incluido. Por lo tanto no provee lineamientos para todos los elementos que se manejan en un proyecto pero tiene los componentes básicos que pueden servir de base a procesos específicos. La mayoría de los elementos de OpenUP están declarados para fomentar el intercambio de información entre los equipos de desarrollo y mantener un entendimiento compartido del proyecto, sus objetivos, alcance y avances.

**Principios del OpenUP.**

- Colaborar para sincronizar intereses y compartir conocimiento. Este principio promueve prácticas que impulsan un ambiente de equipo saludable, facilitan la colaboración y desarrollan un conocimiento compartido del proyecto.
- Equilibrar las prioridades para maximizar el beneficio obtenido por los interesados en el proyecto. Este principio promueve prácticas que permiten a los participantes de los proyectos desarrollar una solución que maximice los beneficios obtenidos por los participantes y que cumple con los requisitos y restricciones del proyecto.
- Centrarse en la arquitectura de forma temprana para minimizar el riesgo y organizar el desarrollo.
- Desarrollo evolutivo para obtener retroalimentación y mejoramiento continuo. Este principio promueve prácticas que permiten a los equipos de desarrollo obtener retroalimentación temprana y continua de los participantes del proyecto, permitiendo demostrarles incrementos progresivos en la funcionalidad.

**Organización de los componentes del OpenUP.**

El OpenUP está organizado en dos dimensiones diferentes pero interrelacionadas: el método y el proceso. El contenido del método es donde sus elementos (roles, tareas, artefactos y lineamientos) son definidos, sin tener en cuenta como son utilizados en el ciclo de vida del proyecto. El proceso es donde los elementos del método son aplicados de forma ordenada

en el tiempo. Muchos ciclos de vida para diferentes proyectos pueden ser creados a partir del mismo conjunto de elementos del método.

Los elementos del OpenUP dirigen la organización del trabajo en los niveles personal, de equipo y de interesados. A nivel personal, los integrantes de un proyecto contribuyen con su trabajo con pequeños incrementos en funcionalidad, denominados microincrementos, los cuales representan los resultados obtenidos en pocas horas o pocos días de trabajo. La solución evoluciona basada en dichos microincrementos de tal forma que el progreso puede ser visualizado efectivamente cada día. Los integrantes del equipo de desarrollo de forma abierta comparten su progreso diario el cual incrementa la visibilidad en el trabajo, la confianza y el trabajo en equipo.

El proyecto en general se divide en iteraciones, las cuales son planificadas en un intervalo definido de tiempo que no superan las pocas semanas. El OpenUP tiene elementos que ayudan a los equipos de trabajo a enfocar los esfuerzos a través del ciclo de vida de cada iteración de tal forma que se puedan distribuir funcionalidades incrementales de una manera predecible, una versión totalmente probada y funcional al final de cada iteración.

El OpenUP estructura el ciclo de vida de un proyecto en cuatro fases: concepción, elaboración, construcción y transición. El ciclo de vida del proyecto provee a los interesados un mecanismo de supervisión y dirección para controlar los fundamentos del proyecto, su ámbito, la exposición a los riesgos, el aumento de valor y otros aspectos.

## 1.6 Tendencias y/o Tecnologias Actuales.

### 1.6.1 Tecnologías Web.

Al iniciarse el Siglo XXI, de la mano de la Internet, la web empezó a ocupar un lugar importante en la vida de las personas. Se vive un proceso vertiginoso y progresivo que ha conducido la migración hacia la web de grandes volúmenes de información desde todos los rincones del planeta, y al desarrollo de aplicaciones web empleando la tecnología de n-capas, que han ido sustituyendo a los programas de escritorio debido a las múltiples ventajas que proporcionan [6]:

- Una empresa puede migrar de sistema operativo o cambiar el hardware libremente sin afectar el funcionamiento de las aplicaciones de servidor.
- No se requieren complicadas combinaciones de hardware/software para utilizar estas aplicaciones, solo un computador con un buen navegador web, ya que las aplicaciones basadas en web no necesitan ser descargadas, instaladas y configuradas.
- Se facilita el trabajo a distancia. Se puede trabajar desde cualquier PC o computador portátil con conexión a Internet.
- Actualizar o hacer cambios en el Software es sencillo y sin riesgos de incompatibilidades.
- Al residir y correr en los servidores del proveedor, esas aplicaciones usan en muchos casos la memoria de las computadoras que ellos corren, dejando más espacio para correr múltiples aplicaciones del mismo tiempo sin incurrir en frustrantes deterioros en el rendimiento.

Unido a este avance tecnológico se han desarrollado también diferentes tecnologías, que se analizarán a continuación para decidir cuáles emplear en la confección de la solución propuesta.

## Tecnologías del lado del Cliente.

Las tecnologías del lado del cliente son incluidas en el código HTML y son directamente interpretados y ejecutadas por el navegador.

**Tecnologías de programación.**

**Lenguajes de desarrollo del lado del cliente:** Los lenguajes del lado del Cliente, como su nombre lo indica son los que se ejecutan en el cliente o navegador. Estos son los encargados de darle dinamismo a la página sin necesidad de enviar la información al servidor para realizar las operaciones que se requieran.

Estos lenguajes son interpretados y pueden acceder a la información HTML que se muestra en un navegador modificándolo o actualizándolo según las necesidades de los programadores.

Algunos de los lenguajes de desarrollo del lado del cliente estudiados son:

## HyperText Markup Language (HTML).

**HTML** (Lenguaje de Marcas de Hipertexto), es el lenguaje de marcado predominante para la construcción de páginas web. Es usado para describir la estructura y el contenido en forma de texto, así como para complementar el texto con objetos tales como imágenes. HTML se escribe en forma de "etiquetas", rodeadas por corchetes angulares (<,>) y puede describir, hasta cierto punto, la apariencia de un documento incluyendo un script (por ejemplo Java Script), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML. Es compatible con navegadores reconocidos como: Internet Explorer, Opera, Firefox, Netscape o Safari.

## Java Script.

Java Script es un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el más utilizado. Es interpretado por el navegador para controlar su apariencia y manipular los eventos que ocurran en su ventana. A través del Java Script se pueden conseguir interesantes efectos en las páginas web, validar formularios, abrir y cerrar ventanas, cambiar dinámicamente el aspecto y los contenidos de una página, realizar operaciones matemáticas sencillas y definir interactividades con el usuario. El navegador del cliente es el mayor recurso, y tal vez el único, con que se cuenta para interpretar las instrucciones Java Script y ejecutarlas. Es un lenguaje de programación bastante sencillo y pensado para hacer las cosas con ligereza.

Entre las acciones típicas que se pueden realizar con este lenguaje se tienen dos vertientes: por un lado los efectos especiales sobre páginas web, para crear contenidos dinámicos y elementos de la página que tengan movimiento y cambiar de color o cualquier otro

dinamismo; por el otro, Java Script permite ejecutar instrucciones como respuesta a las acciones del usuario, con las que se puede crear páginas interactivas con programas como calculadoras, agendas, o tablas de cálculo. Es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones y estructuras de datos complejas. Además, pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente. Con Java Script el programador se convierte en el verdadero dueño y controlador de cada cosa que ocurre en la página cuando la está visualizando el cliente. [7]

## CSS

CSS es un lenguaje de hojas de estilos creado para controlar la presentación de los documentos electrónicos definidos con HTML y XHTML. Separar la definición de los contenidos y la definición de su aspecto presenta numerosas ventajas, ya que obliga a crear documentos bien definidos y con significado completo. Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo en infinidad de dispositivos diferentes. Al crear una página web, se utiliza en primer lugar el lenguaje HTML/XHTML para marcar los contenidos, es decir, para designar la función de cada elemento dentro de la página: párrafo, titular, texto destacado, tabla, lista de elementos, entre otros. Una vez creados los contenidos, se utiliza el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página. [8]

## Tecnologías del lado del Servidor

Estas tecnologías son reconocidas, ejecutadas e interpretadas por el propio servidor y se envían al cliente en un formato comprensible para él.

## Servidores Web.

Un servidor web es un programa que implementa el protocolo HTTP (HyperText Transfer Protocol) que pertenece a la capa de aplicación del modelo OSI y está diseñado para transferir lo que llamamos hipertextos, páginas web o páginas HTML (HyperText Markup

Language): textos complejos con enlaces, figuras, formularios, botones y objetos incrustados como animaciones o reproductores de música y se ejecuta continuamente en un ordenador que responde adecuadamente a las peticiones por parte de un cliente (un navegador web), mediante una página web que se exhibe en el navegador o mostrando el respectivo mensaje si se detectó algún error. [9]

## Servidor Web Apache.

Un Servidor Web Apache [10]: es un sistema de código abierto para plataformas Windows, Unix, Macintosh y otras que implementa el protocolo HTTP. Este servidor posee varias características que lo favorecen como por ejemplo:

- Es una tecnología gratuita de código fuente abierta, lo cual le atribuye transparencia.
- Corre en varios Sistemas Operativos, lo que provoca que sea una herramienta prácticamente universal.
- Es un servidor altamente configurable de diseño modular. Permite aumentar fácilmente su capacidad e instalar cualquier módulo para cumplir una función específica. Otra aspecto importante es que cualquiera que posea experiencia en la programación de C o Perl puede escribir un módulo para realizar una acción determinada.
- Permite personalizar la respuesta ante posibles errores. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto.
- Tiene una alta configurabilidad en la creación y gestión de logs[2]. Apache permite la creación de ficheros de log facilitando de este modo el control de las acciones realizadas en el servidor.

## Servidor Web Internet Information Server (IIS).

Internet Information Server (IIS) [11]: brinda una serie de servicios para los ordenadores que funcionan con Windows. Originalmente era parte del Option Pack (paquete opcional) para Windows NT. Luego fue integrado en otros sistemas operativos de Microsoft destinados a ofrecer servicios, como Windows 2000 o Windows Server 2003. Windows XP Profesional

[2] Un log es un registro oficial de eventos durante un periodo de tiempo en particular. Para los profesionales en seguridad informática es usado para registrar datos o información sobre quién, qué, cuándo, dónde y por qué un evento ocurre para un dispositivo en particular o aplicación.

incluye una versión limitada de IIS. Los servicios que ofrece son: FTP, SMTP, NNTP y HTTP/HTTPS. Este servicio convierte a un ordenador en un servidor de Internet o Intranet, es decir que en las computadoras que tienen este servicio instalado se pueden publicar páginas web tanto local como remotamente. El servidor web se basa en varios módulos que le dan capacidad para procesar distintos tipos de páginas, por ejemplo Microsoft incluye los de Active Server Pages (ASP) y ASP.NET. También pueden ser incluidos los de otros fabricantes, como PHP o Perl.

## Servidor Web Appserv.

Appserv: es una herramienta de código abierto (OpenSource) para Windows que facilita la instalación de Apache, MySQL y PHP en una sola herramienta, dejando las aplicaciones configuradas para su funcionamiento inmediato. En su última versión (AppServ 2.6.0) incluye:

**Apache 2.2.8:** servidor HTTP multiplataforma. **PHP 6.0.0-dev:** lenguaje de programación dinámico que utilizan la mayoría de los gestores de contenidos más populares. Se integra a la perfección con MySQL y Apache. **MySQL 6.0.4-alpha:** gestor de bases de datos, rápido y seguro. **PhpMyAdmin-2.10.3:** interfaz gráfica de administración para MySQL. [12]

Una vez instalado AppServ, se dispone de un servidor web y otro de base de datos propio, configurado de manera local, y que permite tanto publicar como realizar todas las pruebas necesarias en la aplicación web antes de lanzarla a la red.

## Lenguajes de programación del lado del servidor.

**Lenguajes de programación del lado del servidor:** La programación del lado del servidor es un elemento muy importante en el diseño o construcción de sitios web, ya que permite de una u otra forma el manejo dinámico de los datos y disponer de un entorno de aplicación mucho más complejo del que se podría conseguir solamente con HTML y la programación en el lado del cliente. Por ejemplo, se podría necesitar acceder a algunos datos, o incluso programas de un gran sistema. Esos programas podrían trabajar sobre una cantidad considerable de datos y enviar los resultados que pondríamos entonces, poner en una base de datos para una manipulación posterior, antes de generar la página final que se enviará al

cliente. Así pues, se puede decir que los lenguajes del lado del servidor son aquellos que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él.

Algunos de estos lenguajes:

## Active Server Page (ASP).

ASP (Active Server Pages - Página Activa en el Servidor) no es en sí mismo un lenguaje de programación, sino más bien un marco sobre el que se construyen aplicaciones basadas en Internet, apoyándose para ello en el lenguaje HTML y otros lenguajes de script conocidos (generalmente VBScript, pero también JavaScript), en motores de bases de datos y en el lenguaje de consulta SQL. Es una tecnología del lado del servidor desarrollada por Microsoft para páginas web generadas dinámicamente.

Ha pasado por cuatro iteraciones mayores, ASP 1.0 (distribuido con IIS 3.0), ASP 2.0 (distribuido con IIS 4.0), ASP 3.0 (distribuido con IIS 5.0) y ASP.NET (parte de la plataforma .NET de Microsoft). Las versiones pre-.NET se denominan actualmente (desde 2002) como ASP clásico.

ASP no necesita ser compilado para ejecutarse. Existen varios lenguajes que se pueden utilizar para crear páginas ASP. El más utilizado es VBScript, nativo de Microsoft. ASP se puede hacer también en JScript (no JavaScript) y su código puede ser insertado junto con el de HTML. Los archivos cuentan con la extensión (asp). [13]

**Principales ventajas de ASP: [14]**

- Usa Visual Basic Script, siendo fácil para los usuarios.
- Comunicación óptima con SQL Server.
- Soporta el lenguaje JScript (Javascript de Microsoft).

**Algunas desventajas de ASP: [15]**

- Código desorganizado.
- Se necesita escribir mucho código para realizar funciones sencillas.

- Tecnología propietaria.
- Hospedaje de sitios web costosos.

Desde 2002, el ASP clásico está siendo reemplazado por ASP.NET, que entre otras cosas, reemplaza los lenguajes interpretados como VBScript o JScript por otros compilados a código intermedio (llamado MSIL o Microsoft Intermediate Lenguage) como Visual Basic, C# o cualquier otro lenguaje que soporta la plataforma .NET. Fue desarrollado para resolver las limitantes que brindaba su antecesor y para crear Web sencillas o grandes aplicaciones. Los archivos cuentan con la extensión (aspx). Para el funcionamiento de las páginas se necesita tener instalado IIS con el Framework .Net.

**Principales ventajas de ASP.NET: [16]**

- Completamente orientado a objetos.
- Controles de usuario y personalizados.
- División entre la capa de aplicación o diseño y el código.
- Facilita el mantenimiento de grandes aplicaciones.
- Incremento de velocidad de respuesta del servidor.
- Mayor velocidad.
- Mayor seguridad.

**Algunas desventajas de ASP.NET:**

- Mayor consumo de recursos.
- Tecnología propietaria.
- Hospedaje de sitios web costosos.

## Java Server Page (JSP).

Es un lenguaje para la creación de sitios web dinámicos, acrónimo de Java Server Pages. Está orientado a desarrollar páginas web en Java. JSP es un lenguaje multiplataforma creado para ejecutarse del lado del servidor. El denominado contenedor JSP (que sería un componente del servidor web) es el encargado de tomar la página, sustituir el código Java que contiene por el resultado de su ejecución, y enviarla al cliente. JSP fue desarrollado por

Sun Microsystems y comparte ventajas similares a las de ASP.NET, desarrollado para la creación de aplicaciones web potentes. Posee un motor de páginas basado en los servlets de Java[3].

Con JSP se pueden crear aplicaciones web que se ejecuten en variados servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Por tanto, las JSP se pueden escribir con el editor HTML/XML habitual. [17] La principal ventaja de JSP frente a otros lenguajes es que permite integrarse con clases Java lo que permite separar en niveles las aplicaciones web, almacenando en clases java las partes que consumen más recursos así como las que requieren más seguridad, y dejando la parte encargada de formatear el documento HTML en el archivo jsp. Sin embargo JSP no se puede considerar un script al 100% ya que antes de ejecutarse el servidor web compila el script y genera un servlet, por lo tanto se puede decir que aunque este proceso sea transparente para el programador no deja de ser una aplicación compilada. Su ventaja consiste en algo más de rapidez y disponer del API de Java en su totalidad.

**Principales características:**

- Código separado de la lógica del programa.
- Las páginas son compiladas en la primera petición.
- Permite separar la parte dinámica de la estática en las páginas web.
- Los archivos se encuentran con la extensión jsp.
- El código JSP puede ser incrustado en código HTML.

En el lenguaje Java entre las ventajas más comunes que nos brinda su uso podemos encontrar las siguientes:

- Orientado a objetos.
- Distribuido.
- Interpretado y compilado a la vez.
- Robusto.

[3] Los servlets son objetos que corren dentro del contexto de un contenedor de servlets (ej: Tomcat) y extienden su funcionalidad.

- Seguro.
- Indiferente a la arquitectura.
- Portable.
- Multihebra.
- Dinámico.

## Hypertext Preprocessor (PHP).

**PHP** [18]: es un lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas. Es usado principalmente en interpretación del lado del servidor (server-side scripting) pero actualmente puede ser utilizado desde una interfaz de línea de comandos o en la creación de otros tipos de programas incluyendo aplicaciones con interfaz gráfica usando las bibliotecas Qt o GTK+.PHP. Es un acrónimo recursivo que significa **P**HP **H**ypertext **P**re-processor (inicialmente PHP Tools, o, Personal Home Page Tools).

Fue creado originalmente por “Rasmus Lerdorf” en 1994; sin embargo la implementación principal de PHP es producida ahora por “The PHP Group” y sirve como el estándar de facto para PHP al no haber una especificación formal. Publicado bajo la licencia de PHP, la fundación de software libre (Free Software Foundation) considera esta licencia como software libre. Es un lenguaje interpretado de propósito general ampliamente usado y 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, sin costo alguno. PHP se encuentra instalado en más de 20 millones de sitios web y en un millón de servidores.

Este lenguaje libre tiene características que lo convierten en una herramienta ideal para la creación de sitios web dinámicos:

- Es un lenguaje multiplataforma.
- Capacidad de conexión con la mayoría de los manejadores de base de datos que se utilizan en la actualidad, destaca su conectividad con MySQL.

- Capacidad de expandir su potencial utilizando la enorme cantidad de módulos (llamados ext's o extensiones).
- Posee una amplia documentación en su página oficial, entre la cual se destaca que todas las funciones del sistema están explicadas y ejemplificadas en un único archivo de ayuda.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Permite las técnicas de Programación Orientada a Objetos.
- Biblioteca nativa de funciones sumamente amplia e incluida.
- No requiere definición de tipos de variables.
- Tiene manejo de excepciones (desde PHP5).

Por todas estas características y ventajas previamente descritas, por políticas del Área Temática donde el trabajo está inmerso, y además porque el sistema se desplegará en servidores con una arquitectura definida que incluye la utilización del lenguaje programación PHP para el desarrollo de las aplicaciones web, es escogido para la implementación del sistema.

## 1.7 Lenguaje XML.

XML, sigla en inglés de eXtensible Markup Language («lenguaje de marcas extensible»), es un metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Permite definir la gramática de lenguajes específicos (de la misma manera que HTML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. XML no ha nacido sólo para su aplicación en Internet, sino que se propone como un estándar para el intercambio de información estructurada entre diferentes plataformas. Se puede usar en bases de datos, editores de texto, hojas de cálculo y casi cualquier cosa imaginable. Es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil. Es extensible, lo que quiere decir que una vez diseñado un lenguaje y puesto en producción, igual es posible extenderlo con la adición de nuevas etiquetas de manera que los antiguos consumidores de la vieja versión puedan entender el nuevo formato. El

analizador es un componente estándar, no es necesario crear un analizador específico para cada lenguaje. Esto posibilita el empleo de uno de los tantos disponibles. De esta manera se evitan bugs y se acelera el desarrollo de la aplicación. Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarlo. Mejora la compatibilidad entre aplicaciones.

## 1.8 Navegadores Web.

### Mozilla Firefox

**Mozilla Firefox [19]:** es un navegador de Internet libre y de código abierto descendiente de Mozilla Application Suite, desarrollado por la Corporación Mozilla, la Fundación Mozilla y un gran número de voluntarios externos. Cuenta con el 20.78% del mercado de navegadores web a partir de noviembre de 2008, por lo que es el segundo navegador más popular en todo el mundo, después de Internet Explorer.

Para visualizar páginas web, Firefox usa el motor de renderizado Gecko, que implementa algunos estándares web actuales además de otras funciones, algunas de las cuales están destinadas a anticipar probables adiciones a los estándares web.

Algunas de las propiedades de Firefox:

**Portabilidad.**

Usando Mozilla Firefox tenemos la suerte de que este va a muchos de los sistemas operativos existentes, entre ellos: Windows (en cualquiera de sus versiones), Linux (Desde la versión 2.2 del Kernel) y Mac (Desde Mac OS X 10.1.x a la 10.2.x y nuevas versiones). Con esto, cualquier usuario corriente de PC puede usar el navegador sin tener la necesidad de cambiar su sistema operativo.

**Instalación.**

El Proyecto Mozilla trabajó arduo durante la nueva versión del navegador, logrando crear instaladores para Windows, estos instaladores son como los de cualquier otra aplicación, en unos cuantos pasos ya se tiene Firefox funcionando. En Linux también hay un instalador muy simple de utilizar, aunque no es gráfico.

**Velocidad.**

Velocidad, Mozilla Firefox trabaja de forma excelente en computadoras sin hardware muy potente, el programa está diseñado para realizar un bajo consumo de recursos. Firefox se ejecuta en cuestión de segundos, y la aparición de las páginas es muy rápida.

**Cumplimiento de estándares.**

El cumplimiento de los estándares mantenidos por la W3C es una de las principales intenciones de Mozilla Firefox que mostrará las páginas de una manera limpia y su visualización será excelente, si cumplen con los estándares, cosa que no sucede con IE que agranda los textos y no tiene un manejo tan bueno de las cajas. El navegador Mozilla Firefox siempre estará actualizado en lo último en estándares web para con esto lograr una mejor visualización de las páginas web.

**Personalización**

Mozilla Firefox permite la creación de perfiles, manteniendo así las opciones que cada usuario elija. También existen excelentes extensiones y skins de fácil instalación, estos mejoran la usabilidad y el aspecto del navegador, cosa que no se logra en Internet Explorer que siempre permanece con los mismos colores.

**Extensiones y Skins.**

Poder añadirle cosas nuevas a tu navegador es muy interesante, puedes tener hasta un programa de IRC o un cliente de Messenger en el navegador instalando las extensiones pertinentes, las cuales por supuesto están disponibles al público en la página del Proyecto Mozilla.

**Disponibilidad de lenguajes.**

La gran potencia del navegador y el interés de los usuarios permitieron que personas con ánimos de ayudar con el proyecto tradujeran el producto a muchos idiomas.

**Soporte Técnico.**

En este momento dispone de Foros, Lista de correos y otros servicios para la solución de problemas. En MozillaZine hay un foro disponible exclusivamente para dar ayuda referente a Mozilla Firefox.

**Precio.**

Mozilla Firefox es gratis, todo es cuestión de bajarlos de la página del producto aunque también existe la posibilidad de comprar un CD con los productos y en él se incluyen extensiones y algunos otros productos no disponibles a todos los usuarios. Como nota adicional es posible hacer donaciones al equipo Mozilla y así ayudarlos a que se sigan desarrollando.

**Permite extender sus funcionalidades mediante la instalación de plugins.[4]**

En la realización del diseñador de consultas se explotó fuertemente este aspecto ya que se instaló el plugins Firebug que permite encontrar errores y trasear código JavaScripts. Incluye también un corrector ortográfico, búsqueda progresiva, marcadores dinámicos, un administrador de descargas y un sistema de búsqueda integrado que utiliza el motor de búsqueda que desee el usuario. Además se pueden añadir funciones a través de complementos desarrollados por terceros. Este programa es multiplataforma y está disponible en varias versiones de Microsoft Windows, Mac OS X, GNU/Linux y algunos sistemas basados en Unix. La última versión estable es la 3.0.5, publicada el 16 de diciembre de 2008.Su código fuente es software libre, publicado bajo una triple licencia GPL/LGPL/MPL.

Por ello es el navegador web recomendado para la realización de pruebas y posterior despliegue de la propuesta desarrollada.

## Internet Explorer (IE).

Fue creado en 1995 tras la adquisición por parte de Microsoft del código fuente de Mosaic, un navegador desarrollado por Spyglass[5], siendo rebautizado entonces como Internet Explorer. Actualmente es el navegador de Internet más popular y más utilizado en el mundo, rebasando en gran medida a las competencias existentes, aún cuando algunas de éstas han incrementado su popularidad en los últimos años. Su popularidad es debido a que Internet

[4] Un complemento (o plug-in en inglés) es una aplicación que se relaciona con otra para aportarle una función nueva y generalmente muy especifica.

[5] Spyglass fue una compañía de software con central en Champaign. Fundada en 1990, se creó por la Universidad de Illinois. Entre sus principales logros se hallaba el navegador web Mosaic.

Explorer es el navegador oficial de Windows, y viene incluido de fábrica en dicho sistema operativo.[20]

Algunas de las Características de Internet Explorer son:

- **Soporte de estándares:**

  Internet Explorer, utilizando el motor de diseño Trident[6], casi en su totalidad soporta HTML 4.01, CSS 1.0 y XML 1, con pequeñas lagunas de contenido. Soporta parcialmente CSS nivel 2 y DOM Nivel 2, con importantes deficiencias en el contenido y cuestiones de conformidad. El soporte para CSS 2.1 está en el proyectado para Internet Explorer 8, es totalmente compatible con XSLT 1.0[7] y está proyectado para versiones futuras de Internet Explorer.

- **Usabilidad y accesibilidad:**

  Internet Explorer hace uso de la accesibilidad previstas en Windows. También es una interfaz de usuario de FTP, con operaciones similares a los del Explorador de Windows (aunque esta característica requiere una ventana que se abre en las últimas versiones del navegador, en lugar de forma nativa en el navegador). Las versiones recientes bloquean las ventanas emergentes e incluyen navegación por pestañas. La navegación con pestañas también puede ser añadida a las versiones anteriores mediante la instalación en la barra de herramientas de MSN Search o de Yahoo.

- **Caché:**

  Internet Explorer guarda archivos temporales para permitir un acceso más rápido (o el acceso fuera de línea) a páginas visitadas anteriormente. El contenido está indexado en un archivo de base de datos, conocida como Index.dat. Los archivos múltiples que existen son diferentes índices de contenido, contenido visitado, RSS, Autocompletar,

[6] Trident fue diseñado como un componente software que permitía a los desarrolladores de software añadir la funcionalidad de navegación web a sus propias aplicaciones fácilmente.
[7] XSLT o Transformaciones XSL es un estándar de la organización W3C que presenta una forma de transformar documentos XML en otros e incluso a formatos que no son XML

páginas web visitadas, las cookies, etc. Antes de IE7, la limpieza del caché se utilizaba para borrar el índice, pero los archivos no eran eliminados. Esta característica puede ser un riesgo potencial para la seguridad tanto de los individuos como de las empresas. A partir de Internet Explorer 7, tanto el índice de entradas de los archivos como ellos mismos se quitan la memoria caché cuando se borra.

- **Políticas de grupo:**

  Internet Explorer es totalmente configurable mediante directiva de grupo. Los administradores de dominios Windows Server puede aplicar y hacer cumplir una serie de ajustes que afectan a la interfaz de usuario (por ejemplo, deshabilitar elementos de menú y las opciones de configuración individual), así como las características de seguridad tales como: la descarga de archivos, la configuración de la zona, por configuración del sitio, Comportamiento de control ActiveX, y otros. La configuración puede hacerse para cada usuario y para cada máquina. Internet Explorer también soporta autenticación integrada de Windows.

## 1.9 Sistemas gestores de bases de datos.

Los sistemas de gestión de base de datos (SGBD) [21]: son un tipo de software dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan. El propósito general de los sistemas de gestión de base de datos es el de manejar de manera clara, sencilla y ordenada un conjunto de datos que posteriormente se convertirán en información relevante.

Existen un conjunto de objetivos que deben cumplir los sistemas gestores de bases de datos:

- Abstracción de la información.
- Independencia.
- Consistencia.
- Seguridad.
- Integridad.
- Respaldo.

- Control de la concurrencia.
- Manejo de Transacciones.
- Tiempo de respuesta.

## Microsoft SQL Server.

Es un SGBD basado en el lenguaje Transact-SQL,[8] y específicamente en Sybase IQ,[9] capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. [22]

Algunas características de Microsoft SQL Server son:

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

## MySQL.

Es un sistema de gestión de base de datos relacional, multihilo y multiusuario .MySQL como software libre en un esquema de licenciamiento dual. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C [23].

[8] Transact SQL es el lenguaje de programación que proporciona SQL Server para ampliar SQL con los elementos característicos de los lenguajes de programación: variables, sentencias de control de flujo y bucles.
[9] Sybase IQ es un motor de bases de datos altamente optimizado para inteligencia empresarial, desarrollado por la empresa Sybase.

MySQL es propietario y está patrocinado por una empresa privada, que posee el copyright de la mayor parte del código.

Posee ventajas como:

- Soporta la mayoría de los comandos del lenguaje SQL (structured query language), el estándar en bases de datos.
- Acceso a las bases de datos de forma simultánea por varios usuarios y/o aplicaciones.
- Seguridad: En forma de permisos y privilegios, determinados usuarios tendrán permiso para consulta o modificación de determinadas tablas. Esto permite compartir datos sin que peligre la integridad de la base de datos o protegiendo determinados contenidos.
- Portabilidad: SQL es también un lenguaje estandarizado, de modo que las consultas hechas usando SQL son fácilmente portables a otros sistemas y plataformas
- Escalabilidad: es posible manipular bases de datos enormes, del orden de seis mil tablas y alrededor de cincuenta millones de registros, y hasta 32 índices por tabla.
- MySQL está escrito en C y C++ y probado con multitud de compiladores y dispone de Apis[10] para muchas plataformas diferentes.
- Conectividad: es decir, permite conexiones entre diferentes máquinas con distintos sistemas operativos. Es corriente que servidores Linux o Unix, usando MySQL, sirvan datos para ordenadores con Windows, Linux, Solaris, etc. Para ello se usa TCP/IP, tuberías, o sockets Unix.
- Es multihilo, con lo que puede beneficiarse de sistemas multiprocesador.
- Permite manejar multitud de tipos para columnas.
- Permite manejar registros de longitud fija o variable.

[10] Una interfaz de programación de aplicaciones o API es el conjunto de funciones y procedimientos (o métodos, si se refiere a programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

## Oracle

Es un sistema de gestión de base de datos relacional (o RDBMS por el acrónimo en inglés de Relational Data Base Management System), desarrollado por Oracle Corporation [24].

Se considera a Oracle como uno de los sistemas de bases de datos más completos, destacando su:

- Soporte de transacciones.
- Estabilidad.
- Escalabilidad.
- Soporte multiplataforma.

Ha sido criticada por algunos especialistas la seguridad de la plataforma, y las políticas de suministro de parches de seguridad, modificadas a comienzos de 2005 y que incrementan el nivel de exposición de los usuarios. En los parches de actualización provistos durante el primer semestre de 2005 fueron corregidas 22 vulnerabilidades públicamente conocidas, algunas de ellas con una antigüedad de más de 2 años. Sus últimas versiones han sido certificadas para poder trabajar bajo Linux.

Surge a finales de los 70 bajo el nombre de Relational Software a partir de un estudio sobre SGBD (Sistemas Gestores de Base de Datos) de George Koch. Computer World definió este estudio como uno de los más completos jamás escritos sobre bases de datos. Este artículo incluía una comparativa de productos que erigía a Relational Software como el más completo desde el punto de vista técnico. Esto se debía a que usaba la filosofía de las bases de datos relacionales, algo que por aquella época era todavía desconocido.

## PostGreSQL.

Es un sistema de gestión de base de datos relacional orientada a objetos de software libre. Posee varias ventajas como [25]:

- Ágil navegación y administración de base de datos.

- Administración sencilla de todos los objetos PostgreSQL.
- Herramientas de manipulación de datos avanzada.
- Administración efectiva de seguridad.
- Excelentes Herramientas visuales y de texto para elaboración de consultas.
- Acceso al servidor PostgreSQL a través del protocolo HTTP.
- Conexión vía reenvío de puerto local a través del túnel SSH[11].
- Impresionantes opciones de exportación e importación de datos.
- Poderoso diseñador visual de base de datos.
- Asistentes fáciles de usar que realizan mantenimiento de tareas de PostgreSQL.

## 1.10 Herramientas de Modelado.

### Rational Rose Enterprise Edition.

Rational Rose Enterprise Edition [26]: es una herramienta que permite el diseño detallado del software y la generación de código fuente (de programas y bases de datos) e ingeniería inversa (obtención del diseño a partir del código fuente), basado en modelos con soporte UML[12]. Es una forma de ayuda para la comprensión del sistema y de sus distintos componentes. Su característica más significativa consiste en la creación de componentes, que contengan una serie de archivos dentro de los cuales se encuentran las distintas clases pertenecientes a dicho componente.

### Visual Paradigm (VP).

Es una herramienta diseñada para desarrollar software con programación orientada a objetos. Busca reducir la duración del ciclo de desarrollo, brindando ayuda a arquitectos, analistas, diseñadores y desarrolladores; permite el uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación y tiene capacidades de ingeniería directa e inversa. Usa UML como lenguaje de modelado. Presenta una interfaz de uso

[11] SSH (Secure SHell, en español: intérprete de órdenes seguro) es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquinas remotas a través de una red. Permite manejar por completo la computadora mediante un intérprete de comandos.

[12] Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad.

intuitiva y con muchas facilidades a la hora de modelar los diagramas que soportan la Ingeniería de Software. Una de las características más importantes de su uso es que brinda la posibilidad de sincronización del modelo de diseño y el código en todo el ciclo de desarrollo una vez que se integra con el Eclipse, permitiendo la facilidad de programar directamente sobre el código fuente generado y a su vez actualizar el diseño con cambios que se realicen en la programación. Tiene una alta disponibilidad de múltiples versiones para cada necesidad, al igual que en las plataformas Linux, MacOS y Windows. Presenta un innovador analizador textual donde se introduce texto extraído de conversaciones con el cliente, se definen actores, casos de uso, entidades, disponibles para la generación de artefactos posteriores. Así mismo, posee una herramienta de generación de reportes en formato PDF o HTML configurable y selectiva, se integra con entornos como Eclipse y Subversion, e importa o exporta formatos estándares de otras herramientas CASE como el Rational Rose. [27]

## 1.11 Entornos de Desarrollo Integrados (IDE) de Programación.

### Notepad++.

Notepad++ es un poderoso editor de código fuente que soporta diversos lenguajes de programación. Este interesante editor permite imprimir el código escrito con los diferentes colores de la sintaxis, posee la opción de autocompletado con la posibilidad de definir una API personalizada, permite la edición de varios documentos al mismo tiempo, con sus correspondientes vistas separadas o integradas a la misma interfaz. Notepad++ posee soporte para búsqueda y reemplazo de expresiones regulares, detección automática del estado del texto, incluye una herramienta de Zoom, trabaja con puntos de marca para editar y desplazarse más dinámicamente y permite la creación de marcos con su correspondiente secuencia de teclas para acceso rápido. [28] Con Notepad++ se puede programar en los siguientes lenguajes: C, C++, Java, C#, XML, HTML, PHP, CSS, makefile, ASCII art (.nfo), doxygen, ini file, batch file, Javascript, ASP, VB/VBS, SQL, Objective-C, RC resource file, Pascal, Perl, pitón, Lua, TeX, TCL, Assembler, Ruby, Lisp, Scheme, Properties, Diff, Smalltalk, Postscript, VHDL, Ada, Caml, AutoIt, KiXtart, Matlab, Verilog, Haskell, InnoSetup y CMake. Es un excelente editor para tener en cuenta. [29].

## Zend Studio for Eclipse 6.0.

Zend Studio for Eclipse 6.0 es un IDE de desarrollo que soporta varios lenguajes aunque fue construido especialmente para PHP. Los expertos en PHP consideran a Zend Studio como el entorno IDE más maduro y con más características útiles. Se trata de un programa de la casa Zend, impulsores de la tecnología de servidor PHP, orientada a desarrollar aplicaciones web, en lenguaje PHP. El programa, además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código. Está escrito en Java, lo que a veces supone que no funcione tan rápido como otras aplicaciones de uso diario. Sin embargo, esto ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para Windows, Linux y MacOS[13]. Zend Studio consta de dos partes en las que se dividen las funcionalidades de parte del cliente y las del servidor. Las dos partes se instalan por separado, la del cliente contiene el interfaz de edición y la ayuda. Permite además hacer depuraciones simples de scripts, aunque para disfrutar de toda la potencia de la herramienta de depuración habrá que disponer de la parte del servidor, que instala Apache y el módulo PHP o, en caso de que estén instalados, los configura para trabajar juntos en depuración. [30]

Se ha convertido en el IDE más potente en el mercado del software para PHP, ofrece al desarrollador profesional en PHP la potencia del Zend Studio y el soporte multilenguaje de Eclipse y su enorme conjunto de extensiones (plugins) .Con el empleo de esta herramienta instalando el spket para completamiento de código JavaScripts se hace mucho más sencillo el trabajo con ficheros JavaScript.

En concreto, Zend Studio para Eclipse posee las siguientes características: [31]

- No requiere la instalación previa de PHP ni del entorno de ejecución de Java.
- Soporte para PHP 4 y PHP 5.
- Resaltado de sintaxis, autocompletado de código, ayuda de código y lista de parámetros de funciones y métodos de clase.
- PHPDoc integrado.

[13] Mac OS X es una línea de sistemas operativos computacionales desarrollada, comercializada y vendida por Apple Inc.

- Plegado de código (comentarios, bloques de phpDoc, cuerpo de funciones y métodos e implementación de clases).
- Inserción automática de paréntesis y corchetes de cierre.
- Sangrado automático y otras ayudas de formato de código.
- Emparejamiento (matching) de paréntesis y corchetes (si se sitúa el cursor sobre un paréntesis (corchete) de apertura (cierre), Zend Studio localiza el correspondiente paréntesis (corchete) de cierre (apertura)).
- Detección de errores de sintaxis en tiempo real.
- Funciones de depuración: Botón de ejecución y traza, marcadores, puntos de parada (breakpoints), seguimiento de variables y mensajes de error del intérprete de PHP. Permite también la depuración en servidores remotos (requiere Zend Platform).
- Instalación de barras de herramientas para Internet Explorer y Mozilla Firefox (opcional).
- Soporte para gestión de grandes proyectos de desarrollo.
- Manual de PHP integrado.
- Soporte para control de versiones usando CVS o Subversion (a elección del desarrollador).
- Cliente FTP integrado.
- Soporte para navegación en bases de datos y ejecución de consultas SQL.

Zend Studio fue diseñado para usarse con el lenguaje PHP; sin embargo ofrece soporte básico para otros lenguajes web, como HTML, Javascript y XML.

## Visual Studio 2008.

Visual Studio 2008 fue publicado el 17 de Noviembre de 2007 en inglés, mientras que la versión en castellano no fue publicada hasta el 2 de Febrero de 2008.El nuevo framework (.Net 3.5) está diseñado para aprovechar las ventajas que ofrece el nuevo sistema operativo "Windows Vista" a través de sus subsistemas "Windows Communication Foundation" (WCF) y "Windows Presentation Foundation" (WPF).El primero tiene como objetivo la construcción de aplicaciones orientadas a servicios mientras que el último apunta a la creación de interfaces de usuario más dinámicas que las conocidas hasta el momento.

A las mejoras de desempeño, escalabilidad y seguridad con respecto a la versión anterior, se agregan entre otras, las siguientes novedades.

• La mejora en las capacidades de Pruebas Unitarias permiten ejecutarlas más rápido independientemente de si lo hacen en el entorno IDE o desde la línea de comandos. Se incluye además un nuevo soporte para diagnosticar y optimizar el sistema a través de las herramientas de pruebas de Visual Studio. Con ellas se podrán ejecutar perfiles durante las pruebas para que ejecuten cargas, prueben procedimientos contra un sistema y registren su comportamiento; y utilizar herramientas integradas para depurar y optimizar.

• Con Visual Studio Tools para Office (VSTO) integrado con Visual Studio 2008 es posible desarrollar rápidamente aplicaciones de alta calidad basadas en la interfaz de usuario (UI) de Office que personalicen la experiencia del usuario y mejoren su productividad en el uso de Word, Excel, PowerPoint, Outlook, Visio, InfoPath[14] y Project. Una completa compatibilidad para implementación con ClickOnce[15] garantiza el entorno ideal para una fácil instalación y mantenimiento de las soluciones Office.

• Visual Studio 2008 permite incorporar características del nuevo Windows Presentation Foundation sin dificultad tanto en los formularios de Windows existentes como en los nuevos. Ahora es posible actualizar el estilo visual de las aplicaciones al de Windows Vista debido a las mejoras en Microsoft Foundation Class Library (MFC) y Visual C++. Visual Studio 2008 permite mejorar la interoperabilidad entre código nativo y código manejado por .NET. Esta integración más profunda simplificará el trabajo de diseño y codificación.

• LINQ (Language Integrated Query) es un nuevo conjunto de herramientas diseñado para reducir la complejidad del acceso a Base de Datos, a través de extensiones para C++ y Visual Basic así como para Microsoft .NET Framework. Permite filtrar, enumerar, y crear proyecciones de muchos tipos y colecciones de datos utilizando todas las mismas sintaxis, prescindiendo del uso de lenguajes especializados como SQL.

[14] Microsoft Office InfoPath es una aplicación usada para desarrollar formularios de entrada de datos basados en XML.
[15] ClickOnce es una nueva tecnología de implementación que hace que la tarea de implementar una aplicación basada en Formularios Windows sea tan fácil como si fuera una aplicación Web.

• Visual Studio 2008 ahora permite la creación de soluciones multiplataforma adaptadas para funcionar con las diferentes versiones de .Net Framework: 2.0. (Incluido con Visual Studio 2005), 3.0 (incluido en Windows Vista) y 3.5 (incluido con Visual Studio 2008).

• .NET 3.5 incluye biblioteca ASP.NET AJAX para desarrollar aplicaciones web más eficientes, interactivas y altamente personalizadas que funcionen para todos los navegadores más populares y utilicen las últimas tecnologías y herramientas web. [32]

## 1.12 Frameworks

### Ext 2.2

Ext: es un framework JavaScript del lado del cliente para el desarrollo de aplicaciones web. Tiene un sistema dual de licencia: Comercial y Open Source (Código Abierto). Este framework puede correr en cualquier plataforma que pueda procesar POST y devolver datos estructurados (PHP, Java, .NET y algunas otras) en tiempo de ejecución carga y crea todos los objetos html a través del uso intenso de DOM. Los datos son obtenidos mediante mensajes AJAX a través de XML y/o JSON. El diseño está completamente separado de la funcionalidad. Presenta funciones comunes como validación, combobox editables, ventanas deslizables y grillas editables. Se determinó utilizar para la implementación de la interfaz de usuario este framework en su versión estable Ext.2.2, por las facilidades que desde el punto de vista del diseño brinda y lo amigable que resulta la interfaz resultante para el usuario final.

### Open-jACOB Draw2D

Open-jACOB Draw2D**:** es un framework Javascript que permite crear gráficos y diagramas. Su interfaz da la posibilidad de dibujar directamente desde su navegador, sin necesidad de instalar ningún software adicional. Consta de un editor de marco de trabajo (Workflow) en el cual se pueden insertar los distintos componentes específicos del diagrama que se desee desarrollar .Se determino utilizar este framework para la implementación del marco de trabajo del Diseñador de Consultas en su versión estable draw2d_0 [1].9.17, por la facilidades que desde el punto de vista del diseño web para modelar datos nos proporciona, diseño de tablas, las relaciones entre ellas, marco de trabajo.

## AJAX

**AJAX** es la unión de varias tecnologías que juntas pueden lograr cosas realmente impresionantes, incorporando: presentación basada en estándares usando XHTML y CSS; exhibición e interacción dinámicas usando el Document Object Model; intercambio y manipulación de datos usando XML and XSLT; recuperación de datos asincrónica usando XMLHttpRequest; y Javascript para unir todo el contenido. AJAX, en resumen, es el acrónimo para Asynchronous Javascript + XML y el concepto es: Cargar y renderizar una página, luego mantenerse en esa página mientras scripts y rutinas van al servidor buscando los datos que son usados para actualizar la página solo re-renderizando la página y mostrando u ocultando porciones de la misma. Ofrece 10 ventajas que se pueden denominar razones para usar estas tecnologías y que provocan que AJAX sea la palabra de moda en la comunidad desarrolladora de aplicaciones web.

**Basado en los estándares abiertos.**

Ajax está formado por las tecnologías Javascript, HTML, XML, CSS, y XML HTTP Request Object, siendo este último el único que "no es" estándar pero es soportado por los navegadores más utilizados de internet como son los basados en Mozilla, Internet Explorer, safari y opera.

**Usabilidad.**

Permite a las páginas hacer una pequeña petición de datos al servidor y recibirla sin necesidad de cargar la página entera. El incremento de las actualizaciones "on the fly" elimina el tener que refrescar el navegador, algo bastante apreciado a la hora de operar en una aplicación web.

**Válido en cualquier plataforma y navegador.**

Internet Explorer, los basados en Mozilla y Firefox son los que se llevan la palma en el mercado de internet y además son los navegadores en los que es más fácil programar aplicaciones web AJAX, pero ahora es posible construir aplicaciones web basadas en AJAX para que funcionen en los navegadores más modernos. Es una de las razones más

importantes por las que AJAX se ha vuelto tan popular. Aunque si bien muchos desarrolladores sabían que era posible usarse años atrás con Internet Explorer, no era viable realizarse. Ahora ya es posible su avance gracias a Mozilla y Firefox.

**Beneficia las aplicaciones web.**

AJAX es la cara del presente en las aplicaciones web - las aplicaciones web conllevan ciertos beneficios sobre las aplicaciones sobre escritorio (aplicaciones que dependan de un sistema operativo, librerías, lo que se entiende por programas compilados). Esto incluyó un menor coste de creación, facilidad de soporte y mantenimiento, menor tiempo a la hora de desarrollarlas, y sin necesidad de instalaciones; éstas son algunos de los beneficios llevados a las empresas y usuarios el adoptar aplicaciones web desde mediados de los 90. AJAX solo ayudará a las aplicaciones web a mejorar y conseguir un mejor resultado de cara al usuario final.

**No es difícil su utilización.**

Porque AJAX está basada en los estándares que han sido utilizados durante muchos años, muchos desarrolladores web han tenido que utilizar las tecnologías que las aplicaciones requieren. Esto significa que no es un gran esfuerzo el aprendizaje de los desarrolladores el pasar de un simple código HTML y aplicaciones web a una potente aplicación AJAX. También significa que los desarrolladores pueden actualizar poco a poco las interfaces de usuario hacia unas interfaces con AJAX; no necesita una re-escritura de la aplicación entera, se puede hacer incrementalmente.

**Compatible con Flash.**

Muchos desarrolladores tienen serias dudas sobre usar Flash o AJAX. Definitivamente hay ventajas y desventajas en ambas tecnologías según la situación que se dé, pero también hay muchas posibilidades y muy buenas para que ambas funcionen en conjunto.

**Adoptado por los "gordos" de la tecnología web.**

La difusión de AJAX en los líderes de la industria de internet prueba que el mercado acepta y valida el uso de esta tecnología. Todo el mundo está migrando hacia AJAX incluyendo

Google, Yahoo, Amazon, Microsoft (por nombrar unos pocos). Google Maps fue lo que captó la atención de los desarrolladores web. Cuando empezaron a investigar como google era capaz de llevar esa increíble herramienta dentro de un navegador sin necesidad de ningún tipo de plug-in, encontraron que AJAX estaba detrás del tema.

**Es independiente del tipo de tecnología de servidor que se utilice.**

Así como AJAX funciona en cualquier navegador, es perfectamente compatible con cualquier tipo de servidor estándar y lenguaje de programación web. PHP, ASP. ASP.Net, Perl, JSP. El ser completamente compatible al desarrollo en estas tecnologías, ha ayudado a AJAX a su auge.

**Mejora la estética de la web.**

Con AJAX se puede interactuar la imaginación del desarrollador con la usabilidad de una aplicación web de forma que se pueda realizar una aplicación que si no estuviera dentro de un navegador, podría pasar por una aplicación normal de escritorio. [33]

## Symfony.

Symfony es un framework para construir aplicaciones web con PHP. En otras palabras, Symfony es un enorme conjunto de herramientas y utilidades que simplifican el desarrollo de las aplicaciones web. Este emplea el tradicional patrón de diseño MVC (modelo-vista-controlador) para separar las distintas partes que forman una aplicación web. El modelo representa la información con la que trabaja la aplicación y se encarga de acceder a los datos. [30] La vista transforma la información obtenida por el modelo en las páginas web a las que acceden los usuarios. El controlador es el encargado de coordinar todos los demás elementos y transformar las peticiones del usuario en operaciones sobre el modelo y la vista. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. [34] Symfony está desarrollado completamente con PHP 5. Ha sido probado en numerosos proyectos reales y se utiliza en sitios web de comercio

electrónico de primer nivel, es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas Unix, Linux, etc. como en plataformas Windows. [35]

### Propel

Propel es un framework para el mapeo objeto-relacional escrito en PHP 5. Este permite el acceso a las bases de datos a través de un grupo de objetos proporcionando una API sencilla para almacenar y recuperar información. Con el uso de Propel los desarrolladores de sistemas web trabajan con las bases de datos de igual forma que con otras clases y objetos de PHP 5. Este framework brinda herramientas por línea de comandos para la generación automática de la capa de acceso a datos a partir de una base de datos e incluso es posible generar la base de datos a partir de una descripción en formato XML o YML del modelo de datos. Debido a que se utiliza Creole como capa de abstracción de bases de datos Propel es capaz de interactuar con varios sistemas gestores de bases de datos. Para utilizar dichas herramientas se necesita configurar el fichero propel.ini, el cual contiene la configuración que cargarán las herramientas de Propel. Luego de hacer las configuraciones donde se debe especificar el tipo de gestor de bases de datos, la dirección del servidor, un usuario y una contraseña válida así como la base de datos a mapear el framework puede ser empleado.

### Creole

Creole es una capa de abstracción de bases de datos para PHP 5 que abstrae la API específica nativa de PHP para cada gestor creando un código mucho más portable y brindando a los desarrolladores una interfaz completamente orientada a objetos. Fue creado originalmente como un subproyecto de Propel para resolver el problema de que ninguna de las capas de abstracción disponibles satisfacía las necesidades de éste.

## 1.13 Conclusiones.

En el presente capítulo, después de haber hecho un análisis de las tendencias, metodologias de desarrollo, tecnologías, lenguajes de programación y gestores de bases de datos; se selecciona la metodología OpenUP, como guía para la documentación del software

propuesto, ya que es apropiada para proyectos pequeños ,que no disponen de mucho tiempo para su desarrollo y de bajos recursos ,permite disminuir las probabilidades de fracaso en los proyectos e incrementar las probabilidades de éxito y detectar errores tempranos a través de un ciclo iterativo. Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP y por ser una metodología ágil tiene un enfoque centrado al cliente y con iteraciones cortas. Se decidió utilizar el servidor web Apache 2.0, el cual es libre de código abierto y bajo la licencia Apache/GPL Como lenguaje de programación del lado del servidor se empleará PHP en su versión 5, como gestor de bases de datos se usará PostGreSQL en la versión 8.2.4 por las características mostradas anteriormente; además por las siguientes razones: PHP lo maneja de forma fácil debido a la gran cantidad de funciones que tiene explícitas. Es multiplataforma y hasta su versión 8.2.4 no tiene precio en el mercado, se adquiere libremente. Para la elaboración de la aplicación, se utilizará el IDE de programación Zend Studio 6.0 por las grandes prestaciones que este brinda y el Notpad++. El sistema además está implementado usando el framework de clases de PHP Symfony que a su vez emplea Propel y Creole para gestionar el acceso a datos y de Javascript Ext 2.2 y Openjacop, se hace rehúso de todas las librerías y clases que estos tres potentes frameworks ofrecen. El lenguaje empleado para la trasmisión de datos cliente-servidor es XML y se utilizará el lenguaje CSS para definir el aspecto de cada elemento: color, tamaño y tipo de letra del texto, separación horizontal y vertical entre elementos, posición de cada elemento dentro de la página. Se recomienda la utilización de Mozilla Firefox como navegador web

La selección expuesta se ha realizado teniendo en cuenta además la arquitectura propuesta para el Módulo Diseñador de Consultas elaborada por el grupo de arquitectura de la línea de herramientas integrales del CENTALAD.

# Capítulo 2 Elementos de Arquitectura.

## 2.1 Introducción

En este capítulo se hace una valoración de las arquitecturas escogidas para la realización del sistema, de cada una se hace un análisis de por qué es apropiada y se exponen los criterios o argumentos necesarios de su selección. Se da una explicación de los patrones de diseño más usados en la realización del sistema, se describe el estándar de codificación y se explican los procesos fundamentales que intervienen en la construcción del mismo.

## 2.2 Arquitectura de Software.

Una Arquitectura de Software, también denominada Arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema. La Arquitectura de Software establece los fundamentos para que analistas, diseñadores y programadores trabajen en una línea común que permita alcanzar los objetivos del sistema. Se selecciona y diseña con base en objetivos y restricciones. Los objetivos son aquellos prefijados para el sistema, pero no solamente los de tipo funcional, también otros objetivos como la mantenibilidad, auditabilidad, flexibilidad e interacción con otros sistemas de información. Las restricciones son aquellas limitaciones derivadas de las tecnologías disponibles para la implementación del sistema. Unas arquitecturas son más recomendables implementar con ciertas tecnologías mientras que otras tecnologías no son aptas para determinadas arquitecturas. La arquitectura de software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación ente ellos. Toda arquitectura debe ser implementable en una arquitectura física, que consiste simplemente en determinar qué computadora tendrá asignada cada tarea.

La arquitectura de software, tiene que ver con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requerimientos no funcionales, como la confiabilidad, escalabilidad, portabilidad, y disponibilidad. La

Arquitectura de Software representa un alto nivel de abstracción común que la mayoría de los participantes, si no todos, pueden usar como base para crear entendimiento mutuo, formar consenso y comunicarse entre sí. En sus mejores expresiones, la descripción arquitectónica expone las restricciones de alto nivel sobre el diseño del sistema, así como la justificación de decisiones arquitectónicas fundamentales. [36]

## 2.3 Requisitos no funcionales del sistema propuesto.

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. En muchos casos los requerimientos no funcionales son fundamentales en el éxito del producto. Y normalmente están vinculados a requerimientos funcionales, es decir, una vez se conozca lo que el sistema debe hacer se puede determinar cómo ha de comportarse, qué cualidades debe tener o cuán rápido o grande debe ser. Los requerimientos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con toda la funcionalidad requerida, las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación.

A continuación se muestran los requerimientos no funcionales correspondientes al sistema en cuestión:

**Interfaz externa.**

La interfaz de usuario será sencilla, amigable, atractiva y de fácil navegación por el usuario, con el objetivo de obtener un entorno visual agradable para el mismo. Se seleccionará un esquema de colores a la vez atractivo. Presentará un diseño perfectamente encuadrado para resoluciones de 1024 x 768, pero preparado para verse en otras resoluciones.

**Usabilidad.**

La aplicación web será flexible y de fácil aprendizaje, pues se trata en todo lo posible de mantener un estándar de operabilidad que logre que las interacciones del usuario con el sistema sean sugerentes y familiares. La misma podrá ser usada por cualquier persona que posea conocimientos básicos en el manejo de la computadora y de un ambiente web en sentido general. Deberá visualizarse bien en los principales navegadores que existen en el mundo y estará disponible en todo momento.

**Rendimiento.**

Las interfaces estarán poco cargadas de imágenes y contenido para garantizar que el tiempo de ejecución de los hipervínculos, las adiciones, modificaciones y eliminaciones no excedan los 3 segundos y garantizar de esta manera una respuesta rápida del sistema.

**Soporte.**

El sistema contará con una ayuda detallada para el usuario con la cual podrá aprender rápidamente a utilizar la aplicación web y estará bien documentado para garantizar futuros mantenimientos.

**Portabilidad.**

El sistema podrá ejecutarse sobre plataforma Linux, Windows 98 o superior.

**Seguridad.**

La autentificación de los usuarios en el sistema, será garantizada por el Sistema de Autentificación, Autorización y Auditoría (SAAA) implementado para el ERP Cubano al cual estará conectada la aplicación.

- Se debe restringir las funcionalidades mediante roles de usuarios garantizando que la información sea accesible al usuario autorizado.
- Las consultas podrán ser diseñadas en dependencia del rol que ocupe el usuario en el sistema.
- El sistema deberá estar protegido contra accesos no autorizados y las modificaciones de información.

**Confiabilidad.**

Todas las partes del diseño del sistema serán realmente aplicadas y se hará una transformación correcta del diseño en un lenguaje de programación. Además deberá prevenir los posibles fallos y/o errores que pudieran presentarse así como posibilitar una rápida recuperación en dichos casos.

**Software Para el cliente:**

- Navegador web, recomendado: Mozilla 1.5 o una versión superior.
- Sistema operativo Linux o Windows 98 ó Superior.

**Para el servidor:**

- Sistema operativo Linux.
- Servidor web Apache 2.0 y PHP 5.
- Framework Symfony.
- Servidor de Base de Datos PostGreSQL.

**Hardware PC clientes:**

- Procesador Pentium III o superior.
- 256 de memoria RAM o superior.
- Monitor VGA o superior.
- Tarjeta de red.

**Servidor:**

- Procesador Pentium IV o superior.
- 512 de memoria RAM o superior.
- Disco Duro de 80 GB.

**Implementación.**

Se utilizará como metodología de desarrollo OpenUP (Open Unified Process) y como herramienta case el Visual Paradigm.

Para la implementación se utilizará como lenguaje de programación PHP y Java Script y como IDE de desarrollo Zend Studio 6.0.

**Ayuda y Documentación en línea.**

Contará con un manual de usuario para que se pueda explotar al máximo y con una ayuda digital, a la cual se podrá acceder desde cualquier parte de la aplicación.

## 2.4 Patrones o estilos arquitectónicos presentes en la solución. Justificación de su uso y características.

### Modelo Cliente/Servidor.

La arquitectura Cliente/Servidor es un modelo 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. Los clientes realizan generalmente funciones como:

- Manejo de la interfaz de usuario.
- Captura y validación de los datos de entrada.
- Generación de consultas e informes sobre las bases de datos.

Por su parte los servidores realizan las siguientes funciones:

- Gestión de periféricos compartidos.

- Control de accesos concurrentes a bases de datos compartidas.
- Enlaces de comunicaciones con otras redes de área local o extensa.

Siempre que un cliente requiere un servicio lo solicita al servidor correspondiente y éste le responde proporcionándolo. Normalmente, pero no necesariamente, el cliente y el servidor están ubicados en distintos procesadores. Los clientes se suelen situar en ordenadores personales y/o estaciones de trabajo y los servidores en procesadores departamentales o de grupo. Entre las principales características de la arquitectura Cliente/Servidor se pueden destacar las siguientes: [37]

- El servidor presenta a todos sus clientes una interfaz única y bien definida.
- El cliente no necesita conocer la lógica del servidor, sólo su interfaz externa.
- El cliente no depende de la ubicación física del servidor, ni del tipo de equipo físico en el que se encuentra, ni de su sistema operativo.
- Los cambios en el servidor implican pocos o ningún cambio en el cliente.

**Ventajas:**

- Menores costes de operación:
    - Permiten un mejor aprovechamiento de los sistemas existentes, protegiendo la inversión. Por ejemplo, la compartición de servidores (habitualmente caros) y dispositivos periféricos (como impresoras) entre máquinas clientes permite un mejor rendimiento del conjunto.
    - Proporcionan un mejor acceso a los datos. La interfaz de usuario ofrece una forma homogénea de ver el sistema, independientemente de los cambios o actualizaciones que se produzcan en él y de la ubicación de la información.
    - El movimiento de funciones desde un ordenador central hacia servidores o clientes locales origina el desplazamiento de los costes de ese proceso hacia máquinas más pequeñas y por tanto, más baratas.
- Mejora en el rendimiento de la red:

- La arquitectura Cliente/Servidor elimina la necesidad de mover grandes bloques de información por la red hacia los ordenadores personales o estaciones de trabajo para su proceso. Los servidores controlan los datos, procesan peticiones y después transfieren sólo los datos requeridos a la máquina cliente. Entonces, la máquina cliente presenta los datos al usuario mediante interfaces amigables. Todo esto reduce el tráfico de la red, lo que facilita que pueda soportar un mayor número de usuarios.
- Tanto el cliente como el servidor pueden escalarse para ajustarse a las necesidades de las aplicaciones.
- En una arquitectura como ésta, los clientes y los servidores son independientes los unos de los otros con lo que pueden renovarse para aumentar sus funciones y capacidad de forma independiente, sin afectar al resto del sistema.

En el caso del módulo Diseñador de Consultas se utiliza este modelo por las ventajas mencionadas anteriormente y que la aplicación estará ubicada en un servidor central, el cual se encargará de dar respuesta a las peticiones realizadas por los clientes que la utilicen.

## Arquitectura en capas.

El modelo actual de desarrollo ha demostrado que organizar los elementos de las aplicaciones en componentes independientes puede lograr una mayor eficiencia durante el tiempo de desarrollo y mantenimiento. La programación en múltiples capas es la técnica más efectiva para aplicaciones empresariales, dividir los componentes de la aplicación en capas implica una fácil administración y rapidez en entornos Cliente/Servidor. Esta arquitectura consiste en dividir los componentes primarios de una aplicación, programarlos por separado y después unirlos, ya sea en tiempo de diseño o de ejecución. Las aplicaciones web se modelan mediante lo que se conoce como arquitecturas de capas:

**Definición:** Una capa representa un elemento del sistema que procesa o trata la información.

**Características:** Una capa puede residir (se ejecuta) en una máquina diferente o en diferentes espacios o entornos de proceso dentro de la misma máquina.

Existen varias razones que inclinan a utilizar esta arquitectura en el desarrollo de aplicaciones:

- Abstracción total acerca del origen de datos.

Las distintas capas se especializan solamente en la funcionalidad que deben brindar (procesamiento de las reglas del negocio, presentación de los datos y acceso a los mismos) sin importar cuál es el origen de los datos procesados.

- Bajo costo de desarrollo y mantenimiento de las aplicaciones.

La utilización de esta arquitectura deja observar al momento de diseño una mayor carga en complejidad, sin embargo, la utilización de la misma brinda un control más cercano de cada componente, así como también la posibilidad de una verdadera reutilización de código. La modificación de una de las capas puede realizarse con un mínimo de impacto en las demás.

- Estandarización de las reglas del negocio.

Las reglas del negocio se encuentran en una librería común y pueden ser llamadas desde diversas aplicaciones sin saber cómo funciona o ha sido diseñada.

- Mejor calidad de las aplicaciones.

Como las aplicaciones son construidas en unidades separadas, estas pueden ser probadas independientemente y con mucho más detalle, el desarrollador (o los desarrolladores) solo debe preocuparse por el funcionamiento de la capa que está desarrollando, esto conduce a obtener un producto mucho más sólido.

- Mejor funcionamiento de las aplicaciones.

Las capas pueden ejecutarse en máquinas independientes, que se traduce en mejor tiempo de respuesta a los usuarios.

La arquitectura más utilizada es la arquitectura de tres capas. Separa los datos de una aplicación, la interfaz de usuario y la lógica en tres componentes distintos.

**Capas o niveles.**

- **Capa de presentación.**

Es la capa que ve el usuario, presenta el sistema al usuario, le comunica la información y captura la información de este dando un mínimo de proceso (realiza un filtrado previo para comprobar que no hay errores de formato). Esta capa se comunica únicamente con la capa de negocio. La capa de Presentación provee a la aplicación con una interfaz de usuario (IU). Aquí es donde la aplicación presenta información a los usuarios y acepta entradas o respuestas del usuario para usar por el programa. Idealmente, la interfaz de usuario no desarrolla ningún procesamiento de negocios o reglas de validación de negocios. Por el contrario, la IU debería relegar sobre la capa de negocios para manipular estos asuntos. Esto es importante, especialmente hoy en día, debido a que es muy común para una aplicación tener múltiples IU, o para sus clientes o usuarios, que le solicitan que elimine una IU y la remplace con otra.

- **Capa de negocio.**

Es donde reside la lógica de funcionamiento de la aplicación o lógica del negocio, recibiendo las peticiones del usuario y enviando las respuestas tras el proceso. Se denomina capa de negocio pues es aquí donde se establecen todas las reglas que deben cumplirse. Esta capa se comunica con la capa de presentación para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar al gestor de base de datos recuperación o almacenamiento de datos.

- **Capa de datos.**

Es donde residen los datos. Está formada por uno o más gestores de base de datos que realizan todo el almacenamiento de los mismos, reciben solicitudes de almacenamiento y recuperación de información desde la capa de negocio. Todas estas capas pueden residir en un único servidor, pero lo más usual es que las capas estén en ordenadores diferentes, o agrupadas en dependencia de las necesidades de la empresa o de la aplicación.

**Ventajas de una aplicación en tres capas.**

- Las llamadas de la interfaz del usuario en la estación de trabajo, al servidor de capa intermedia, son más flexibles que en el diseño de dos capas, ya que la estación sólo necesita transferir parámetros a la capa intermedia.
- Con la arquitectura de tres capas, la interfaz del cliente no es requerida para comprender o comunicarse con el receptor de los datos. Por lo tanto, esa estructura de los datos puede ser modificada sin cambiar la interfaz del usuario en la PC.
- El código de la capa intermedia puede ser reutilizado por múltiples aplicaciones si está diseñado en formato modular. Esto puede reducir los esfuerzos de desarrollo y mantenimiento, así como los costos de migración.
- La separación de roles en tres capas, hace más fácil reemplazar o modificar una capa sin afectar a los módulos restantes.
- Separando la aplicación de la base de datos, hace más fácil utilizar nuevas tecnologías de agrupamiento y balance de cargas.
- Separando la interfaz del usuario de la aplicación, libera de gran procesamiento a la estación de trabajo y permite que las actualizaciones de la aplicación sean centralizadas en el servidor de aplicaciones.

En el caso del Módulo Diseñador de Consultas, está organizado en 6 capas, que constituye una variante del estilo 3 capas. En la actualidad, no es usual encontrarse tan solo 3 capas, pues se agrega el modelo de datos que normalmente se encuentra en un servidor de bases de datos, o la capa lógica se divide en dos (explorador y servidor web) en el caso de las aplicaciones web.

Capas en las que se encuentra organizado el módulo:

1. La capa de Presentación es la encargada del manejo de las interfaces para la interacción con el sistema. Está conformada sobre los frameworks ExtJS 2.2 y OpenJacob para el desarrollo de aplicaciones en Javascript, lo que le aporta mayor usabilidad y productividad en el desarrollo.

2. La capa de las Vistas incluye la lógica para decidir que debe ver el usuario, los caminos de navegación y cómo interpretar las entradas de los mismos. Esta capa es la encargada de interactuar con la Presentación y suministrarles los datos necesarios a visualizar.
3. El Controlador Frontal es el encargado de atender las peticiones al servidor, es el punto de entrada a la aplicación y su principal función es invocar la acción correspondiente para cada solicitud.
4. La capa de Abstracción de Datos es donde se almacenan todas las entidades del negocio desde un punto de vista orientado a objetos, dentro del MVC representa una parte del Modelo, en la solución al utilizarse Propel como ORM.
5. La capa de Acceso a Datos es la encargada de interactuar con la capa Modelo de Datos para recibir, insertar, actualizar y eliminar información. Es importante destacar que la Capa de Acceso a Datos no es la encargada de administrar o almacenar los datos, esta meramente provee la interfaz entre la lógica del negocio y la base de datos. Se utiliza Creole para la generación de esta capa y para los accesos al Modelo de Datos.
6. El Modelo de Datos es la principal capa que se encarga de la creación, recepción actualización y eliminación de los datos de forma física, soportado en la solución por PostgreSQL 8.3.

Se emplea esta arquitectura debido a las ventajas que posee y a que se logra separar la lógica de presentación de la lógica del negocio, aunque esta última se encuentra un tanto acoplada a la lógica de acceso a datos, sin embargo se logra una gran abstracción en cuanto al Sistema Gestor de Base Datos a utilizar.

## Arquitectura basada en componentes.

La complejidad de los sistemas computacionales actuales ha llevado a buscar la reutilización del software existente. El desarrollo de software basado en componentes permite reutilizar piezas de código pre elaborado que permiten realizar diversas tareas, conllevando a diversos beneficios como las mejoras en la calidad, la reducción del ciclo de desarrollo y el mayor retorno sobre la inversión. [37]

El objetivo fundamental de la Arquitectura Basada en Componentes es construir aplicaciones ensamblando módulos o componentes que han sido anteriormente diseñados por un grupo de personas, con el fin de que puedan ser rehusados en múltiples aplicaciones.

El uso de este paradigma posee algunas ventajas: [38]

- Reutilización del software.
- Simplifica las pruebas, al permitir que sean ejecutadas probando cada uno de los componentes, antes de probar el conjunto completo de componentes ensamblados.
- Simplifica el mantenimiento del sistema, pues cuando existe un débil acoplamiento entre componentes, el desarrollador es libre de actualizar y/o agregar componentes según sea necesario, sin afectar otras partes del sistema.
- Mayor calidad. Dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, la calidad de una aplicación basada en componentes mejorará con el paso del tiempo.

En el caso del módulo Diseñador de Consultas se utiliza está arquitectura debido a que el sistema en sí mismo constituye un componente del Sistema Generador de Reportes Dinámicos y además se integra a otros componentes ya existentes en el mismo como son: el Sistema de Autenticación, Autorización y Auditoría (SAAA) para la seguridad y el Diseñador de Modelos del cual se obtienen los modelos sobre los que se realizarán las consultas. Además, las consultas generadas por el diseñador de consultas serán utilizadas por el Diseñador de Reportes Dinámicos. En el desarrollo de la capa de presentación que fue realizada en Ext2.2 se implementaron varios componentes reutilizables para el desarrollo de interfaces web.

## Modelo Vista Controlador (MVC).

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

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

- **Modelo**: Este es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos; por ejemplo, no permitiendo comprar un número de unidades negativo, calculando si hoy es el cumpleaños del usuario o los totales, impuestos o importes en un carrito de compra.
- **Vista**: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.
- **Controlador**: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

En fin, en el MVC el controlador es el encargado de redirigir un procesamiento determinado para cada petición que reciba. Por tanto, el controlador debe poseer algún modo de conocer la correspondencia entre peticiones y posibles respuestas; deberá tener un mapa de peticiones y respuestas. El modelo representa la aplicación que responde una petición: los datos y las reglas de negocio. Por último, la vista define la forma de mostrar la información al usuario. En el caso del estilo arquitectónico MVC, el procesamiento se lleva a cabo entre sus tres componentes. El controlador recibe una petición y decide quién la lleva a cabo en la capa del modelo. Una vez que el modelo termina las operaciones pertinentes, devuelve el control de ejecución al controlador, y éste envía los resultados a la capa de la vista para que muestre los resultados al usuario. [39]. Muchos sistemas informáticos utilizan un Sistema de Gestión de Base de Datos para gestionar los datos. En MVC corresponde al modelo.

En el caso del módulo Diseñador de Consultas el MVC está presente en las capas que reúnen toda la lógica del negocio que son: la capa de las Vistas, el Controlador Frontal, la capa de Abstracción de Dato y la capa de Acceso a Datos. Así lo tiene establecido Symfony como framework a utilizar en el desarrollo del sistema. En la solución la Vista brindada por Symfony se utiliza como una interfaz para la comunicación e intercambio de información con la capa de Presentación, el Modelo (Abstracción y Acceso a Datos) gestiona los datos que necesita el Controlador para ejecutar la lógica del negocio, se utiliza Propel como

ORM[16] para acceder de forma efectiva a la base de datos desde un contexto orientado a objetos. Es importante destacar que nuestro sistema utiliza una variante del MVC, en este caso, el patrón Controlador Frontal. Consiste en la existencia de un único controlador en el sistema Es el único punto de entrada a la aplicación, carga la configuración y determina la acción a ejecutarse.

La utilización de este patrón nos permite centralizar diferentes aspectos del sistema como son el tratamiento de excepciones, el tratamiento de la configuración, las peticiones y respuestas al servidor web, la persistencia de los datos del usuario, entre otros. Además con el uso del Controlador garantizamos una mejor seguridad manejada al controlar cada uno de las acciones a ejecutar.

## 2.5 Estándar de codificación.

Un estándar de codificación completo comprende todos los aspectos de la generación de código. Si bien los programadores deben implementar un estándar de forma prudente, éste debe tender siempre a lo práctico. Un código fuente completo debe reflejar un estilo armonioso, como si un único programador hubiera escrito todo el código de una sola vez. Al comenzar un proyecto de software, es necesario establecer un estándar de codificación para asegurarse que todos los programadores del proyecto trabajen de forma coordinada.

La legibilidad del código fuente repercute directamente en lo bien que un programador comprende un sistema de software. La mantenibilidad del código es la facilidad con que el sistema de software puede modificarse para añadirle nuevas características, modificar las ya existentes, depurar errores, o mejorar el rendimiento. Aunque la legibilidad y la mantenibilidad son el resultado de muchos factores, una faceta del desarrollo de software en la que todos los programadores influyen especialmente, es en la técnica de codificación. El mejor método para asegurarse de que un equipo de programadores mantenga un código de calidad es establecer un estándar de codificación sobre el que se efectuarán luego, revisiones del código de rutinas. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento. Además, si se aplica de forma continuada un estándar de codificación bien

[16] El **mapeo objeto-relacional** es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional.

definido, se utilizan técnicas de programación apropiadas y posteriormente se efectúan revisiones del código de rutinas, caben muchas posibilidades de que un proyecto de software se convierta en un sistema de software fácil de comprender y de mantener. Las técnicas de codificación incorporan muchos aspectos del desarrollo del software. Aunque generalmente no afectan a la funcionalidad de la aplicación, sí contribuyen a una mejor compresión del código fuente. En esta fase se tienen en cuenta todos los tipos de código fuente, incluidos los lenguajes de programación, de marcado o de consulta. En general una técnica de codificación no pretende formar un conjunto inflexible de estándares de codificación. Más bien intenta servir de guía en el desarrollo de un estándar de codificación para un proyecto específico de software. Cuando se trabaja en equipo es necesario hacer código legible y entendible no sólo para quien lo escribe, sino también para quien lo lee, y para eso es necesario tener en cuenta varios aspectos:

- Las cláusulas, es decir, la notación que se utilizará para nombrar cada uno de los identificadores que se declaran.
- La estructura del código en sí, es decir, lo referente a las tabulaciones y los espacios entre líneas, y dentro de las líneas, los espacios entre los operadores y estructuras que componen el lenguaje en que programamos.

Para la solución del problema tratado en este trabajo se ha establecido un estándar de codificación de la siguiente manera.

**Nombre de las Aplicaciones.**

Los nombres de las aplicaciones deben permitir que con solo leerlo se conozca el propósito del proceso a automatizar.

Ejemplo:

Diseñador de Consultas.

**Clases y objetos.**

Los nombres de las clases y los objetos deben permitir que con sólo leerlos, se reconozca el propósito de la misma.

**Atributos de las clases.**

Para el nombre de los atributos de una clase se utilizará la notación lowerCamelCase. Si es una sola palabra se mantiene todo en minúscula. Permite con sólo leerlo, reconocer el propósito del mismo dentro de la clase.

Ejemplo:

```
$cantTotal;

$nombre;

class cbdClase

{

  $nombreTabla; //el atributo denota el nombre de una tabla

}
```

**Funciones.**

Los nombres de las funciones utilizarán la notación lowerCamelCase. Con sólo leerlo se reconoce el propósito dentro de la función.

Ejemplo:

```
function buscarUnidad()
```

```
{

    //instrucciones

}
```

**Variables con nombres compuestos.**

En los nombres de las variables se utilizará la notación lowerCamelCase.

Ejemplo:

```
$cantTotal;  //en el caso de un nombre compuesto

$nombre;     //en el caso de un nombre simple
```

**Nombre de clases.**

En los nombres de las clases se utilizará la notación UpperCamelCase ejemplo “Parameters” y en caso de que la clase sea compuesta “MainModel”.

**Indentación.**

Nunca se debe usar tabulaciones en el código. Los saltos de la indentación serán a 2 espacios.

Ejemplo:

```
<?php

class sfFoo

{
```

```
  public function bar()
  {
    sfCoffee::make();
  }
}
```

**Comentarios.**

Los comentarios del código se deben localizar en sus líneas correspondientes y en el siguiente formato.

```
// space first, with no full stop needed
```

**Espacios en blanco entre operadores lógicos y aritméticos.**

Se usan espacios en blanco entre los operadores lógicos y aritméticos para lograr una mayor legibilidad en el código.

Ejemplo:

```
$tabla = ‘nom_producto’;

if (($tabla) && ($fields))
```

Como se observa, los estilos de código hacen que el programa fuente sea más legible, lo cual ayuda en la comunicación entre desarrolladores, y permite una temprana detección de errores. la propuesta es completamente extensible.(Ver anexo 5)

## 2.6 Modelo de Despliegue.

Para el desarrollo de una aplicación informática es muy importante que el arquitecto de software preste especial atención a la configuración de hardware en la cual se desplegará el sistema, identificando nodos procesadores (computadoras), dispositivos y protocolos; todo lo cual en su conjunto conforma el modelo de despliegue, el cual define la arquitectura física del sistema por medio de nodos interconectados. Estos nodos son elementos hardware sobre los cuales pueden ejecutarse los elementos software y para representar esta información se emplea el elemento de UML denominado Diagrama de Despliegue.

Para el despliegue del Módulo Diseñador de Consultas se contará con un Servidor de Aplicaciones donde se encontrará el código fuente de la aplicación. Además contará con un Servidor de Base de Datos donde estará la base de datos de la aplicación, el cual se comunicará mediante protocolo TCP/IP con el Servidor de Aplicaciones; y por último contará con las PC Cliente que serán las encargadas de permitirle al usuario el acceso al sistema mediante protocolo HTTP.

A continuación se presenta el Diagrama de Despliegue correspondiente al Módulo Diseñador de Consultas del Diseñador de Reportes Dinámico. (Figura 2.1)

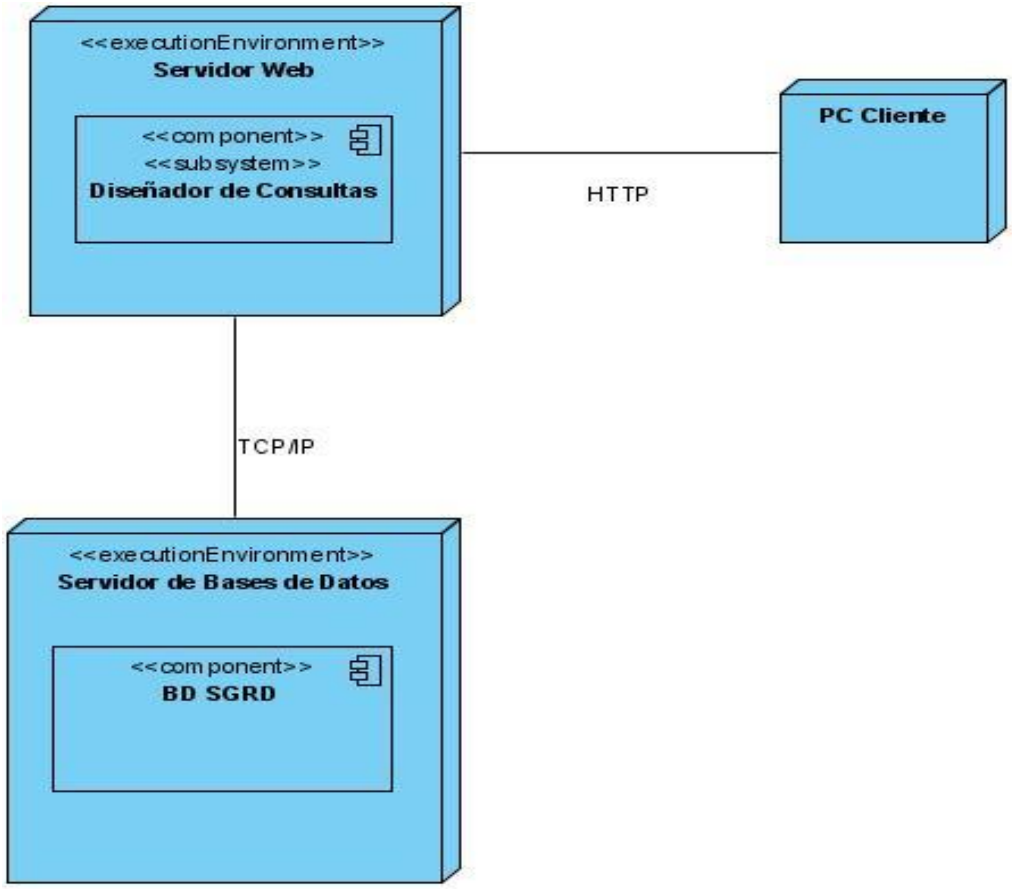


Figura 2.1. Diagrama de la Vista de Despliegue.

**Servidor Web:** En este nodo se encuentra la aplicación del Diseñador de Consultas.

**Servidor de Base de Datos:** es donde se encuentra almacenada la base de datos del Sistema Generador de Reportes Dinámicos. En la cual se encontraran los modelos generados por el Diseñador de Modelos que serán empleados pera el diseño de la consulta.

**PC Cliente:** Desde este nodo se accede a través de un navegador al sistema.

## 2.7 Vista de implementación.

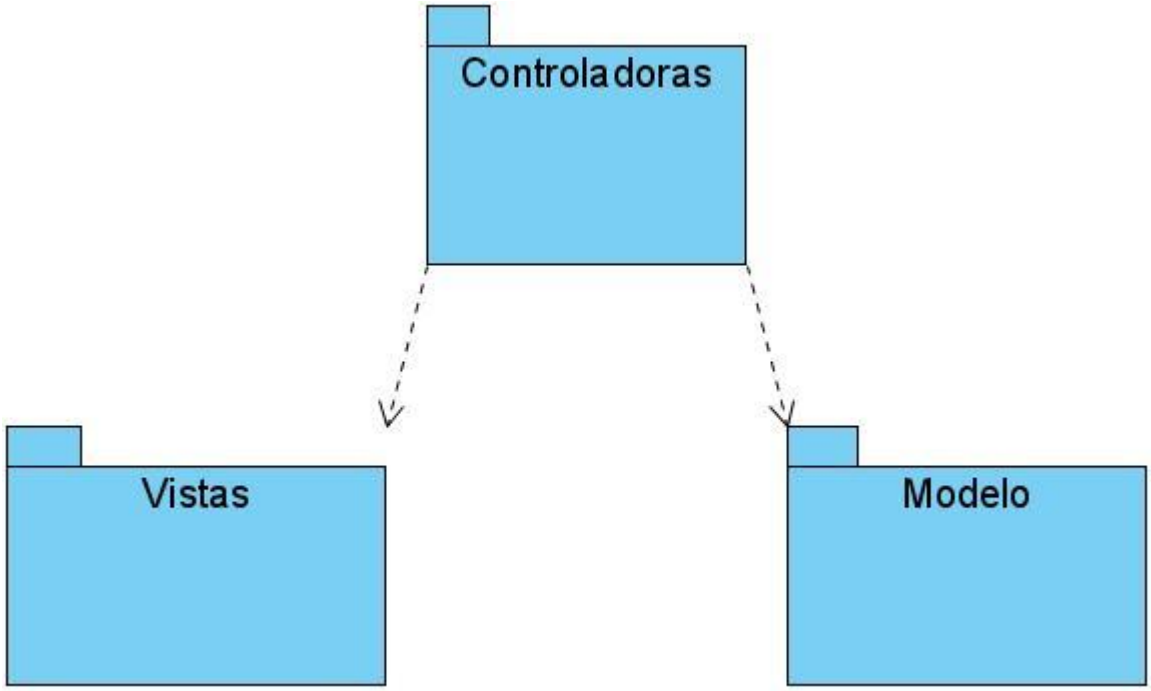


Fig. 2.2 Diagrama de paquetes.

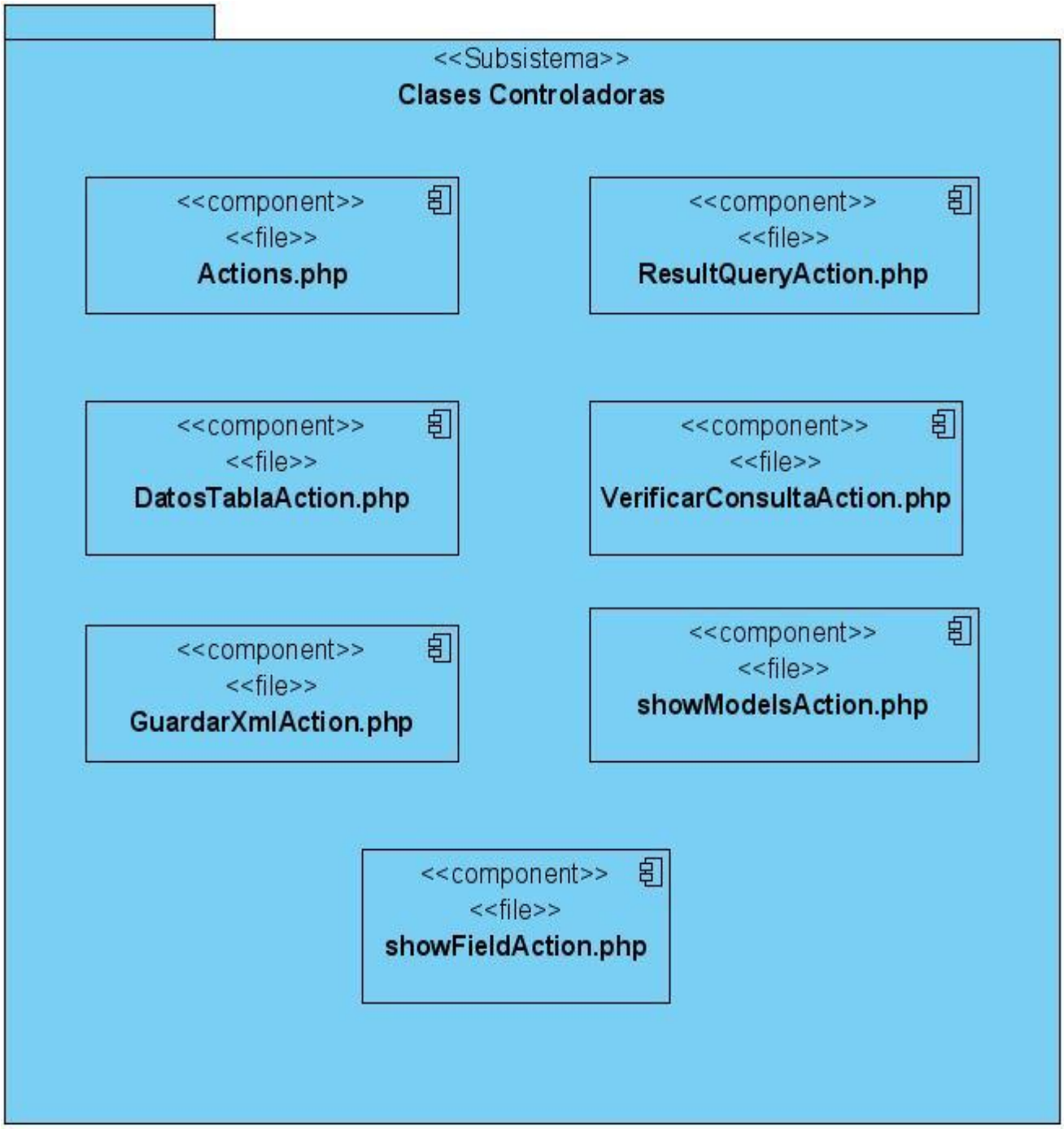


Fig.2.3 Diagrama de Componentes de las clases controladoras.

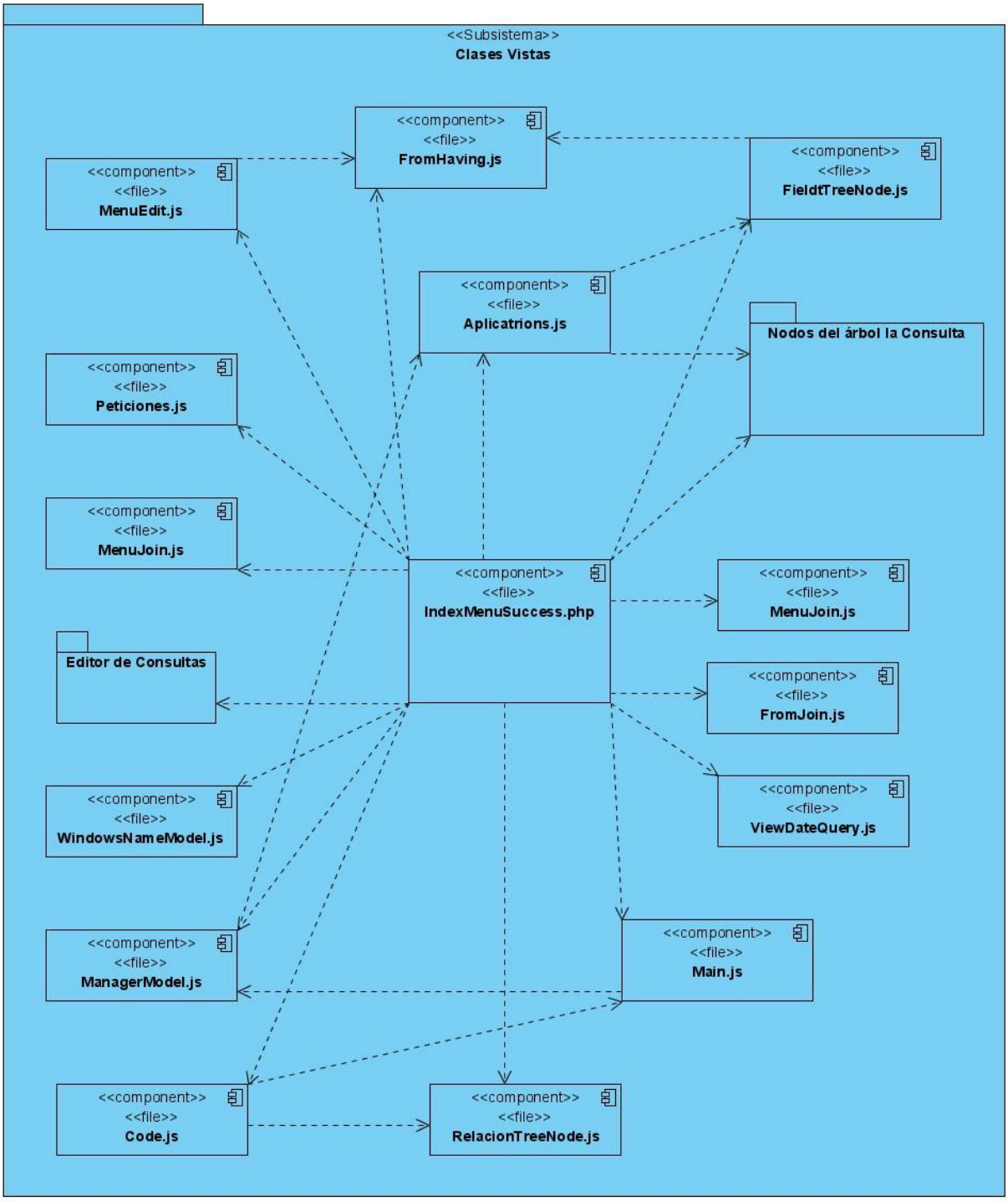


Fig.2.4 Diagrama de Componentes de las Vista

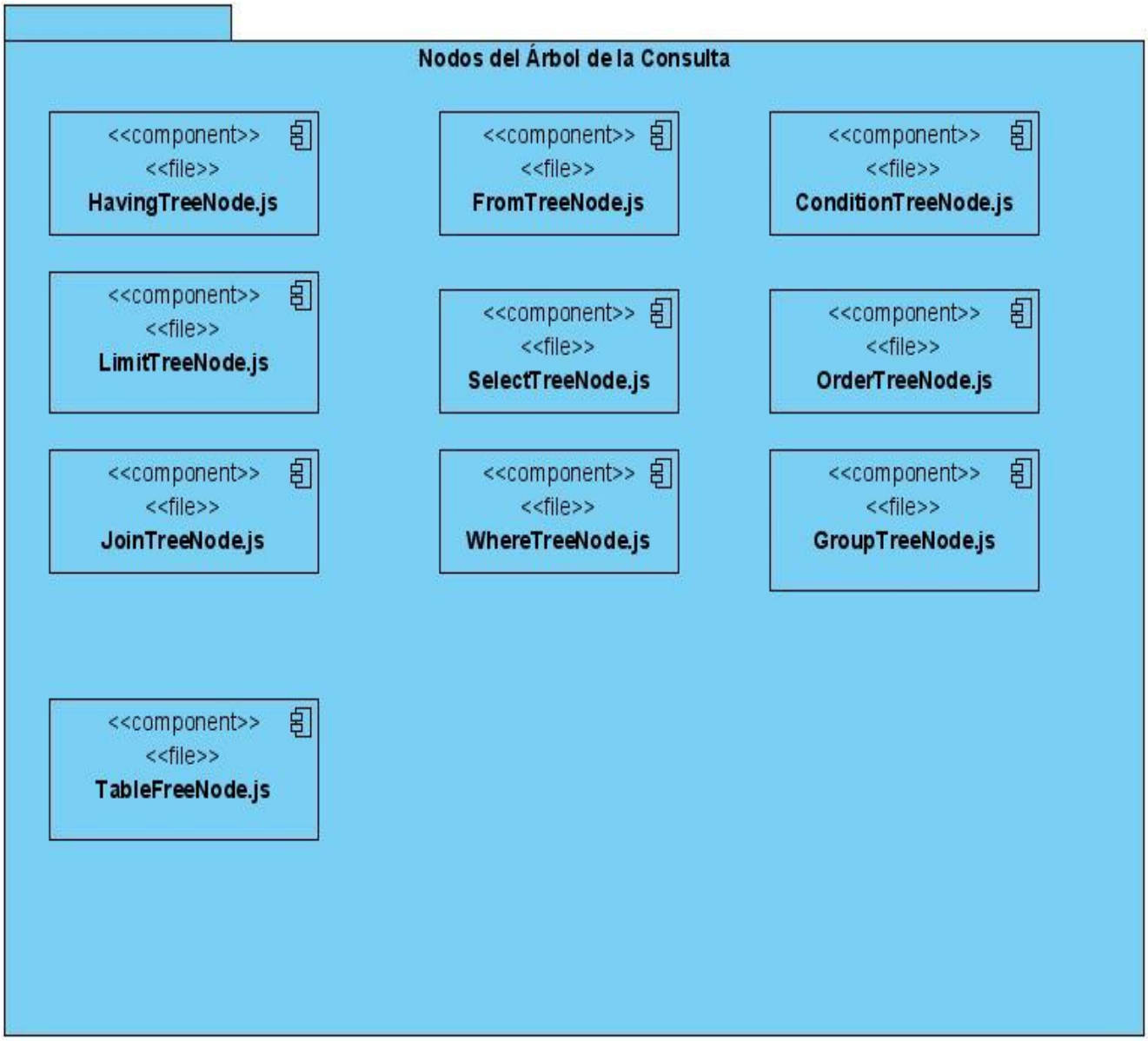


Fig.2.5 Diagrama de componentes del paquete  Nodos del Árbol de la Consulta

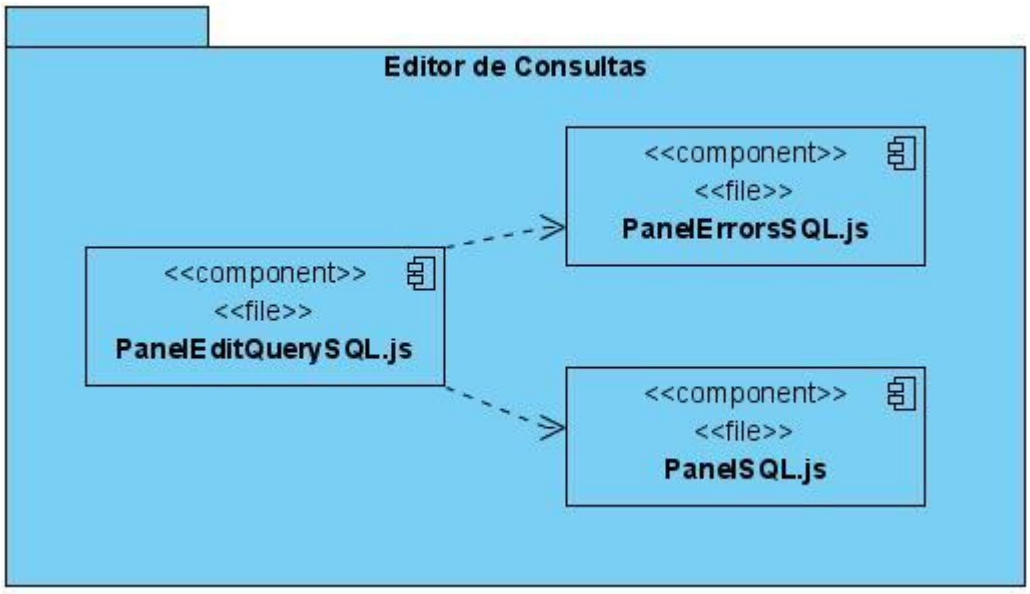


Fig.2.6 Diagrama de Componentes del paquete del Editor de Consultas

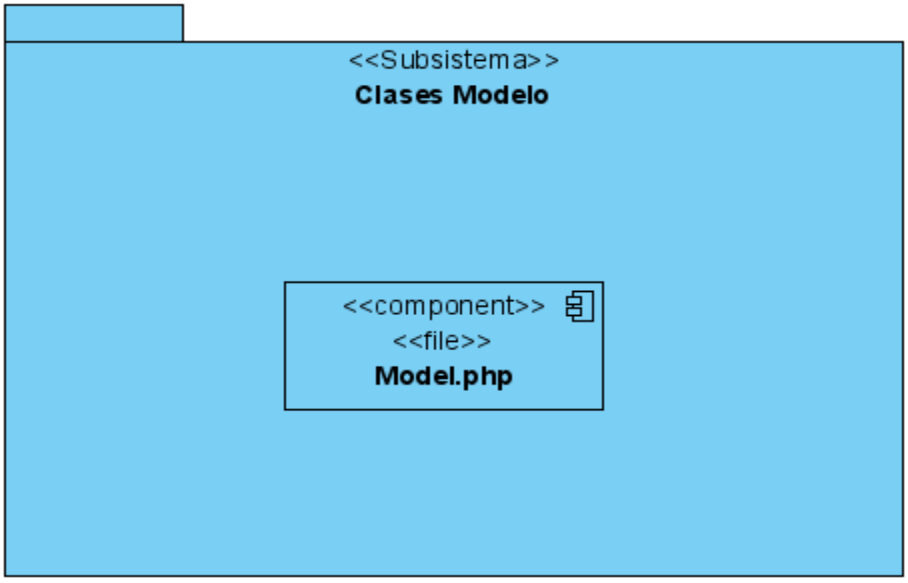


Fig. 2.7 Diagrama de componentes de la clase del modelo.

## 2.8 Conclusiones.

En este capítulo se definieron los requisitos no funcionales del sistema, los que se tendrán en cuenta para realizar la aplicación. Se especificó el estándar de codificación que estará presente en la realización del sistema, para que el código esté más organizado y fácil de entender. Además, se realizó un estudio de los patrones y estilos arquitectónicos necesarios en la solución del sistema donde se ha establecido una arquitectura cliente/servidor, basada en componentes, en tres capas y empleando el estilo arquitectónico modelo-vista- controlador.

# Capítulo 3 Descripción y análisis de la solución propuesta.

## 3.1 Introducción

El capítulo que a continuación se presenta, aborda primeramente las estrategias a seguir para la integración con otros módulos, además de la descripción de las clases del sistema y de la tabla de la Base de Datos donde serán almacenadas las consultas pertenecientes a cada modelo.

## 3.2 Integración con otros Módulos o Sistemas.

Se utiliza una arquitectura basada en componentes, donde cada módulo presenta funcionalidades propias y utiliza los servicios brindados por otros componentes, Esto hace que el funcionamiento del Módulo de Diseño de Consultas Dinámicas dependa de otros módulos del Generador de Reportes Dinámicos y a su vez que estos dependan de él, garantizando la reutilización y evitando la duplicidad de la información. El componente o módulo del Sistema Generador de Reportes Dinámicos que fue identificado como necesario para complementar las funcionalidades que el sistema brinda es: El Módulo Diseñador de Modelos, aunque también emplea un módulo, que no pertenece al Generador de Reportes Dinámicos, que es el de Autenticación, Autorización y Auditoría (SAAA) para garantizar la seguridad del sistema y a su vez el módulo que depende de él es el Diseñador de Reportes.

### 3.2.1 Componente de Seguridad (SAAA).

Es el sistema que gestiona el nivel de acceso a los distintos componentes del Generador de Reportes Dinámicos, permitiendo que los usuarios tengan un solo identificador y una contraseña para acceder a todos los sistemas. Este brinda a través de Web Services una serie de métodos que facilitan todos los servicios necesarios para asegurar el sistema y consiste en suministrar un nombre de usuario único y una contraseña. Si el usuario no se encuentra registrado se reportará un error de acceso. Si el usuario autenticado se encuentra registrado se autoriza su acceso y se crea un certificado digital que contiene un

identificador único (token) de 32 caracteres, que tiene como información: el identificador del usuario y la contraseña. Para su uso se crearon diferentes clases que encapsulan el manejo de la seguridad así como la comunicación con este módulo, de manera que esta queda de forma global para todo el sistema sin necesidad de ser implementada en cada subsistema pero sí utilizada por todos y de manera particular por cada uno.

### 3.2.2 Diseñador de Modelos.

Este módulo permite; dado el tipo de gestor, la dirección IP o el nombre del servidor, el puerto, el usuario y la clave; establecer la conexión con el gestor de bases de datos del cliente y obtener la base de datos, para luego poder crear un modelo a partir de esta que puede contener todas las tablas o un subconjunto de ellas según desee el usuario. Sobre estos modelos se diseñaran las consultas, empleando el Diseñador de Consultas Dinámicas.

### 3.2.3 Diseñador de Reportes.

Este módulo utiliza las consultas previamente diseñadas por el del Diseñador de Consultas Dinámicas para el diseño del reporte, permitiendo asignarle a un campo, que puede ser de texto o número, el valor devuelto por la consulta. Si el campo es de tipo texto, solo se podrá asignar al mismo el devuelto por la consulta de tipo texto, si el campo es de tipo número solo se podrán asignar al mismo los devueltos por la consulta de tipo número.

## 3.3 Descripción de las clases u operaciones necesarias.

Para darle cumplimiento a las funcionalidades que requiere el sistema se implementaron 12 clases vistas: 5 controladoras y 1 modelo. A continuación se describen aquellas que se consideran críticas debido a que representan procesos fundamentales en lo que a negocio y presentación se refiere.

### 3.3.1 Clases Interfaces.

Tabla 1.Descripción de la clase interfaz< Workflow>

| Nombre:Workflow | |
|---|---|
| Tipo de clase: interfaz | |
| Atributo | Tipo |
| gridWidthX | int |
| gridWidthY | int |
| verticalSnapToHelperLine | Line |
| horizontalSnapToHelperLine | Line |
| Figures | ArrayList |
| Lines | ArrayList |
| commonPorts | ArrayList |
| dropTargets | ArrayList |
| Compartments | ArrayList |
| selectionListeners | ArrayList |
| Dialogs | ArrayList |
| Dragging | bool |
| commandStack | CommandStack |
| connectionLine | Line |
| resizeHandleStart | LineStartResizeHandle |
| resizeHandleEnd | LineStartResizeHandle |
| resizeHandle1 | ResizeHandle |
| resizeHandle2 | ResizeHandle |
| resizeHandle3 | ResizeHandle |
| resizeHandle4 | ResizeHandle |
| resizeHandle5 | ResizeHandle |
| resizeHandle6 | ResizeHandle |
| resizeHandle7 | ResizeHandle |

| resizeHandle8 | ResizeHandle |
|---|---|
| **Por cada responsabilidad:** | |
| **Nombre:** | showMenu |
| Descripción: | Función que permite mostrar un menú en una posición x, y del Workflow de trabajo. |
| **Nombre:** | addFigure |
| Descripción: | Función que permite adicionar una figura al marco de trabajo. |
| **Nombre:** | removeFigure |
| Descripción: | Función que permite eliminar una figura del marco de trabajo. |
| **Nombre:** | getFigure |
| Descripción: | Permite obtener una figura del marco de trabajo dado el id. |
| **Nombre:** | getFigures |
| Descripción: | Función que permite obtener todas figura del marco de trabajo. |
| **Nombre:** | getCurrentSelection |
| Descripción: | Función que permite obtener una figura seleccionada en el marco de trabajo. |
| **Nombre:** | getLine |
| Descripción: | Función que permite obtener una línea de relación entre figuras dado el id del marco de trabajo. |
| **Nombre:** | getLines |
| Descripción: | Función que permite obtener las líneas de relación entre figuras del marco de trabajo. |
| **Nombre:** | registerPort |
| Descripción: | Para adicionar un puerto al marco de trabajo. |
| **Nombre:** | unregisterPort |
| Descripción: | Para eliminar un puerto del marco de trabajo. |
| **Nombre:** | showConnectionLine |
| Descripción: | Muestra la línea de conexión entre 2 puertos. |
| **Nombre:** | onKeyDown |
| Descripción: | Para eliminar figuras en el marco de trabajo por el teclado. |

| | |
|---|---|
| **Nombre:** | hideLineResizeHandles. |
| Descripción: | En caso que una figura sea eliminada y tenga relación con otra, eliminar la línea automáticamente. |

Tabla 2.Descripción de la clase interfaz< MenuJoin >

| Nombre:MenuJoin | |
|---|---|
| Tipo de clase: interfaz | |
| Atributo | Tipo |
| sourcePort | OutputPort |
| targetPort | OutputPort |
| lineSegments | Array |
| Workflow | Workflow |
| Menú | Menu |
| **Nombre:** | connection |
| Descripción: | Para crear una nueva conexión entre figuras donde se mostrará la línea de conexiones. |
| **Nombre:** | onContextMenu |
| Descripción: | Permite crear un menú sobre la línea de conexiones. |

Tabla 3.Descripción de la clase interfaz< FromHaving>

| Nombre: FromHaving | |
|---|---|
| Tipo de clase: interfaz | |
| Atributo | Tipo |
| fpHaving | FormPanel |

| | |
|---|---|
| winHaving | Window |
| cbQB | Checkbox |
| cbNull | Checkbox |
| cmbFromWhere | Combobox |
| cmbFromArray | Combobox |
| taDescripcion | TexArea |
| taDocnum | TexArea |
| operadorRelacional | Array |
| operadorCondicional | Array |
| checkN | int |
| btnAplicar | Button |
| btnCerrar | Button |
| Descripción: | Para cada button en su opción de configuración handler se le asigna una función que define la programación del evento de un botón. |

Tabla 4.Descripción de la clase interfaz< Main>

| | |
|---|---|
| Nombre: Main | |
| Tipo de clase: interfaz | |
| Atributo | Tipo |

| | |
|---|---|
| treeWestQueryBuilder | TreePanel |
| Tbar | Toolbar |
| Bbar | Toolbar |
| centerQueryBuilder | Panel |
| editorQB | PanelEditorConsultaSQL |
| tabMarcoQB | TabPanel |
| viewportQueryBuilder | Panel |
| rootGroupQB | string |
| rootQueryNode | TreeNode |

| | |
|---|---|
| **Nombre:** | addWorkFlowQB |
| Descripción: | Adiciona al Workflow el button archivo para guardar la consulta a xml y actualiza el root del árbol de la consulta. |
| **Nombre:** | getTreeModelsQB |
| Descripción: | Carga el árbol con todos los modelos y de cada uno sus respectivas consultas. |
| **Nombre:** | startQueryBuilder |
| Descripción: | Función que carga dentro de un panel el árbol de los modelos y el árbol de la consulta. |
| **Nombre:** | updateGroupQB |
| Descripción: | Función que actualiza los hijos del Group. |

Tabla 5.Descripción de la clase interfaz< PanelEditorConsultaSQL >

| Nombre: PanelEditorConsultaSQL | |
|---|---|
| Tipo de clase: interfaz | |
| Atributo | Tipo |
| panelSecuenciaSQL | TreePanel |
| panelErroresSQL | Toolbar |
| Guery | string |
| btnAceptar | Button |
| btnCancelar | Button |
| btnVerificar | Button |
| **Nombre:** | handler |
| Descripción: | Para cada button en su opción de configuración handler se le asigna una función que define la programación del evento de un botón. |

Tabla 6.Descripción de la clase interfaz< ResultadoQuery >

| Nombre: ResultadoQuery |
|---|
| Tipo de clase: interfaz |

| Atributo | Tipo |
|---|---|
| Contenido | Store |
| grNumero1 | GridPanel |
| frGrigPanelResult | Form |

| **Nombre:** | datosTableGrid |
|---|---|
| Descripción: | Función para obtener el texto de las cabeceras de las columnas de Grid. |

## 3.3.2 Clases Controladoras.

Tabla 7.Descripción de la clase Controladora< GuardarXmlAction >

| Nombre: GuardarXmlAction | |
|---|---|
| Tipo de clase: controladora | |
| Atributo | Tipo |
| **Nombre:** | execute |
| Descripción: | Función que permite guardar el XML generado por el diseño de la consulta. |

Tabla 8.Descripción de la clase Controladora< ResultQueryAction >

| Nombre: ResultQueryAction |
|---|
| Tipo de clase: controladora |

| Atributo | | Tipo |
|---|---|---|
| **Nombre:** | execute | |
| Descripción: | Función que permite ejecutar la consulta y devuelve los datos en formato JSON | |

Tabla 9.Descripción de la clase Controladora< VerificarConsultaAction >

| Nombre: VerificarConsultaAction | | |
|---|---|---|
| Tipo de clase: controladora | | |
| Atributo | | Tipo |
| **Nombre:** | execute | |
| Descripción: | Función que permite verificar si la consultas se encuentra bien editada y en caso de error devuelve el tipo de error y una breve descripción del mismo. | |

Tabla 10.Descripcion de la clase Controladora< ShowModelsAction>

| Nombre: ShowModelsAction | | |
|---|---|---|
| Tipo de clase: controladora | | |
| Atributo | | Tipo |
| **Nombre:** | execute | |
| Descripción: | Función que devuelve todos los modelos. | |

| | |
|---|---|
| **Nombre:** | tableArray |
| Descripción: | Función que muestra las tablas pertenecientes a cada modelo. |
| **Nombre:** | viewQuery |
| Descripción: | Función que devuelve las query pertenecientes a cada modelo. |

Tabla 11.Descripción de la clase Controladora< ShowFieldAction>

| Nombre: ShowFieldAction | |
|---|---|
| Tipo de clase: controladora | |
| Atributo | Tipo |
| **Nombre:** | execute |
| Descripción: | Función que devuelve todos los campos de una tabla. |

## 3.3.3 Clase Modelo.

Tabla 11.Descripción de la clase Modelo< Model>

| Nombre: Model | |
|---|---|
| Tipo de clase: Modelo | |
| Atributo | Tipo |

| | |
|---|---|
| $readerXML | string |
| $reader | string |

| | |
|---|---|
| **Nombre:** | getFields |
| Descripción: | Función que devuelve todos los campos de una tabla. |
| **Nombre:** | getAll |
| Descripción: | Devuelve un array con los modelos. |
| **Nombre:** | InitReaderXML |
| Descripción: | Activa el lector para obtener XML a partir de su xmlModel. |
| **Nombre:** | InitReader |
| Descripción: | Activa el lector para obtener objetos a partir de su xmlModel |
| **Nombre:** | getXMlTables |
| Descripción: | Devuelve un xml con las tablas: nombre, alias y tipo. |
| **Nombre:** | getTables |
| Descripción: | Devuelve un array de las tablas con alias, nombre y tipo. |
| **Nombre:** | getFunctions |
| Descripción: | Devuelve array de funciones. |
| **Nombre:** | getViews |
| Descripción: | Devuelve array de vistas. |

| | |
|---|---|
| **Nombre:** | getFunctionByName |
| Descripción: | Devuelve una función dado su nombre. |
| **Nombre:** | getXMLTableByName |
| Descripción: | Devuelve un XML con una tabla dado su nombre. |
| **Nombre:** | getTableByName |
| Descripción: | Devuelve una tabla dado su nombre. |
| **Nombre:** | getXml_DSN |
| Descripción: | Devuelve el nombre del DataSource en xml. |
| **Nombre:** | getDataSource |
| Descripción: | Devuelve el datasource. |
| **Nombre:** | getModelByName |
| Descripción: | Devuelve un modelo dado el nombre. |
| **Nombre:** | getById |
| Descripción: | Obtener modelo por el id. |
| **Nombre:** | hasSchema |
| Descripción: | Verificar si el gestor de bases de datos es PostgreSQL ya que tiene la característica de poseer esquemas. |
| **Nombre:** | getSchemas |
| Descripción: | Obtener esquemas en caso de que el gestor de bases de datos sea PostgreSQL. |

| | |
|---|---|
| **Nombre:** | validateName |
| Descripción: | Verificar que dado un nombre de un modelo realmente exista. |
| **Nombre:** | addQuery |
| Descripción: | Adicionar una consulta al modelo. Consiste en adicionar la consulta como un hijo del XML del modelo verificando siempre que no exista antes una consulta con igual nombre. |
| **Nombre:** | getQueriesNames |
| Descripción: | Obtener los nombres de las consultas existentes. |
| **Nombre:** | getQueryXMlByName |
| Descripción: | Dado el nombre de una query, es obtenida en formato XML o en caso de que no exista se devuelve false. |
| **Nombre:** | getTableByID |
| Descripción: | Obtener una tabla dado su id. |
| **Nombre:** | getFunctionByID |
| Descripción: | Obtener una función dado su id. |
| **Nombre:** | getXMLTableByID |
| Descripción: | Obtener en formato XML la tabla dado su id. |

### 3.3.4 Descripción de la tabla utilizada por el Diseñador de Consultas de la BD SGRD.

Tabla 3.12 Descripción de la Tabla tbmodel.

| **Nombre:** tbmodel | | |
|---|---|---|
| **Descripción:** Se encarga de guardar en formato XML los modelos obtenidos por el Diseñador de Modelos y en ellos se adicionan las consultas diseñadas por el Diseñador de Consultas. | | |
| **Atributo** | **Tipo** | **Descripción** |
| idmodelo | [pk] integuer | Guarda el identificador de cada modelo. |
| xmlmodel | text | Guarda el xml del modelo. |
| namemodel | varchar | Para guardar el nombre del modelo. |
| iddatasource | integuer | Guarda el id del datasource. |

## 3.4 Conclusiones

En este capítulo se describe cómo está implementado el módulo. Se muestra la descripción de las clases y operaciones necesarias, así como de la tabla que utiliza en la base de datos el módulo en cuestión, el cual cumple con los requisitos funcionales establecidos por los analistas. También, se describió la integración del módulo con otros ya existentes, así como sus características, para mejorar la comprensión del trabajo realizado, y del sistema en general.

# Capítulo 4 : Validación de la Solución Propuesta

## 4.1 Introducción

En el desarrollo del software existen grandes posibilidades de cometer errores. Pueden darse por una mala especificación de los requisitos funcionales, uso indebido de las estructuras de datos, errores de programación, etc. El mismo ha de ir acompañado de alguna actividad que garantice la calidad; la prueba es un elemento crítico para la garantía de calidad del software. La prueba y validación de los resultados es un proceso que se realiza en cada una de las etapas de desarrollo no solo después del software ser desarrollado. Si se siguen encontrando errores cada vez que se procesa el programa, las pruebas deben continuar. Durante el mantenimiento debe de existir documentación de pruebas que incluya casos de prueba y resultados esperados. Por lo dicho anteriormente, es desarrolló este capítulo, con el objetivo de especificar las pruebas que serán hechas al software y describir todo lo encontrado en las mismas.

## 4.2 Técnicas de prueba.

Las técnicas de evaluación dinámica o prueba proporcionan distintos criterios para generar casos de prueba que provoquen fallos en los programas. Estas técnicas se agrupan en: [40]

- Técnicas de caja blanca o estructural, que se basan en un minucioso examen de los detalles procedimentales del código a evaluar, por lo que es necesario conocer la lógica del programa.
- Técnicas de caja negra o funcionales, que realizan pruebas sobre la interfaz del programa a probar, entendiendo por interfaz las entradas y salidas de dicho programa. No es necesario conocer la lógica del programa, únicamente la funcionalidad que debe realizar.

El anexo 6 representa gráficamente la filosofía de las pruebas de caja blanca y caja negra. Como puede observarse las pruebas de caja blanca necesitan conocer los detalles procedimentales del código,

mientras que las de caja negra únicamente necesitan saber el objetivo o funcionalidad que el código ha de proporcionar.

Puede parecer que una prueba de caja blanca completa llevará a disponer de un código perfectamente correcto. De hecho esto ocurriría si se han probado todos los posibles caminos por los que puede pasar el flujo de control de un programa. Sin embargo, para programas de cierta envergadura, el número de casos de prueba que habría que generar sería excesivo, el número de caminos incrementa exponencialmente a medida que el número de sentencias condicionales y bucles aumenta. Sin embargo, este tipo de prueba no se desecha como impracticable. Se pueden elegir y ejercitar ciertos caminos representativos de un programa.

Por su parte, tampoco sería factible en una prueba de caja negra probar todas y cada una de las posibles entradas a un programa, por lo que análogamente a como ocurría con las técnicas de caja blanca, se seleccionan un conjunto representativo de entradas y se generan los correspondientes casos de prueba, con el fin de provocar fallos en los programas.

En realidad estos dos tipos de técnicas son técnicas complementarias que han de aplicarse al realizar una prueba dinámica, ya que pueden ayudar a identificar distintos tipos de faltas en un programa. A continuación, se describen los procedimientos propuestos por ambos tipos de técnicas para generar casos de prueba para el sistema en cuestión.

## 4.3 Pruebas de Caja Blanca o Estructurales.

A este tipo de técnicas se le conoce también como Técnicas de Caja Transparente o de Cristal. Este método se centra en cómo diseñar los casos de prueba atendiendo al comportamiento interno y la estructura del programa. Se examina así la lógica interna del programa sin considerar los aspectos de rendimiento.

El objetivo de la técnica es diseñar casos de prueba para que se ejecuten, al menos una vez, todas las sentencias del programa, y todas las condiciones tanto en su vertiente verdadera como falsa. Como se ha

indicado ya, puede ser impracticable realizar una prueba exhaustiva de todos los caminos de un programa. Por ello se han definido distintos criterios de cobertura lógica, que permiten decidir qué sentencias o caminos se deben examinar con los casos de prueba.

Estos criterios son: [41]

- Cobertura de Sentencias: Se escriben casos de prueba suficientes para que cada sentencia en el programa se ejecute, al menos, una vez.
- Cobertura de Decisión: Se escriben casos de prueba suficientes para que cada decisión en el programa se ejecute una vez con resultado verdadero y otra con el falso.
- Cobertura de Condiciones: Se escriben casos de prueba suficientes para que cada condición en una decisión tenga una vez resultado verdadero y otra falso.
- Cobertura Decisión/Condición: Se escriben casos de prueba suficientes para que cada condición en una decisión tome todas las posibles salidas, al menos una vez, y cada decisión tome todas las posibles salidas, al menos una vez.
- Cobertura de Condición Múltiple: Se escriben casos de prueba suficientes para que todas las combinaciones posibles de resultados de cada condición se invoquen al menos una vez.
- Cobertura de Caminos: Se escriben casos de prueba suficientes para que se ejecuten todos los caminos de un programa. Entendiendo camino como una secuencia de sentencias encadenadas desde la entrada del programa hasta su salida.

Este último criterio es el que se empleo en el desarrollo de las pruebas de caja blanca debido al previo conocimiento del mismo por su utilización en los proyectos del la Universidad de Ciencias Informáticas.

**Cobertura de Caminos.**

La aplicación de este criterio de cobertura asegura que los casos de prueba diseñados permiten que todas las sentencias del programa sean ejecutadas al menos una vez y que las condiciones sean probadas tanto para su valor verdadero como falso.

Una de las técnicas empleadas para aplicar este criterio de cobertura es la **Prueba del Camino Básico**. Esta técnica se basa en obtener una medida de la complejidad del diseño procedimental de un programa (o de la lógica del programa). Esta medida es la complejidad ciclomática de McCabe[17], y representa un límite superior para el número de casos de prueba que se deben realizar para asegurar que se ejecuta cada camino del programa.

Los pasos a realizar para aplicar esta técnica son:

- Representar el programa en un grafo de flujo.
- Calcular la complejidad ciclomática.
- Determinar el conjunto básico de caminos independientes.
- Derivar los casos de prueba que fuerzan la ejecución de cada camino.

A continuación, se detallan cada uno de estos pasos.

**Representar el programa en un grafo de flujo.**

El grafo de flujo se utiliza para representar flujo de control lógico de un programa. Para ello se utilizan los tres elementos siguientes:

Nodos: representan cero, una o varias sentencias en secuencia. Cada nodo comprende como máximo una sentencia de decisión (bifurcación).

Aristas: líneas que unen dos nodos.

Regiones: áreas delimitadas por aristas y nodos. Cuando se contabilizan las regiones de un programa debe incluirse el área externa como una región más.

[17] Esta métrica, propuesta por Thomas McCabe en 1976, se basa en el diagrama de flujo determinado por las estructuras de control de un determinado código. De dicho análisis se puede obtener una medida cuantitativa de la dificultad de de crear pruebas automáticas del código y también es una medición orientativa de la fiabilidad del mismo.

Nodos predicados: cuando en una condición aparecen uno o más operadores lógicos (AND, OR, XOR,...) se crea un nodo distinto por cada una de las condiciones simples. Cada nodo generado de esta forma se denomina nodo predicado.

Así, cada construcción lógica de un programa tiene una representación. El anexo 7 muestra dichas representaciones.

**Calcular la complejidad ciclomática.**

La complejidad ciclomática es una métrica del software que proporciona una medida cuantitativa de la complejidad lógica de un programa. En el contexto del método de prueba del camino básico, el valor de la complejidad ciclomática define el número de caminos independientes de dicho programa, y por lo tanto, el número de casos de prueba a realizar.

Existen varias formas de calcular la complejidad ciclomática de un programa a partir de un grafo de flujo: [42]

1. El número de regiones del grafo coincide con la complejidad ciclomática, V (G).

2. La complejidad ciclomática, V (G), de un grafo de flujo G se define como:

V (G) = Aristas – Nodos + 2

3. La complejidad ciclomática, V (G), de un grafo de flujo G se define como

V (G) = Nodos Predicado + 1

Esta complejidad ciclomática determina el número de casos de prueba que deben ejecutarse para garantizar que todas las sentencias de un programa se han ejecutado al menos una vez, y que cada condición se habrá ejecutado en sus vertientes verdadera y falsa. Veamos ahora, cómo se identifican estos caminos.

**Determinar el conjunto básico de caminos independientes.**

Un camino independiente es cualquier camino del programa que introduce, por lo menos, un nuevo conjunto de sentencias de proceso o una condición, respecto a los caminos existentes. En términos del diagrama de flujo, un camino independiente está constituido por lo menos por una arista que no haya sido recorrida anteriormente a la definición del camino. En la identificación de los distintos caminos de un programa para probar se debe tener en cuenta que cada nuevo camino debe tener el mínimo número de sentencias nuevas o condiciones nuevas respecto a los que ya existen. De esta manera se intenta que el proceso de depuración sea más sencillo.

El conjunto de caminos independientes de un grafo no es único. No obstante, a continuación, se muestran algunas heurísticas para identificar dichos caminos:

(a) Elegir un camino principal que represente una función válida que no sea un tratamiento de error. Debe intentar elegirse el camino que atraviese el máximo número de decisiones en el grafo.

(b) Identificar el segundo camino mediante la localización de la primera decisión en el camino de la línea básica alternando su resultado mientras se mantiene el máximo número de decisiones originales del camino inicial.

(c) Identificar un tercer camino, colocando la primera decisión en su valor original a la vez que se altera la segunda decisión del camino básico, mientras se intenta mantener el resto de decisiones originales.

(d) Continuar el proceso hasta haber conseguido tratar todas las decisiones, intentando mantener como en su origen el resto de ellas.

Este método permite obtener V (G) caminos independientes cubriendo el criterio de cobertura de decisión y sentencia.

**Derivar los casos de prueba que fuerzan la ejecución de cada camino.**

El último paso es construir los casos de prueba que fuerzan la ejecución de cada camino.

### 4.3.1 Descripción de los casos de pruebas de Caja Blanca aplicados.

Para llevar a cabo los casos de pruebas de caja blanca, fue preciso tomar muestra del código fuente de la aplicación, para esto se tuvo en cuenta la implementación del método addQuery.

En el caso particular de esta aplicación se realizó una prueba a nivel de unidad empleándose el criterio de Cobertura de Caminos y dentro del mismo la técnica del Camino Básico.

En este caso se determinó el grafo de flujo y por ende la complejidad ciclomática, obteniéndose la cantidad de caminos independientes necesarios para aplicar los casos de pruebas correspondientes a cada cual.

Luego de tener elaborados los Grafos de Flujos y los caminos a recorrer, se preparan los casos de prueba que forzarán la ejecución de cada uno de esos caminos. Se escogen los datos de forma que las condiciones de los nodos predicados estén adecuadamente establecidas, con el fin de comprobar cada camino.

### 4.3.2 Casos de pruebas.

Siguiendo los pasos de la Técnica del Camino Básico se obtuvo por cada clase:

```
public function addQuery($query)
 {
      $names=array(); 1
      $oXml = simplexml_load_string($query); 1
      $titles = $oXml->xpath('//QUERY/@titulo'); 1
      $names=$this->getQueriesNames(); 1
```

```
        if (array_search($titles[0],$names)===false) 2

{
         $simple = simplexml_load_string($this->getXmlmodel()); 3
         $simple->addChild($query); 3
         $model = $simple->asXML(); 3
         $model = str_replace("<<","<",$model); 3
         $model = str_replace(">/>",">",$model);    3
         $this->setXmlmodel($model); 3
         return true; 3
        } else { return false; } //El nombre de la consulta se encuentra ya en el modelo  4


  } 5
```

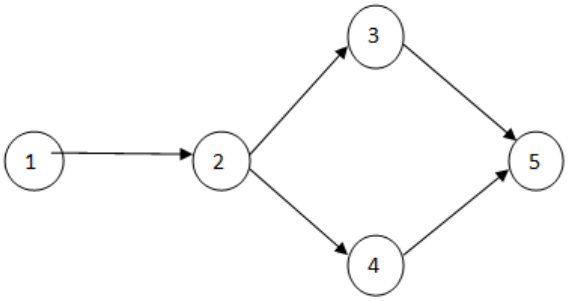


Fig.4.1  Representación del Grafo de flujo del Método addQuery.

En este primer caso se obtuvo que:

- el grafo de flujo tiene dos regiones.
- V (G)=5 arsitas-5 nodos+2=2.
- V (G)=1nodos predicado +1=2.

Por tanto la complejidad ciclomática del grafo de flujo de la figura 4.1 es igual a 2.

Como caminos independientes se determinaron:

Camino1:1-2-3-5

Camino2:1-2-4-5

Para estos caminos independientes se conformaron los siguientes casos de prueba:

Caso de prueba para el Camino1:

Descripción y asignación:

Se quiere determinar si dada una consulta se puede adicionar a un modelo si no existe otra en el modelo con igual nombre, para esto es asignado a la variable query una consulta cuyo nombre no se encuentra en las consultas existente en el modelo en el que se adicionará la consulta, siguiendo estas especificaciones se garantiza que se corra este camino independiente en su totalidad.

Resultado esperado:

Al entrar la consulta y comprobar que no existe una con igual nombre en el modelo donde se requiere la adición, el sistema debe adicionar la consulta satisfactoriamente al modelo y mostrar un mensaje “La consulta fue adicionada satisfactoriamente” avisando que la consulta ha sido adicionada al modelo.

**Caso de prueba para el Camino 2:**

Descripción y asignación:

Se quiere determinar si dada una consulta se puede adicionar a un modelo si existe otra en el modelo con igual nombre, para esto es asignado a la variable query una consulta cuyo nombre se encuentra en las consultas existente en el modelo en el que se adicionará la consulta, siguiendo estas especificaciones se garantiza que se corra este camino independiente en su totalidad.

Resultado esperado:

Al entrar la consulta y comprobar que existe una con igual nombre en el modelo donde se requiere la adición, el sistema no debe adicionar la consulta al modelo y mostrar un mensaje “Existe una consulta con igual nombre en el modelo” avisando que la consulta no ha sido adicionada al modelo debido a que existe una antes adicionada con el mismo nombre.

Evaluación de los resultados obtenidos.

Los resultados de la realización de los casos de pruebas de caja blanca para el caso de Adicionar Consultas a los modelos existentes en la base de datos, fueron satisfactorios al lograr la consistencia de las consultas permitiendo que no se modificaran, alteraran o incluso perdieran las consultas existentes en caso de que se adicionaran consultas con el mismo nombre en los modelos de datos, teniendo en cuenta la importancia que tienen las consultas para los gestores de bases de datos.

## 4.4 Pruebas de Caja Negra.

Se llevan a cabo sobre la interfaz del software.

**Objetivo**

El objetivo es demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y se produce un resultado correcto, y que la integridad de la información externa se mantiene. [43]

- Se centran principalmente en los requisitos funcionales del software.
- Permiten encontrar:
    - Funciones incorrectas o ausentes.
    - Errores de interfaz.
    - Errores en estructuras de datos o en accesos a las Bases de Datos externas.
    - Errores de rendimiento.
    - Errores de inicialización y terminación.

### 4.4.1 Descripción de los Casos de Prueba de Caja Negra.

Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales del programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales del sistema y ejercitándolos.

La prueba de Caja Negra no es una alternativa a las técnicas de prueba de la Caja Blanca, sino un enfoque complementario que intenta descubrir diferentes tipos de errores a los encontrados en los métodos de la Caja Blanca.

Dentro de las técnicas de Prueba de Caja Negra, se utilizó la Partición de Equivalencia. Esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones de

software. Mediante la aplicación de esta se examinó los valores válidos e inválidos de las entradas existentes en el software, descubriendo de forma inmediata clase de errores que, de otro modo, requerirían 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 en número de clases de prueba que hay que desarrollar.

Para definir las clases de equivalencia se tuvieron en cuenta un conjunto de reglas:

- Si una condición de entrada especifica un rango, entonces se confeccionan una clase de equivalencia válida y dos inválidas.
- Si una condición de entrada especifica la cantidad de valores, se identifica una clase de equivalencia válida y dos inválidas.
- Si una condición de entrada especifica un conjunto de valores de entrada y existen razones para creer que el programa trata en forma diferente a cada uno de ellos, se identifica una clase válida para cada uno de ellos y una clase inválida.
- Si una condición de entrada especifica una situación de tipo “debe ser”, se identifica una clase válida y una inválida.

### 4.4.2 Casos de Prueba.

Luego de tener las clases válidas e inválidas definidas, se procedió a definir los casos de pruebas, para los casos de uso más significativos (Diseñar Consulta, Editar Consulta). Estos casos de pruebas fueron desarrollados por el grupo de calidad de CENTALAB para garantizar el correcto funcionamiento del software.

Cada uno de estos casos de pruebas se definió teniendo en cuenta lo siguiente:

- Escribir un nuevo caso de prueba que cubra tantas clases de equivalencia válidas no cubiertas como sea posible hasta que todas las clases de equivalencia hayan sido cubiertas por casos de prueba.

- Escribir un nuevo caso de prueba que cubra una y solo una clase de equivalencia inválida hasta que todas las clases de equivalencias inválidas hayan sido cubiertas por casos de pruebas.

Diseño del caso de prueba correspondiente al caso de uso Diseñar Consulta.

**Descripción General:**

El caso de uso se encarga de diseñar una consulta a partir de las tablas de un modelo.

**Condiciones de Ejecución:**

- El usuario debe estar autenticado.
- Debe seleccionar el vínculo Diseñador de Consultas.

**Secciones a probar en el Caso de Uso:**

| Nombre de la sección | Escenarios de la sección | Descripción de la funcionalidad | Flujo Central |
|---|---|---|---|
| SC 1: " Diseñar Consulta" | EC 1.1: Seleccionar tablas. | El sistema permite seleccionar de los modelos las tablas que conformarán la consulta. | 1. El actor debe desplegar el árbol de modelos, hasta encontrar las tablas que desea.<br><br>2. El actor debe seleccionar cada tabla y arrastrarla hacia el área de trabajo. |

| | | | |
|---|---|---|---|
| | EC 1.2: Seleccionar campos de la consulta. | El sistema muestra cada uno de los atributos de cada tabla donde se selecciona los campos a chequear en la consulta. | 1. El actor selecciona los campos de la consulta.<br><br>2. Cada selección se verá reflejada en el árbol de consulta |
| | EC 1.3: Seleccionar condición de la consulta. | El sistema de la posibilidad de que a cada atributo de una tabla se le pueda adicionar una condición. | 1. El actor selecciona el campo a realizar la condición.<br><br>2. Selecciona la condición que desea realizar.<br><br>3. Cada condición se verá reflejada en el árbol de la consulta. |
| | EC 1.4: Establecer relación entre tablas de un modelo. | El sistema permite dadas las tablas que se encuentran en el área de trabajo establecer relaciones entre ellas. | 1. El actor selecciona una tabla en el área de trabajo.<br><br>2. Desplaza de un punto de esa tabla hacia otra con la cual desea establecer la relación. |
| | EC 1.5: Eliminar relación entre tablas de un modelo. | De las tablas que se encuentran en el área de trabajo y se encuentran relacionadas eliminar dicha relación. | 1. El actor selecciona una relación en el área de trabajo. |

| | | | |
|---|---|---|---|
| | | | 2. El actor presiona la tecla delete y elimina la relación. |

**SC 1:** Diseñar Consulta**.**

| Id del escenario | Escenario | Campos | Opciones | Condición | Relación | Respuesta del Sistema | Resultado de la Prueba |
|---|---|---|---|---|---|---|---|
| EC 1.1 | Seleccionar tablas. | *N/A* | *N/A* | *N/A* | *N/A* | Se adiciona al área de trabajo la tabla seleccionada. | Satisfactorio |
| EC 1.2 | Seleccionar campos de la consulta. | *V* | *N/A* | *N/A* | *N/A* | Se refleja en el árbol de consultas los campos seleccionados. | Satisfactorio |
| EC 1.3 | Seleccionar condición de la consulta. | *V* | *V* | *V* | *N/A* | Se refleja en el árbol de consultas la condición. | Satisfactorio |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| EC 1.4 | Establecer relación entre tablas de un modelo. | *V* | *NA* | *NA* | *V* | El sistema muestra la relación entre tablas con una línea, permitiendo que se establezcan las condiciones del Join y actualizando el árbol de la consulta. | Satisfactorio |
| EC 1.5 | Eliminar relación entre tablas de un modelo. | *V* | *NA* | *NA* | *V* | El sistema actualiza la relación entre tablas eliminando la línea y actualiza el árbol de la consulta. | Satisfactorio |

**Registro de defectos y dificultades detectados**

| Elemento | No | No Conf. | Aspecto | Etapa de detección | Significativa | No Significativa | Recomendación | Estado NC | Respuesta de Equipo Desarrollo |
|---|---|---|---|---|---|---|---|---|---|
| Condición | 1 | La ventana para editar la condición no se puede deslizar. .Ver anexo 8 | CU: Diseñar Consulta | Primera iteración | x | | | PD | NP |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Condición | 1 | El menú para editar el Join aparece detrás de las tablas. Ver anexo 9 | CU: Diseñar Consulta | Primera iteración | x | | | PD | NP |

Diseño del caso de prueba correspondiente al caso de uso Editar Consulta.

**Descripción General:**

Este caso de uso es para modificar las consultas generadas a través del Diseñador de Consultas Dinámicas a gusto del usuario empleando un Editor de Consultas.

**Condiciones de Ejecución:**

- El usuario debe estar autenticado.
- Debe seleccionar el vínculo Editor de Consultas.
- La consulta debe haber sido generada con anterioridad.

| **Nombre de la sección** | **Escenarios de la sección** | **Descripción de la funcionalidad** | **Flujo Central** |
|---|---|---|---|
| SC 1: "Editar Consulta" | EC1.1: Solicitar Editar Consulta. | El sistema muestra el panel de edición de consultas. | 1. Se carga el panel donde se editará la consulta.<br><br>2. El actor edita la consulta. |

| | | | |
|---|---|---|---|
| | EC 1.2: Solicitar Verificar Consulta. | El actor solicita la verificación de la consulta. | 1. El actor presiona la opción verificar consulta.<br><br>2. En caso que la consulta sea editada correctamente se mostrará los datos devueltos por la misma.<br><br>3. En caso que la consulta tenga errores mostrara el tipo de error. |

**SC 1:** Editar Consulta**.**

| **Id del escenario** | **Escenario** | **Consulta Generada** | **Respuesta del Sistema** | **Resultado de la Prueba** |
|---|---|---|---|---|
| EC 1.1 | Solicitar Editar Consulta | *N/A* | El sistema muestra un panel donde se editará la consulta | Satisfactorio |
| EC 1.2 | Solicitar Verificar Consulta. | *V* | El sistema muestra los datos correspondientes a la consulta. | Satisfactorio |

| | | | | |
|---|---|---|---|---|
| | | I | El sistema muestra el error correspondiente a la mala edición de la consulta. | Satisfactorio |

En la siguiente tabla se muestran algunos de los errores encontrados durante la realización de las pruebas de caja negra en varias de las funcionalidades de la aplicación y el tiempo de respuesta por parte del equipo de desarrollo.

| FUNCIONA-LIDAD | ERROR | TIEMPO DE SOLUCIÓN |
|---|---|---|
| Diseñar Consulta | La ventana para editar la condición no se puede deslizar. | 30 min |
| Diseñar Consulta | El menú de editar las relaciones entre tablas aparecía detrás de las mismas. | 4 horas |

**Tabla 4.1:** Errores detectados durante la primera iteración de pruebas.

Durante la primera iteración de pruebas realizadas por el equipo de calidad se detectaron una serie de no conformidades (algunas de ellas se muestran en la tabla 4.1) que luego fueron entregadas al equipo de desarrollo.

## 4.5 Conclusiones.

En este capítulo se abordaron algunas de las diferentes pruebas que se le realizaron al software. Durante todo el proceso de desarrollo se realizaron continuas pruebas de caja blanca, con las que se comprobó el correcto funcionamiento del módulo desarrollado. También se mostraron los resultados arrojados en la aplicación de las pruebas de caja negra por un grupo de especialistas que pertenecen al grupo de calidad de CENTALAD y se comprobó que cumplía con todas las normas, parámetros y estándares establecidos. Por todo lo antes expuesto se puede decir que el software posee una calidad aceptable.

# Conclusiones Generales.

1. Con el desarrollo del trabajo se obtuvo una aplicación web, que permite al usuario diseñar consultas dinámicas para ser usadas por el Sistema Informático Generador de Reportes Dinámicos, dándose cumplimiento al objetivo propuesto.
2. Se realizó el estudio del estado del arte de algunos sistemas diseñadores de consultas, llegando a la conclusión de que ninguno cumple cabalmente con los requisitos establecidos por el grupo de arquitectura del CENTALAD.
3. Se seleccionaron las herramientas y tecnologías a utilizar, siguiendo el principio de software libre.
4. Las tecnologías, herramientas de desarrollo de software y el lenguaje de implementación seleccionado se entendió que eran las adecuadas para el desarrollo del trabajo.
5. El estándar de codificación permitió implementar las capas con un código legible y fácil de entender por cualquier programador que posea conocimientos del lenguaje PHP y Java Script.
6. El módulo Diseñador de Consultas utiliza los patrones de diseños establecidos.
7. El módulo cumple satisfactoriamente con los requisitos que garantizan su funcionamiento.
8. La interfaz gráfica del sistema orientada al usuario es agradable y fácil de operar.

# Recomendaciones.

Luego de la presentación del estudio realizado que culmina con la implementación del módulo Diseñador de Consultas Dinámicas, perteneciente al producto Generador de Reportes Dinámicos, se listan a continuación una serie de recomendaciones para la ampliación, modificación, mejora y construcción de nuevas versiones de este sistema:

- Integrarle un parse SQL (SQLParser), que en estos momentos se encuentra desarrollado, para que pueda actualizarse el modelo a partir del Editor de Consultas.
- Impartir cursos de capacitación a las personas que trabajarán con la aplicación.
- Mantener sobre el sistema un estricto cumplimiento del proceso de mantenimiento y actualización periódica, en un periodo de 6 meses como máximo logrando así que se mantenga la fiabilidad y funcionamiento óptimo del sistema.
- Utilizar el framework Synfony para facilitar la implementación de las capas de los futuros proyectos productivos.

# Referencias bibliográficas.

1. Pentaho Report Designer. [En línea] 2009. http://reporting.pentaho.org/report_designer.php.

2. Active Query Builder .NET Edition. [En línea] 2009 . http://www.activequerybuilder.com/.

3. Visual Database Tools. [En línea] 2009 . http://technet.microsoft.com/es-es/library/ms175485.aspx.

4. Jacobson, Ivar; Booch, Grady y Rumbaugh, James. El Proceso Unificado de Desarrollo de software. — La Habana: Editorial Felix Varela, 2004.

5. Introduction to OpenUP. [En línea] 2009. http://epf.eclipse.org/wikis/openup/.

6. Las aplicaciones web. [2009]. http://www.dimagin.net/es/contenido.php?t_id=6.

7. Alvarez, Miguel Angel. Que es JavaScript.desarrolloweb.com. [En línea] 2009. http://www.desarrolloweb.com/articulos/25.php.

8. Introducción a CSS. [En línea] 2009. http://www.librosweb.es/css/capitulo1.html.

9. Jesús Vegas. El Servidor Web. [En línea] 2009. http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/node20.html.

10. Servidor HTTP Apache 2.0. [En línea] 2009. http://httpd.apache.org/docs/2.0/es/.

11. Internnet Information Server. [En línea] 2009. http://www.htmlpoint.com/iis/index.html.

12. Appserv. [En línea] 2009. http://www.appservnetwork.com/.

13. Active Server Page. [En línea] 2009.http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/.

14. Active Server Page. [En línea] 2009.http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/.

15. Active Server Page. [En línea] 2009.http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/.

16. Active Server Page. [En línea] 2009.http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/.

17. Torre, Aníbal de la. Lenguajes del lado servidor o cliente. [En línea] 2009:http://www.adelat.org/media/docum/nuke_publico/lenguajes_del_lado_servidor_o_cliente.html.

18. PHP. [En línea] 2009. http://www.php-es.com/.

19. Mozilla Firefox. [En línea] 2009. http://www.mozilla-europe.org/es/firefox/

20. Internet Explorer. [En línea] 2009. http://www.microsoft.com/spain/windows/products/winfamily/ie/features.mspx.

21. Sistemas Gestores de Base de Datos. [En línea] 2009. http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html.

22. Microsoft SQLServer. [En línea] 2009. http://www.microsoft.com/spain/sql/default.mspx.

23. MySQL. [En línea] 2009. http://www.desarrolloweb.com/directorio/bases_de_datos/mysql/.

24. Oracle. [En línea] 2009. http://es.wikipedia.org/wiki/Oracle.

25. PostgreSQL. [En línea] 2009. http://www.lawebdelprogramador.com/cursos/mostrar.php?id=72&texto=PostgreSQL.

26. Rational Suite. [En línea] 2009. http://www.vico.org/TallerRationalRose.pdf.

27. Sitio Web oficial Visual-Paradigm. [En línea] 2009.http://www.visual-paradigm.com/product/vpuml/.

28. Notepad++. [En línea] 2009.http://notepad-plus.sourceforge.net/es/site.htm.

29. Notepad++. [En línea] 2009.http://descargar.mp3.es/lv/group/view/kl36301/Notepad%2B%2B.htm.

30. Zend Studio for Eclipse 6.0. [ En línea] 2009.http://www.desarrolloweb.com/articulos/1178.php.

31. Zend Studio for Eclipse 6.0. [En línea] 2009. http://www.zend.com/en/products/studio/.

32 .Visual Studio 2008. [En línea] 2009 .http://www.microsoft.com/spanish/msdn/latam/visualstudio2008/.

33. AJAX. [En línea] 2009. http://es.wikipedia.org/wiki/AJAX.

34. El framework Symfony, una introducción práctica. [En línea] 2009. http://www.maestrosdelweb.com/editorial/el-framework-symfony-una-introduccion-practica-i-parte.

35. Symfony Framework. [En línea] 2009. http: //geekymty.blogspot.com/2007/03/symfony-framework.html.

36. Reynoso, Carlos Billy. Introducción a la arquitectura de software. 2009.

37. Terreros, Julio Casal. Desarrollo de Software basado en Componentes. [En línea] 2009. http://msdn.microsoft.com/en-us/library/bb972268(es-es).aspx.

38. Ídem a Referencia 37.

39. Leon Welicki. Modelo Vista Controlador [En línea] 2009. http://msdn.microsoft.com/en-us/library/bb972251(es-es).aspx.

40. Natalia Juristo, Ana M. Moreno, Sira Vegas. TÉCNICAS DE EVALUACIÓN DE SOFTWARE, Octubre de 2006

41. Natalia Juristo, Ana M. Moreno, Sira Vegas. TÉCNICAS DE EVALUACIÓN DE SOFTWARE, Octubre de 2006

42. Natalia Juristo, Ana M. Moreno, Sira Vegas**.** TÉCNICAS DE EVALUACIÓN DE SOFTWARE, Octubre de 2006

43. Natalia Juristo, Ana M. Moreno, Sira Vegas**.** TÉCNICAS DE EVALUACIÓN DE SOFTWARE, Octubre de 2006

# Bibliografía Consultada.


1. Jacobson, I., Booch, G., Rumbaugh, J. El Proceso Unificado de Desarrollo del Sofware.
2. Tutorial de SQL. Disponible en: http://www.desarrolloweb.com/manuales/9/.
3. LOCKHART, T. PostgreSQL Programmer's Guide. Disponible en: http://www.screenart.es/documentacion/postgres/programmer/.
4. Pressman, Roger. Ingeniería del Software. Un enfoque práctico. Ciudad Habana : Félix Varela, 2005.
5. Lenn, Bass, C., P. and Kazman, Rick. Software Architecture in Practice. s.l.: Addison-Wesley Professional, 2003.
6. Ricardo, Febe Ángel Ciudad. Utilización del patrón modelo – vista – controlador (mvc) en el diseño de software educativo. [En línea] 2007. http://www.informaticahabana.com/evento_virtual/?q=node/223&ev=3er%20Congreso%20Internacional%20de%20Tecnolog%C3%ADas,%20Contenidos%20Multimedia.
7. Documentación del Servidor HTTP Apache 2.0, 2007. Disponible en: http://httpd.apache.org/docs/2.0/es/.
8. FRANCO NAVARRO, J. A. UML en acción. Modelando Aplicaciones Web, 2005.
9. GALLEGO VÁZQUEZ, J. A. Desarrollo Web con PHP y MySQL. S.A, A. M., 2003.
10. EGUILUZ PEREZ, J. Introducción a AJAX, 2009. Disponible en: http://www.librosweb.es/ajax/.
11. BRADENBAUGH, J. Aplicaciones JavaScript. S.A., A. M., 2000.
12. BRUEGGE, B. D., A. Ingeniería de Software Orientado a Objetos. Prentice Hall-Pearson
13. Descripción de la arquitectura en módulos del Apache, 2007. Disponible en: http://www.desarrolloweb.com/articulos/1112.php.
14. EMS soluciones, 2007.Disponible en: http://www.sqlmanager.net/.

15. FRANCO NAVARRO, J. A. UML en acción. Modelando Aplicaciones Web, 2005.

16. GALLEGO VÁZQUEZ, J. A. Desarrollo Web con PHP y MySQL. S.A, A. M., 2003.

17. EGUILUZ PEREZ, J. Introducción a AJAX, 2007. Disponible en: http://www.librosweb.es/ajax/.

18. LARMAN, C. UML y Patrones. Prentice Hall Iberoamericana. 1999. p.

19. Manual de referencia de MySQL 5.0, 2007 (revisión 488). Disponible en: http://www.calitae.com/manuales/manual-mysql-5.0-es.pdf.

20. Manual de PHP, 2006, Disponible en: http://www.php-es.com/.

21. MYSQL, 2007. Disponible en: http://mysql.conclase.net/curso/index.php.

22. PHP, Disponible en: http://www.php-es.com/.

23. Rational Suite, 2007. Disponible en: http://www.vico.org/TallerRationalRose.pdf.

24. Programación en capas. 2009. Slideshare. [En línea] 2009. http://www.slideshare.net/Decimo/arquitectura-3-capas.

25. Cornejo, José Enrique González. 2001. DocIris. [En línea] 25 de marzo de 2001. http://www.docirs.cl/arquitectura_tres_capas.htm.

26. Jacobson, I., Booch, G., Rumbaugh, J. El Proceso Unificado de Desarrollo del Sofware.

27. Juristo, N., M.Moreno, A., Vegas, S. TÉCNICAS DE EVALUACIÓN DE SOFTWARE.

# Anexos

**Anexo 1: Consulta realizada sobre el diseñador de consultas del (Pentaho Report Designer 1.7.0).**

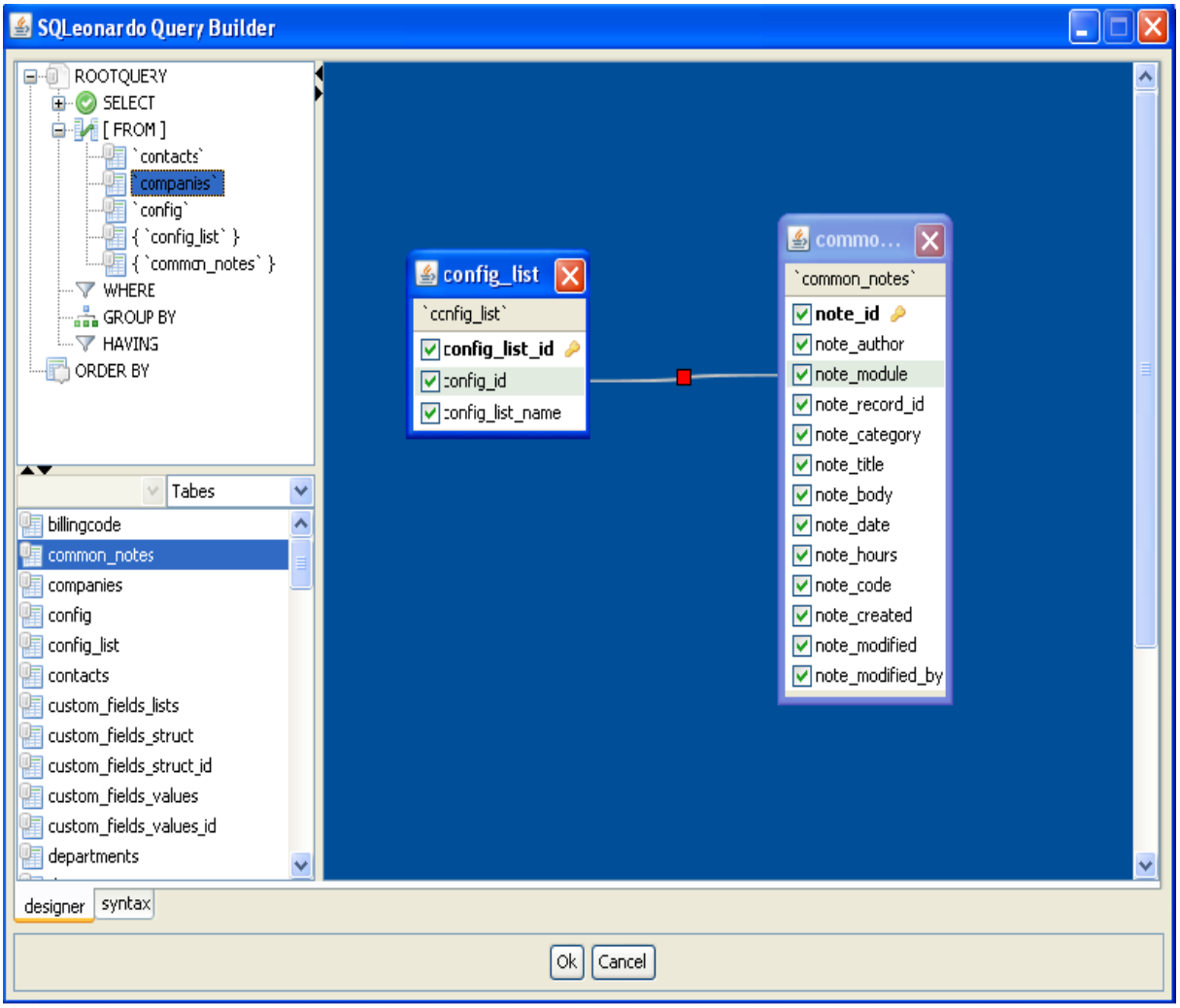


**Fig. 1: El diseñador de consultas del (Pentaho Report Designer 1.7.0).**

**Anexo 2: Consulta realizada sobre el editor de consultas del (Pentaho Report Designer 1.7.0).**

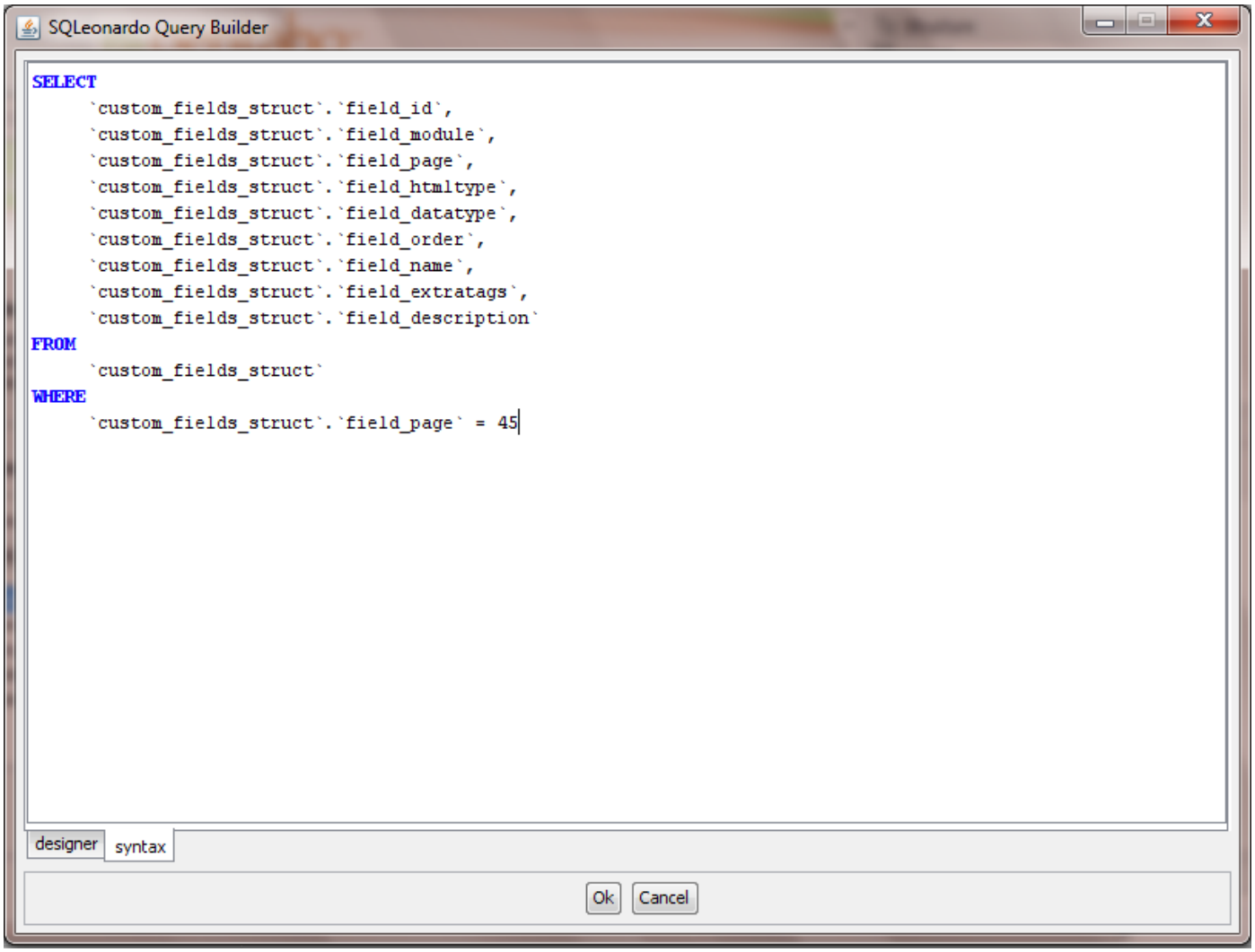


**Figura2: El editor de consultas del (Pentaho Report Designer 1.7.0).**

**Anexo 3: Consulta realizada empleando El "Active Query Builder .NET Edition".**

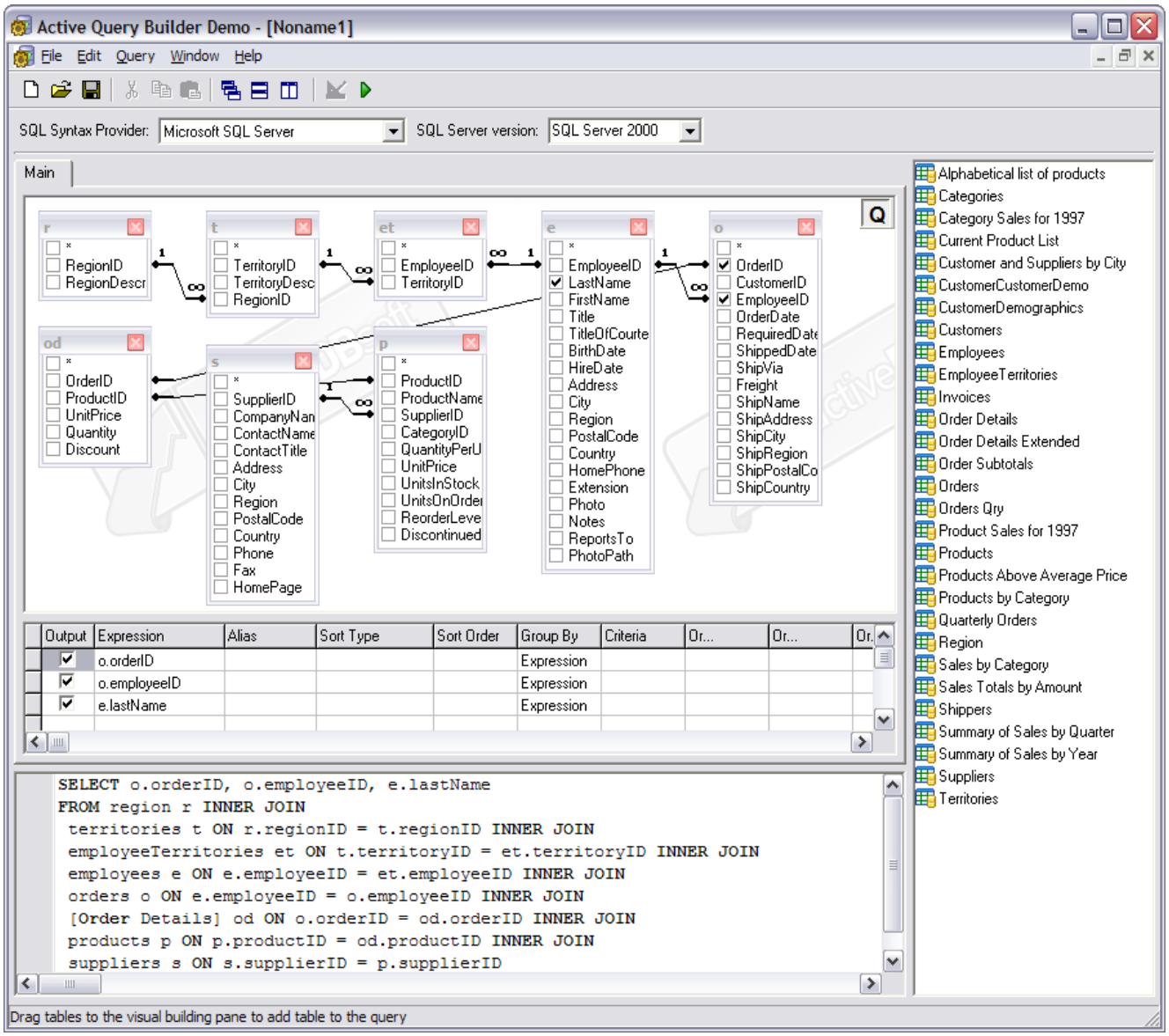


**Fig. 3: El "Active Query Builder .NET Edition".**

**Anexo 4: Consulta realizada en la herramienta "Visual Database Tools".**

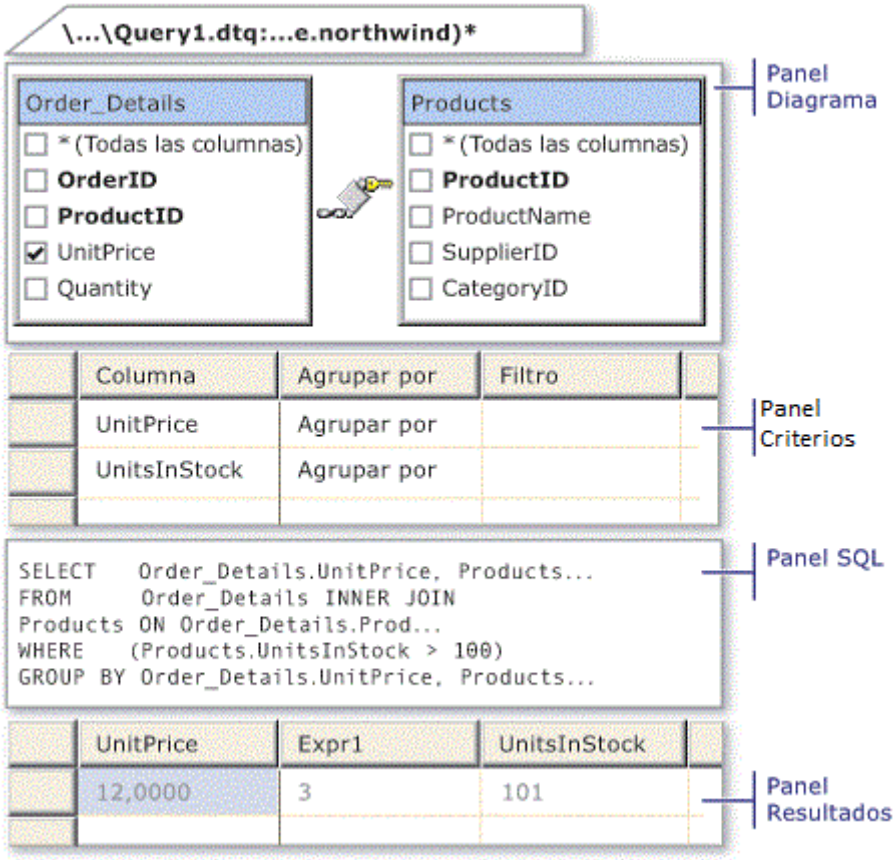


**Figura .4 El "Visual Database Tools".**

**Anexo 5: Estándar de codificación.**

1. **Nombre de las Aplicaciones.**

   Los nombres de las aplicaciones deben permitir que con solo leerlo se conozca el propósito del proceso a automatizar.

   Ejemplo:

   Diseñador de Consultas.

2. **Nombre de los formularios.**

   Comenzarán con el prefijo frm, seguido del nombre y en caso que este sea compuesto se empleará la notación UpperCamelCase.

   Ejemplo:

   <form name="frmMiFormulario"></form>

3. **Prefijos para controles HTML.**

| **Prefijos** | **Control** | **Elemento** | **Tipo** |
|---|---|---|---|
| btn | Button | INPUT | button |
| cb | Checkbox | INPUT | checkbox |
| cmb | Combobox | INPUT | combobox |
| ff | File field | INPUT | file |
| hdn | Hidden | INPUT | hidden |
| psw | Password | INPUT | password |

| | | | |
|---|---|---|---|
| rdb | Radio button | INPUT | radio |
| rsb | Reset button | INPUT | reset |
| sb | Submit button | INPUT | submit |
| tf | Text field | INPUT | text |
| ta | Text area | INPUT | Text |
| ddw | Dropdown | SELECT | |
| lbx | Listbox | SELECT | |
| Img | Image | | |

## 4. Clases y objetos.

Los nombres de las clases y los objetos deben permitir que con sólo leerlos, se reconozca el propósito de la misma.

### 4.1. Atributos de las clases.

Para el nombre de los atributos de una clase se utilizará la notación lowerCamelCase. Si es una sola palabra se mantiene todo en minúscula. Permite con sólo leerlo, reconocer el propósito del mismo dentro de la clase.

Ejemplo:

```
$cantTotal;

$nombre;

class cbdClase

{

  $nombreTabla; //el atributo denota el nombre de una tabla

}
```

### 4.2. Funciones.

Los nombres de las funciones utilizarán la notación lowerCamelCase. Con sólo leerlo se reconoce el propósito dentro de la función.

Ejemplo:

```
function buscarUnidad()
{
    //instrucciones
}
```

### 4.3. Funciones que obtienen o fijan datos.

Funciones que obtienen datos, colocar el prefijo **get** y si fijan algún valor emplear el prefijo **set**.

Ejemplo:

```
function getUnidad()
{
    //instrucciones
}
```

```
function setFicha()
```

```
{

    //instrucciones

}
```

### 4.4. Variables y constantes.

Variables de referencia comienzan con el prefijo "ref" en minúscula. El resto del nombre de la variable se escribe igual que las variables miembro de la clase.

Ejemplo:

```
$entero = 5;

&refUnEntero;

&refUnEntero = $entero;
```

### 4.5. Variables con nombres compuestos.

En los nombres de las variables se utilizará la notación lowerCamelCase.

Ejemplo:

```
$cantTotal;  //en el caso de un nombre compuesto

$nombre;     //en el caso de un nombre simple
```

### 4.6. Nombre de clases.

En los nombres de las clases se utilizará la notación UpperCamelCase ejemplo “Parameters” y en caso de que la clase sea compuesta “MainModel”.

### 4.7. Constantes.

Se declaran con todas sus letras en mayúscula y una constante por cada línea.

Ejemplo:

```
define("CONSTANT1","value1");

define("CONSTANT2","value2");
```

### 4.8. Indentación.

Nunca se debe usar tabulaciones en el código. Los saltos de la indentación serán a 2 espacios.

Ejemplo:

```
<?php

class sfFoo

{

  public function bar()

  {

    sfCoffee::make();

  }

}
```

### 4.9. Comentarios.

Los comentarios del código se deben localizar en sus líneas correspondientes y en el siguiente formato.

// space first, with no full stop needed

Todas las funciones y métodos de las clases deben contar con su propio bloque phpdoc, el cual será de utilidad a la hora de generar la documentación.

```
<?php

/**

 * Notifies all listeners of a given event.

 *

 * @param  sfEvent  $event  A sfEvent instance

 *

 * @return sfEvent          The sfEvent instance

 */

public function notify(sfEvent $event)
```

### 4.10. Líneas en blanco.

Se utilizan líneas en blanco antes de la definición de cada función para dar claridad al código.

Ejemplo:

```
function function1()

{

}


function function2()

{

}
```

### 4.11. Espacios en blanco entre operadores lógicos y aritméticos.

Se usan espacios en blanco entre los operadores lógicos y aritméticos para lograr una mayor legibilidad en el código.

Ejemplo:

```
$tabla = ‘nom_producto’;

if (($tabla) && ($fields))
```

**Anexo 6: Representación gráfica de las pruebas de caja blanca y caja negra.**

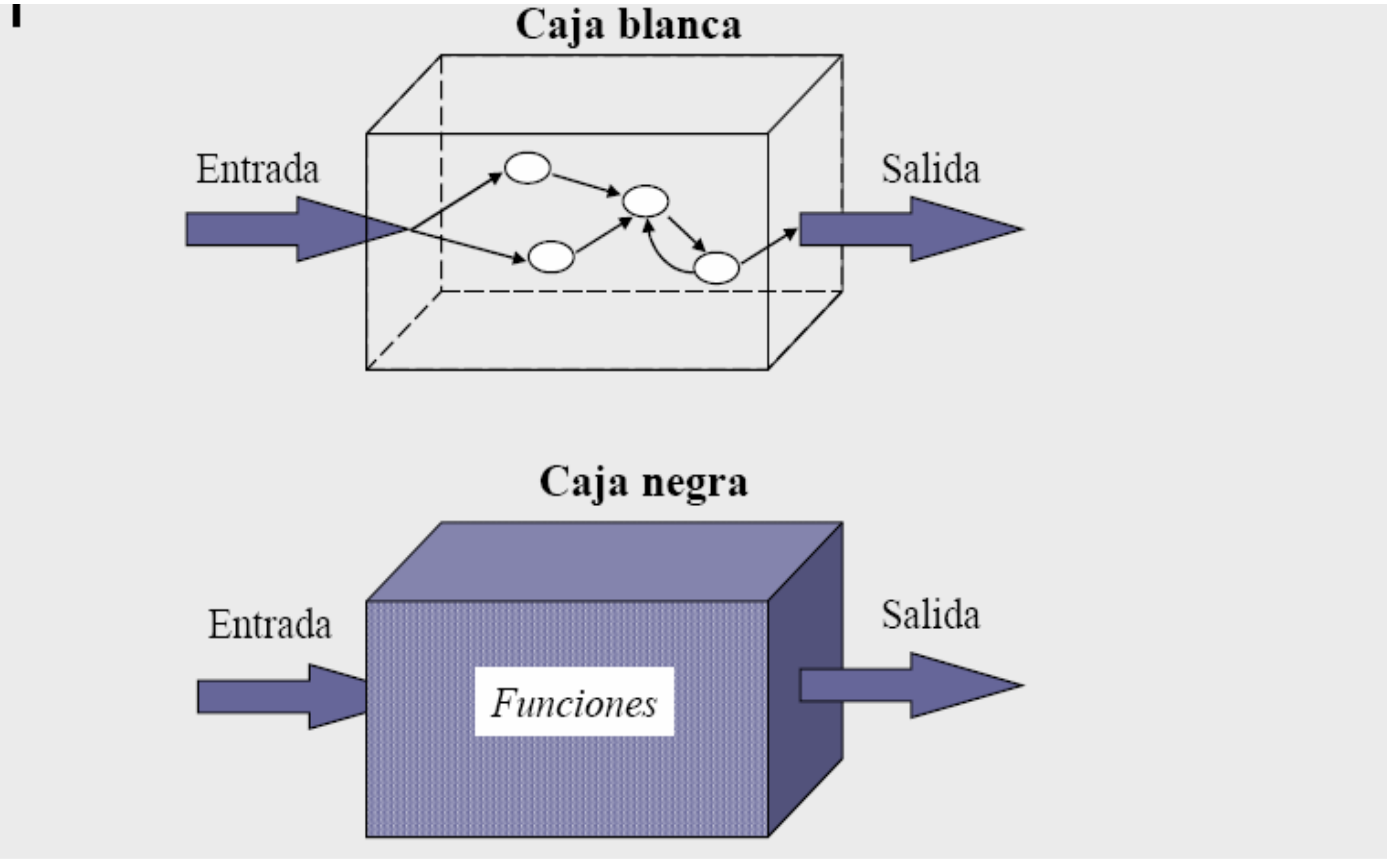


**Figura.5 Representación de pruebas de Caja Blanca y Caja Negra.**

**Anexo 7: Representación en un grafo de flujo de las estructuras lógicas de un programa.**

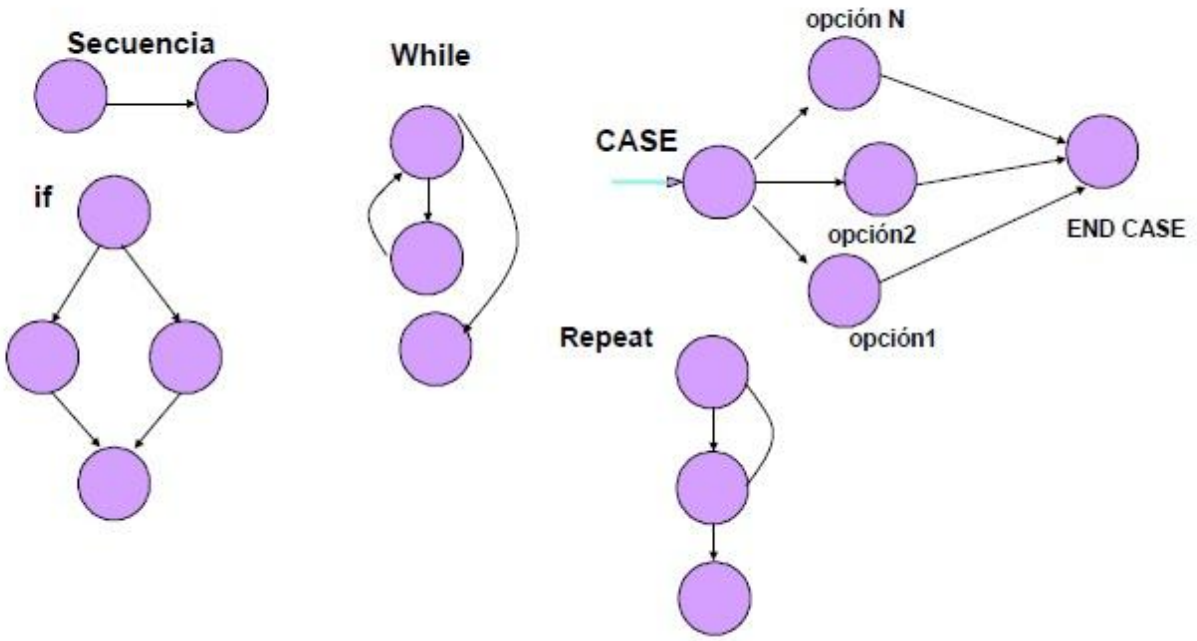


**Figura.6 Grafo de flujo de las estructuras lógicas de un programa.**

**Anexo 8: Error arrojado durante una prueba de caja negra al editar la condición sobre un campo de una tabla.**

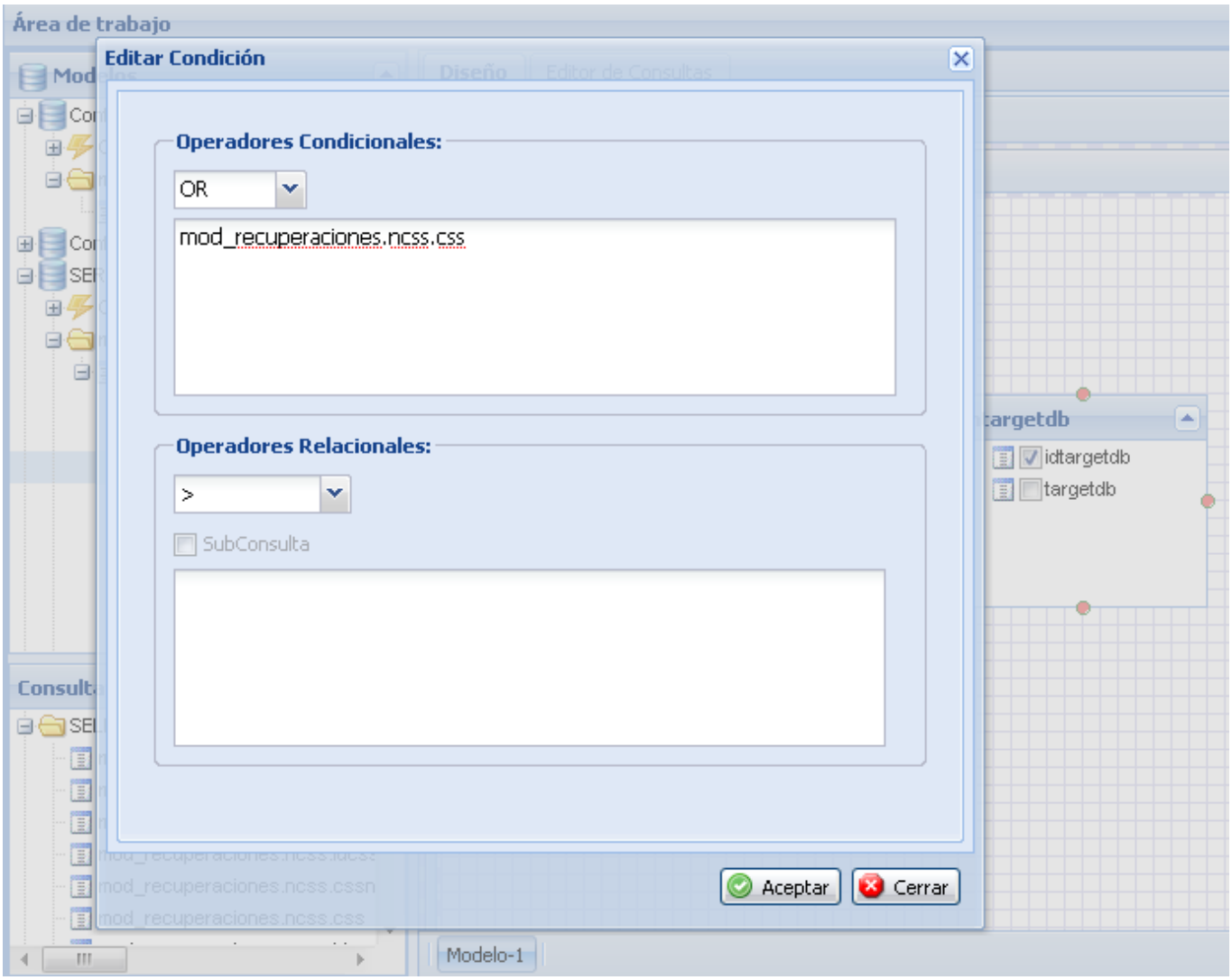


**Figura.7 Interfaz empleada para editar la condición sobre un campo de una tabla.**

**Anexo 9: Error arrojado durante una prueba de caja negra al editar la condición que permite combinar registros de dos o más tablas.**

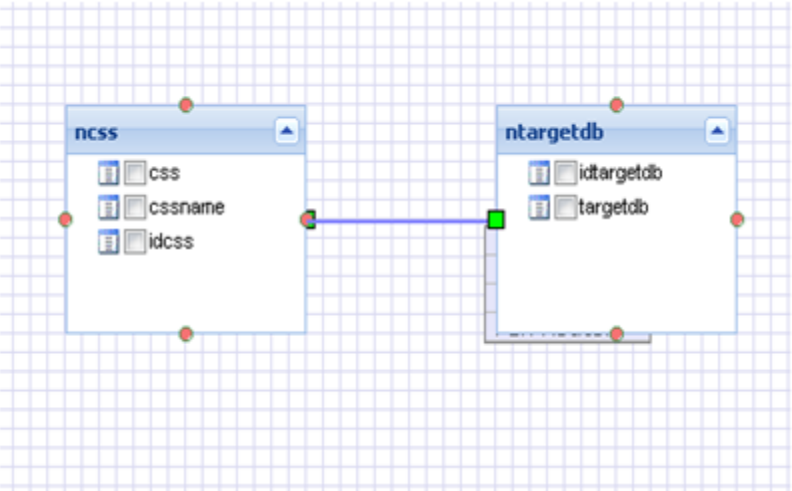


**Figura.8 Interfaz empleada para editar la condición Join.**

# GLOSARIO DE TÉRMINOS

Framework: Es una estructura de soporte definida en la cual un proyecto de software puede ser organizado y desarrollado.

C#: Es un lenguaje de programación orientado a objetos desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, que después fue aprobado como un estándar por la ECMA e ISO.

ASP.NET: Es un framework para aplicaciones web desarrollado y comercializado por Microsoft. Es usado por programadores para construir sitios web dinámicos, aplicaciones web y servicios web XML.

TCP/IP: Conjunto de protocolos de red en la que se basa Internet y que permiten la transmisión de datos entre redes de computadoras.

Servlet: Es un objeto que se ejecuta en un servidor o contenedor JEE, fue especialmente diseñado para ofrecer contenido dinámico desde un servidor web.

IDE: Entorno Integrado de Desarrollo, es un programa compuesto por un conjunto de herramientas para un programador. Puede dedicarse exclusivamente para un lenguaje de programación o bien para varios.

AJAX: Es una técnica de desarrollo para crear aplicaciones web, mediante la cual un grupo de acciones se ejecutan en navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano.

Alias: nuevo nombre que recibe un campo de una base de datos .

API: Interfaz de Programación de Aplicaciones es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

Aplicación Web: Sistema informático utilizado por los usuarios mediante el acceso a un servidor web a través de internet o de una intranet.

Etiqueta: marca con tipo que delimita una región en los lenguajes basados en XM; Dentro de un documento de hipertexto, una instrucción. Una etiqueta de hipertexto empieza con un carácter "<" y termina con un carácter ">".

HTTP: Protocolo de Transmisión Hipertexto, protocolo de comunicaciones utilizado por los programas clientes y servidores de WWW para comunicarse entre sí.

Script: guión o conjunto de instrucciones que permiten la automatización de tareas creando pequeñas utilidades.

Servidor: computador en el que se ejecuta un programa que realiza alguna tarea en beneficio de otras aplicaciones llamadas clientes, tanto si se trata de un ordenador normal, un ordenador central, un miniordenador, un ordenador personal ó un sistema integrado.

SGBD: Sistema Gestor de Base de Datos, tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan; se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta.

SQL: Lenguaje de Consulta Estructurado, lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones sobre las mismas.

UML: Lenguaje de Modelado de Sistemas de Software, es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software, se usa para entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

URL: Localizador Uniforme de Recurso, secuencia de caracteres, de acuerdo a un formato estándar, que se usa para nombrar recursos, como documentos e imágenes en Internet, por su localización.

W3C: siglas de World Wide Web Consortium, consorcio internacional fundado en 1994, donde las organizaciones miembro, personal a tiempo completo y el público en general, trabajan conjuntamente para desarrollar estándares Web.

Componente de Software: son todo aquel recurso desarrollado para un fin concreto y que puede formar solo o junto con otro/s, un entorno funcional requerido por cualquier proceso predefinido. Son independientes entre ellos, y tienen su propia estructura e implementación.

Herramientas de Software: aplicaciones que ayudan en el trabajo a los humanos.

Aplicación: es un tipo de programa informático diseñado como herramienta para permitir a un usuario realizar un o diversos tipos de trabajo.

SELECT: recupera filas de la base de datos y habilita la selección de una o varias filas o columnas de una o varias tablas.

FROM: es obligatoria en todas las instrucciones SELECT en las que se estén recuperando datos de tablas o vistas específicas la tabla sobre la que se realizarán las operaciones.

JOIN: permite combinar registros de dos o más tablas en una base de datos relacional. En el Lenguaje de Consultas Estructurado (SQL), hay tres tipos de JOIN: interno, externo, y cruzado.

WHERE: Especifica una condición de búsqueda para restringir el número de filas devueltas.

GROUP BY: La cláusula GROUP BY en SQL permite realizar agrupaciones de registros de una tabla.

HAVING: Especifica una condición de búsqueda para un grupo o agregado. HAVING sólo se puede utilizar con la instrucción SELECT. Normalmente, HAVING se utiliza en una cláusula GROUP BY. Cuando no se utiliza GROUP BY, HAVING se comporta como una cláusula WHERE.

ORDER BY: especifica el orden utilizado en las columnas devueltas en una instrucción SELECT. La cláusula ORDER BY no es válida en vistas, funciones insertadas, tablas derivadas ni subconsultas, salvo que se especifique también TOP.

TOP: especifica que sólo se devolverá el primer conjunto de filas del resultado de la consulta. El conjunto de filas puede ser un número o un porcentaje de las filas. La expresión TOP se puede usar en instrucciones SELEC.

Action: ejecuta toda la lógica para una petición ejecutada.

Workflow: ventana sobre la cual se realizara el diseño de la consulta. Permite desplazar objetos de tipo tabla sobre él y establecer relaciones entre ellos.