Universidad de las Ciencias Informáticas

Facultad 4

Título:

Subsistema de Configuración del Generador de Recuperaciones Dinámicas para aplicaciones Web.

Trabajo de Diploma para optar por el título de
Ingeniero Informático

Autor(es): *Armando Javier Pérez Armas*
Andy Vidal Martínez Borges
Tutor: *Ing. Yanet Pérez Valcárcel*

Ciudad de la Habana, julio de 2008

DECLARACIÓN DE AUTORÍA

Declaramos que somos los autores de este trabajo y autorizamos al Ministerio de las Fuerzas Armadas Revolucionarias (MINFAR) y a la Universidad de Ciencias Informáticas (UCI) para que hagan el uso que estimen pertinente con él.

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

_____	_____	_____
Firma del Autor	Firma del Autor	Firma del Tutor
Armando Javier Pérez Armas	Andy Vidal Martínez Borges	Yanet Pérez Valcárcel

DATOS DE CONTACTO

TUTOR: Ing. Yanet Pérez Valcárcel. Profesora graduada como Ingeniera Informática en la Universidad de Las Ciencias Informáticas en el año 2007. Ha impartido las asignaturas de las Ciencias Empresariales como son: Administración de Empresas, Comercio Electrónico y Contabilidad y Finanzas.

Email: yperezva@uci.cu

AGRADECIMIENTOS

Agradezco hoy y siempre a mi familia por apoyarme y motivarme para lograr mis objetivos, a mis compañeros por su ayuda sin la cual no hubiera llegado hasta aquí

Armando Javier

Agradezco principalmente a mi familia, en especial a mis padres y a mi hermano, muchas gracias por el apoyo que me brindaron. A mis amigos y compañeros de estudios que compartieron conmigo en las buenas y en las malas, en fin a todos aquellos que de una forma u otra contribuyeron en el desarrollo de este trabajo.

Andy Vidal

DEDICATORIA

Para quienes son y serán por siempre nuestra fuente de inspiración

...nuestros Padres y seres queridos.

RESUMEN

En la actualidad la información es uno de los tesoros más importantes que tiene una entidad u organización, por lo que con el aumento del desarrollo tecnológico se han desarrollado diferentes sistemas de gestión con el fin de controlar y gestionar este activo tan valioso como es la información.

Por supuesto el acceso a la información es extremadamente controlado, pero si el usuario tiene permiso para acceder a la información se hace necesario brindársela de forma rápida y objetiva, agilizando así la toma de decisiones, y aquí es donde cumple su función los recuperadores de información.

Ahora bien, siendo así, estos recuperadores de información son necesarios por lo que en este trabajo dará una respuesta óptima al aspecto de recuperar información evitando el gasto de recursos y de tiempo, al desarrollar un recuperador de información que sea configurable para cualquier Base de Datos.

PALABRAS CLAVE

- ✓ Configurable.
- ✓ Recuperador
- ✓ Sistema de Gestión

ÍNDICE

1.1 Introducción

La información, es uno de los más importantes activos no solo para las empresas y organizaciones. La información existente en el mundo va en un aumento creciente y constante , cada vez más amplia, y al desarrollarse la tecnología se ha logrado almacenarla en diferentes formatos, pero independientemente del formato es necesario que se encuentre a disposición de los usuarios; por lo que es necesario crear aplicaciones de software cuyo objetivo fundamental sea realizar recuperaciones de información, lográndose así que el acceso de forma rápida y objetiva a los datos almacenados en Base de datos, ficheros, etc.

Para lograr una aplicación que cumpla con este objetivo de recuperar información y que sea capaz de acoplarse a diferentes sistemas, se hace imprescindible una buena configuración, porque todo software para su buen funcionamiento necesita una correcta configuración que le permitirá satisfacer las necesidades del usuario y cumplir su objetivo que en este caso sería recuperar información de forma rápida y precisa.

Hoy por hoy con el gran aumento y desarrollo de la tecnología se han creado todo tipo de sistemas de recuperación de información, pero la gran mayoría viene de forma intrínseca a una aplicación determinada, de la cual va a recuperar información. No se puede olvidar que hay recuperadores dinámicos

que son configurables para diversas aplicaciones como son el JasperReport, AgataReport y otros que son configurables para cualquier Base de Datos pero no cumplen con algo que es imprescindible para las necesidades del país en estos momentos y es que son software propietario, por lo que se hace necesario la creación de un recuperador de información desarrollado en software libre que sea adaptable a cualquier Base de datos.

Después de haber hecho un análisis de lo vital que resulta la información y disponibilidad para el usuario se puede determinar que el **problema a resolver** es: La no existencia de una herramienta flexible que permita configurar una aplicación de recuperación dinámica de información y sus formas de acceso.

Las actividades realizadas están enfocadas a garantizar la disponibilidad y accesibilidad adecuada de la información sin olvidar la seguridad de la misma, implementando un modelo que sea adaptable controle los accesos a los datos almacenados teniendo en cuenta las restricciones de acceso.
Por tanto el **objeto de estudio** del presente trabajo**:** Incluye los Sistemas de configuración de la recuperación de información existentes.

De lo anterior se deduce que el **campo de acción** serian Los procesos de configuración para la recuperación de la información existente en las Bases de Datos**.**

A manera de **hipótesis,** partir de la idea que si se contara con una herramienta, basada en la configuración de la recuperación dinámica de información; mejoraría en dinamismo y rapidez el acoplamiento a las aplicaciones y a la vez servir como soporte para el buen funcionamiento del mismo.
. Por ello el **objetivo general** será: Implementar una herramienta que permita la configuración de la información existente en los Sistemas de Gestión de Base de Datos de manera dinámica.

De acuerdo con lo que se ha expuesto hasta el momento se puede inferir que nuestro **objetivo específico** será: la implementación de un sub-módulo para controlar la distribución de los permisos para recuperar determinada información.
Para cumplir con los objetivos trazados y resolver el problema planteado, se proponen las siguientes **tareas**:

- Realizar un estudio del diseño de los diferentes casos de uso que conforman el paquete de configuración.
- Realizar un estudio sobre el lenguaje de programación para la implementación, y ofrecer ventajas y desventajas del mismo.
- Realizar un estudio sobre la arquitectura de desarrollo, así como una detallada descripción de la que se emplee finalmente.
- Implementar los diferentes casos de uso definidos y documentar los componentes utilizados.

La principal ventaja del módulo es que logra gestionar el estado y garantizar el acceso a la información de manera flexible y configurable para la recuperación dinámica de información.

Resultados: Una herramienta flexible de configuración para la recuperación dinámica de información.

Este documento cuenta con 3 capítulos en los cuales se describe todo el proceso de desarrollo para la obtención del producto final. Además de presentar Introducción, Conclusiones, Recomendaciones, bibliografía, anexos y glosario de términos.

El Capítulo I titulado “Fundamentación teórica” ofrece los conceptos básicos asociados al dominio del problema. Además brinda el estado del arte en cuanto a los antecedentes históricos del sistema, así como las técnicas, tecnologías, metodologías y software usados en la actualidad o en las que se apoya para la solución del problema que se enfrenta.

El Capítulo II denominado “Descripción y análisis de la solución propuesta”: se plantea una valoración crítica del diseño propuesto por el analista, un análisis de posibles implementaciones de componentes o módulos ya existentes que puedan ser rehusados y las estrategias de integración, una descripción de los algoritmos no triviales a implementar. Análisis de complejidad

de los mismos y selección de las estructuras de datos apropiadas para la implementación de estos algoritmos, finalizando con la descripción de las nuevas clases u operaciones necesarias.

En el Capítulo III "Validación de la solución propuesta": se muestran las pruebas realizadas para validar la solución propuesta.

1. CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

Este capítulo está dedicado a realizar un análisis detallado del estado del arte de los distintos sistemas especializados que existen en el campo de la recuperación de información tanto nacional como internacional, así como las distintas técnicas de programación que existen. También se analizarán las tendencias, técnicas, tecnologías, metodologías relacionadas con dichas técnicas y plataformas de desarrollo que la soportan.

1.2. Sistemas informatizados existentes vinculados al campo de acción:

A nivel mundial existen diversos gestores para la recuperación, uno de los más comunes es el Microsoft Query que esta incluido en el paquete de herramientas de la compañía Microsoft. También se puede encontrar herramientas más potentes para realizar recuperaciones como el Cristal Report, el AgataReport y el JasperReport. Pero ninguno de ellos tiene un modulo de configuración que los adapte o acople a otros sistemas, por otro lado existen otro configuradores pero que no se ajustan a las necesidades que tiene nuestro país en especial el MINFAR para resolver de una manera compacta y única el problema de las configuraciones.

JasperReports creado por Jaspersoft es una poderosa herramienta de creación de informes Java open source que tiene la habilidad de entregar contenido rico en el monitor, a la impresora o a ficheros PDF, HTML, XLS, CSV y XML.

Está escrito completamente en Java y puede ser usado en gran variedad de aplicaciones de Java, incluyendo J2EE o aplicaciones Web, para generar contenido dinámico.

Su propósito principal es ayudar a crear documentos de tipo páginas, preparados para imprimir en una forma simple y flexible.

JasperReports se usa comúnmente con iReports, un front-end gráfico open source para la edición de informes.

Es una herramienta multiplataforma.

Su última versión es la 1.3.0.

Está publicado bajo la licencia: Common Public License.

Requerimientos de JasperReports:

· Se requiere tener instalado en el equipo el JDK 1.4 (SDK) o posterior. No basta con tener instalado el J2RE (Run Time Environment).

Versat Sarasola

En nuestro país se llevo a cabo en 1998 un sistema contable llamado Versat Sarasola, el cual es un sistema económico integrado. Constituido por 12 módulos que incluyen configuración y seguridad, contabilidad general y de gastos, costos y procesos, análisis económico empresarial y control de activos fijos.

Este sistema financiero tiene entre sus 10 subsistemas uno que se encarga de la configuración.
Este subsistema de configuración define sus usuarios por unidades contables pero la definición de estos usuarios solo la puede efectuar un usuario del Versat con los derechos de Administrador del Sistema.
Hay diferentes niveles de acceso dependiendo del subsistema en que se encuentre en ese momento:
Especialista principal
Especialista en Contabilidad
Contador y Operador

Especialista en Costos
Especialista en Inventario
Consultor
Como se puede ver Versat tiene incluso un subsistema que se encarga de la configuración, e incluso fueron mas allá al hacer este sistema adaptable a casi cualquier entidad del país , su sistema de configuración viene internamente en el software a configurar y solo lo configura a el, por lo cual no cumple con las expectativas definidas por nuestros especialistas.

Otro de los sistemas que esta a nuestro alcance que tiene cierta semejanza o que de una forma u otra recupera información dinámicamente, que además esta implementado en PHP, usando gestor de base de datos PostgresSql y como servidor Web, Apache, es el **Sistema Integral de Gestión de Recursos Humanos (GREHU)**, el cual gestiona la información de recursos humanos a nivel base. En el momento en que se comenzó a confeccionar este sistema, la empresa no contaba con un mecanismo que le brindara la facilidad de una herramienta capaz de integrar de forma dinámica las informaciones referentes a su personal con el fin de tributarla a los niveles corporativos. El Módulo de Recuperación va a ser el encargado de recuperar la información contenida en la Base de Datos del GREHU que se encuentran en las empresas bases, consolidando y exportando las informaciones recuperadas de manera tal que cada uno de los niveles inmediatos superiores puedan cargar toda aquella información referente a sus empresas bases y almacenarlas en su nivel correspondiente.
Este módulo se realizará con la tecnología **Data Warehouse** la que permite consolidar y almacenar datos de variadas fuentes con el propósito de responder preguntas de negocios y apoyar al proceso de toma de decisiones.
La herramienta **GREHU** es un software que se encuentra implantado en gran variedad de empresas de nuestro país. Cuenta con varios módulos o subsistemas como: Inventario de Personal, Selección y contratación, Puesto y case, Evaluación del desempeño, Prenóminas, Nóminas, Pago por resultados, entre otros.
Algunas empresas que usan este GRH del cual forma parte GREHU son Aguas de la Habana, Islazul y Unión Cuba-Petróleo (CUPET).

El configurador conocido como **Measurement & Automation Explorer**, presenta una vista unificada al sistema de hardware de medición soportado por el software de control de servicios.

Con Measurement & Automation Explorer los usuarios pueden definir el nombre de canales para organizar señales o especificar funciones de escalamiento para convertir señales digitalizadas a cantidades medidas.
El beneficio clave del configurador es la integración con los ADEs, como Microsoft Visual Basic y Visual C++ y NI LabVIEW y Measurement Studio. Esta integración le da a los desarrolladores la habilidad de fácilmente integrar múltiples mediciones en una sola aplicación sin programación. Sin estas herramientas de configuración, los desarrolladores deben pasar mucho tiempo configurando las funciones de medición de manera programática.

1.3. Nuestro Software.

En el presente trabajo se evidencia como objetivo fundamental la implementación de un software para la recuperación de la información almacenada en las bases de datos. Puede ser utilizado en un sistema operativo como Windows o libre como Linux y creado por herramientas libres, como son los casos de las herramientas que se van a utilizar tales como: PHP, PostgreSQL, Eclipse las cuales se explican a continuación. Con la utilización de estas herramientas se esta reduciendo el costo del proyecto contribuyendo así a las inversiones del país.

1.4. Tecnologías.

Constituye un objetivo fundamental de los diseñadores de software alcanzar y mantener un nivel técnico acorde con el desarrollo actual en la automatización de la información para la gestión de cualquier proceso a desarrollar, para lo cual es necesario hacer un estudio detallado de las tecnologías a utilizar y las posibilidades de desarrollo que estas brindan, así como los conceptos ligados a estas. A continuación en este epígrafe se describe los principales conceptos, tecnologías y herramientas propuestas para el desarrollo de la solución tratada en el trabajo.

1.4.1 Arquitectura 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.

Es una arquitectura de procesamientos cooperativos, donde uno de los componentes pide servicios a otro, existiendo una colaboración entre dos o más computadoras conectadas a una red.

IBM define al modelo Cliente/Servidor como: "Es la tecnología que proporciona al usuario final el acceso transparente a las aplicaciones, datos, servicios de cómputo o cualquier otro recurso del grupo de trabajo y/o, a través de la organización, en múltiples plataformas. El modelo soporta un medio ambiente distribuido en el cual los requerimientos de servicio hechos por estaciones de trabajo inteligentes o clientes, resultan en un trabajo realizado por otros computadores llamados servidores".

Entre las principales características de la arquitectura Cliente/Servidor, se pueden destacar las siguientes:

- 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.
- Centralización del control: los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema.
- Escalabilidad: se puede aumentar la capacidad de clientes y servidores por separado.
- Se reduce el tráfico de red considerablemente. Idealmente, el cliente se comunica con el servidor utilizando un protocolo de alto nivel de abstracción.

1.4.2. Técnicas de programación.

1.4.2.1 Programación no Estructurada.

La programación no estructurada está basada en una secuencia de instrucciones modificando los datos globales en el transcurso de todo el programa. Donde los saltos y el fin de programa no seguían ninguna estructura. Los saltos podían apuntar a cualquier punto del código, lo que ocasionaba que el algoritmo terminara siendo un poco indescifrable. Tampoco se podía saber cuándo terminaba.

1.4.2.2. Programación Procedimental.

La programación procedimental es un tipo de programación estructurada en donde el código se divide en porciones llamadas "procedimientos" o "funciones".

1.4.2.3. Programación Modular.

Esta programación basada en la técnica de diseño descendente, consiste en dividir el problema original en diversos sub-problemas que se pueden resolver por separado, para después recomponer los resultados y obtener la solución al problema.

1.4.2.4. Programación Orientada a Objetos.

La Programación Orientada a Objetos (POO) es un paradigma de programación que define los programas en términos de "clases de objetos", objetos que son entidades que combinan estado (datos), comportamiento (procedimientos o métodos) e identidad (propiedad del objeto que lo diferencia del resto). La programación orientada a objetos expresa un programa como un conjunto de estos objetos, que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener y reutilizar. De esta forma, un objeto contiene toda la información, (los denominados atributos) que permite definirlo e identificarlo frente a otros objetos

pertenecientes a otras clases (e incluso entre objetos de una misma clase, al poder tener valores bien diferenciados en sus atributos).

A su vez, dispone de mecanismos de interacción (los llamados métodos) que favorecen la comunicación entre objetos (de una misma clase o de distintas), y en consecuencia, el cambio de estado en los propios objetos. Esta característica lleva a tratarlos como unidades indivisibles, en las que no se separan (ni deben separarse) información (datos) y procesamiento (métodos).

Llegando a este punto se pueden exponer algunas de las principales ventajas que presenta el enfoque orientado a objetos sobre las demás tendencias:

- **Uniformidad**. Ya que la representación de los objetos lleva implícita tanto el análisis como el diseño y la codificación de los mismos.

- **Comprensión**. Tanto los datos que componen los objetos, como los procedimientos que los manipulan, están agrupados en clases, que se corresponden con las estructuras de información que el programa trata.

- **Flexibilidad.** Al tener relacionados los procedimientos que manipulan los datos con los datos a tratar, cualquier cambio que se realice sobre ellos quedará reflejado automáticamente en cualquier lugar donde estos datos aparezcan.

- **Estabilidad**. Dado que permite un tratamiento diferenciado de aquellos objetos que permanecen constantes en el tiempo sobre aquellos que cambian con frecuencia permite aislar las partes del programa que permanecen inalterables en el tiempo.

- **Reusabilidad**. La noción de objeto permite que programas que traten las mismas estructuras de información reutilicen las definiciones de objetos empleadas en otros programas e incluso los procedimientos que los manipulan. De esta forma, el desarrollo de un programa puede llegar a ser una simple combinación de objetos ya definidos donde estos están relacionados de una manera particular.

Todo lo referente a POO tiene como premisa básica la reutilización y lograr una estabilidad para un proyecto, Ahí tiene su punto fuerte lenguajes como Java o C++, con una gran base OO (no lo es 100%) logra que el código sea fácil de leer, de mantener e inclusive modificar (ampliar) fundamentales a la hora de sistemas críticos/empresariales sujetos a cambios y estos tengan que ser soportados en un corto tiempo.

1.4.3. Lenguajes y plataformas de desarrollo.

1.4.3.1. Lenguajes.

Son muchos los lenguajes de programación que han aparecido a lo largo de la historia de las computadoras y a pesar de que con cada uno se pueden lograr hacer grandes aplicaciones, como todo, tienen sus ventajas y desventajas. Una de las desventajas de estos programas es que entre ellos existen diferentes maneras de estructurar el código, además de que cada programa maneja sus propias librerías y sintaxis, además de que las funciones en algunos casos se tienen que crear desde cero, lo que conlleva más tiempo a la hora de programar.
De acuerdo a la política de software libre que se está ejerciendo en nuestro país y en nuestra Universidad se ha decidido utilizar un lenguaje que cumpla con todas estas características, hoy en día el Ministerio de las Fuerzas Armadas (MINFAR) a llegado a la conclusión de utilizar el lenguaje llamado PHP con un gestor de Base de Datos PostgreSQL.

1.4.3.1.1. Lenguajes del lado del cliente

La tecnología llamada Cliente /Servidor es actualmente utilizada en casi todas las aplicaciones administrativas e Internet/Intranet. Bajo este esquema, un servidor es un ordenador remoto, en algún lugar de una red, que proporciona información según se le solicite. Mientras que un cliente funciona en su computadora local, se comunica con el servidor remoto y pide a éste información.

Típicamente aunque no necesariamente, un único servidor atiende a una multitud de clientes, ahorrando a cada uno de ellos el problema de tener la información instalada y almacenada localmente. Los sistemas Cliente-Servidor pueden ser de muchos tipos, pues esto depende principalmente de las aplicaciones instaladas en el propio servidor. Entre otros, existen: servidores de impresión mediante los cuales los usuarios comparten impresoras, servidores de archivos con los que los clientes comparten discos duros, servidores de bases de datos donde existe una única base de datos que es consultada por los clientes y puede o no ser modificada por ellos y servidores Web que utilizan también la tecnología Cliente/Servidor, aunque añaden aspectos nuevos y propios a la misma. . A continuación se dará una introducción a los diferentes lenguajes de programación para la Web, teniendo en cuenta la arquitectura cliente-servidor.

-HTML

Es un lenguaje estático para el desarrollo de sitios Web (acrónimo en inglés de HyperText Markup Language, en español Lenguaje de Marcas Hipertextuales). Desarrollado por el World Wide Web Consortium (W3C). Los archivos pueden tener las extensiones (htm, HTML). Es un lenguaje sencillo y de fácil aprendizaje, a la vez que es admitido por todos los navegadores. Vale la pena destacar que diseñar con este lenguaje se hace un poco lento, además de ser estático y que sus etiquetas son limitadas.

-Java script

Este es un lenguaje interpretado, no requiere compilación, nos permite interactuar con el navegador de manera dinámica y eficaz, La mayoría de los navegadores en sus últimas versiones interpretan código Java script. Es el lenguaje de programación del lado del cliente más utilizado, permite de forma eficiente validar formularios, detectar navegadores y mejorar el diseño, además de ser un lenguaje de fácil aprendizaje.

1.4.3.1.2. Lenguajes del lado del servidor

Ahora bien, como antes se menciona, los lenguajes del lado del cliente los cuales se usarán durante nuestro trabajo, ahora se mencionará y caracterizará el lenguaje de programación Web del lado del servidor a utilizar en nuestra aplicación.

PHP es un lenguaje de programación interpretado usado normalmente para la creación de páginas Web dinámicas. PHP es un acrónimo recursivo que significa "PHP Hypertext Pre-processor". Es software libre. Se puede obtener en la Web y su código esta disponible bajo la licencia GPL

- Sencillo de aprender.
- Similar en sintaxis a C y a PERL
- Soporta en cierta medida la orientación a objeto. Clases y herencia.
- El análisis léxico para recoger las variables que se pasan en la dirección lo hace PHP de forma automática. Librándose el usuario de tener que separar las variables y sus valores.
- Se puede incrustar código PHP con etiquetas HTML.
- Excelente soporte de acceso a base de datos.
- Se puede hacer de todo lo que se pueda transmitir por vía HTTP.
- Biblioteca nativa de funciones sumamente amplia e incluida
- No requiere definición de tipos de variables.
- Tiene manejo de excepciones.
- PHP no soporta directamente punteros, como el C, de forma que no existen los problemas de depuración provocados por estos.
- Viene acompañado por una excelente biblioteca de funciones que permite realizar cualquier labor (acceso a base de datos, encriptación, envió de correo, gestión de un e-commerce, XML, creación de PDF ...)
- Al poderse encapsular dentro de código HTML se puede recoger el trabajo del diseñador gráfico e incrustar el código php posteriormente.
- Es multiplataforma, funciona en todas las plataformas que soporten apache.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, PostgreSQL, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite; lo cual permite la creación de Aplicaciones Web robustas.

Fig. 1.13 Procesamiento de código PHP (tomada de [PHP 2005])

Ahora algunas desventajas que trae aparejadas PHP

- Todo el trabajo lo realiza el servidor y no delega al cliente. Por tanto puede ser más ineficiente a medida que las solicitudes aumenten de número.
- La legibilidad del código puede verse afectada al mezclar sentencias HTML y PHP.
- La orientación a objetos es aún deficiente para aplicaciones grandes.
- No posee una abstracción de base de datos estándar, sino bibliotecas especializadas para cada motor (a veces más de una para el mismo motor).

1.4.3.2. Plataformas de Desarrollo.

Una plataforma de desarrollo es el entorno de software común en el cual se desenvuelve la programación de un grupo definido de aplicaciones. Comúnmente se encuentra relacionada directamente a un sistema operativo; sin embargo, también es posible encontrarla ligada a una familia de lenguajes de programación o a una Interfaz de programación de aplicaciones.[ ]

1.4.3.2.1. Eclipse.

Eclipse es un Entorno de Desarrollo Integrado (IDE) que en un principio fue diseñado y desarrollado por IBM y que luego fue lanzada a la comunidad de software libre, y que se distribuye mediante una licencia de código abierto, la Eclipse Public License (EPL). En un primer momento Eclipse era solo un IDE para Java, pero mediante un sistema de plugins flexible, se puede nombrar que esta herramienta es totalmente neutral a la plataforma sobre la cual se ejecuta. Se utiliza eclipse, no solo para programar en java, C/C++ o Delphi, sino para integrar en ella lenguajes tan dispares como PHP, C#, Eiffel, Python o Cobol. [ ]

Eclipse no se debe ver solo como una herramienta de desarrollo, sino como una plataforma que permitirá ejecutar cualquier aplicación con cualquier lenguaje de soporte. Es la llamada, Arquitectura Plug – in.

Actualmente existen centenares de plugins para integrar en el eclipse, como editores de UML, herramientas de Optimización, Asistentes de cualquier tipo, pero no solo modificaciones de Eclipse como IDE de programación sino que pueden ser módulos de facturación, contabilidad, de análisis matemáticos, etc. Se puede decir que este nuevo entorno de desarrollo también incluye otras características que le harían lo suficientemente destacable sin poseer esta arquitectura tan especial. Eclipse permite agrupar código escrito y mostrado visualmente como una estructura empaquetada para que sea fácil poder seguir un código escrito. Otras características destacables son la instalación automática de nuevas funcionalidades o actualizaciones, el soporte con la herramienta de compilación e implantación de Ant de Jakarta o la implementación de unidades de testeo de JUnit. Eclipse no solo está haciendo sombra a otros entornos de programación ya consolidados dentro de la comunidad de desarrolladores, sino que está dejando entrever una luz tras años de oscuridad donde parecía que todos aquellos que escogieron java, esperaban de una herramienta de programación, potencia, fiabilidad, simplicidad e innovación.

En los últimos tiempos, la popularidad de Eclipse ha subido enormemente entre los desarrolladores no sólo de Java, sino también de los otros lenguajes a los que Eclipse da soporte, pues es una plataforma de herramientas universal y portable que proporciona un marco de trabajo o framework para desarrollar aplicaciones y herramientas.

El entorno de trabajo presenta un entorno de desarrollo integrado en perspectivas personalizables. Las perspectivas son combinaciones formadas por vistas y editores que muestran los diversos aspectos de los recursos del proyecto y están organizados por el rol o la tarea del desarrollador. Se proporcionan más de una perspectiva en un momento dado, además de pasar de una a otra mientras se esta trabajando. Las más grandes ventajas de Eclipse son:

Basado en plugins: Puedes conseguir cientos de plugins para programar en C++, Perl, PHP, XML, Java. Y no solo para programar sino también para modelar en UML.

Permite el desarrollo en equipo a través de CVS (Concurrent Version System).

IDE basado en archivos: Todo el contenido esta almacenado en archivos. Los recursos como clases java, archivos HTML, archivos XML (descriptores de despliegue), están almacenados en un sistema de archivo y así puedes obtener fácilmente acceso a ellos.

1.4.4. Framework.

Ext 2.0 basa toda su funcionalidad en JS a través de librerías ya conocidas: YUI, jQuery y Prototype/Script.aculo.us y un core interno poderoso. Así, en tiempo de ejecución carga y crea todos los objetos html a través del uso intenso de DOM.

Ventanas, mensajes emergentes, grillas, date pickers y un sin numero de utilidades son todas creadas en tiempo de ejecución. Los datos son obtenidos con mucho AJAX a través de XML y/o JSON.

.

Ventajas:

- La orientación a objetos intensa hará modular todos los scripts.
- El diseño está completamente separado de la funcionalidad.
- Funciones comunes como validación, combobox editables, ventanas arrastables.
- Buena y amplia documentación, así como también su comunidad.

Desventajas:

- Crear un sistema serio con esta herramienta requiere un previo uso prolongado,
- El tiempo de aprendizaje puede llegar a compararse con aprender a programar en un lenguaje nuevo.
- Al estar todo el sitio en JS, no podrá ser accesible para los buscadores, limitando su uso a sistemas y no sitios Web.

1.4.5. Gestor de Base de Datos.

1.4.5.1. PostgreSQL.

PostgreSQL es un servidor libre, liberado bajo la licencia BSD. Como muchos otros proyectos open source, el desarrollo de PostgreSQL no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo, dicha comunidad es denominada PostgreSQL Global Development Group (PGDG). PostgreSQL ofrece una potencia adicional sustancial al incorporar los siguientes cuatro conceptos adicionales básicos en una vía en la que los usuarios pueden extender fácilmente el sistema: clases, herencia, tipos, funciones.

PostgreSQL provee nativamente soporte para:

- Números de precisión arbitraria.
- Texto de largo ilimitado.
- Figuras geométricas (con una variedad de funciones asociadas)
- Direcciones IP (IPv4 e IPv6).
- Bloques de direcciones estilo CIDR.
- Direcciones MAC.
- Arreglos.

Otras características aportan potencia y flexibilidad adicional:

Restricciones (constraints).
Disparadores (triggers).
Reglas (rules).
Integridad transaccional.

Principales mejoras en el PostgreSQL:

- Se han implementado importantes características del motor de datos, incluyendo subconsultas, valores por defecto, restricciones a valores en los campos (constraints) y disparadores (triggers).
- Se han añadido funcionalidades en línea con el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, forzado de tipos cadena literal, conversión de tipos y entrada de enteros binarios y hexadecimales.
- Los tipos internos han sido mejorados, incluyendo nuevos tipos de fecha/hora de rango amplio y soporte para tipos geométricos adicionales.
- La velocidad del código del motor de datos ha sido incrementada aproximadamente en 40%, y su tiempo de arranque ha bajado el 80% desde que la versión 6.0 fue lanzada.
- Establecer condiciones o Cheks para validar las entradas de datos.
- Permitir transacciones, es decir, múltiples operaciones de tabla o registros de manera segura.
- Bloqueos de registros, útil en entornos hoy por hoy multiusuario.

Algunos de los lenguajes que se pueden usar con el PostgreSQL son los siguientes:

- Un lenguaje propio llamado PL/PgSQL (similar al PL/SQL de Oracle).
- C.
- C++.
- Gambas
- Java PL/Java Web.
- PL/Perl.
- plPHP.
- PL/Python.
- PL/Ruby.
- PL/sh.
- PL/Tcl.
- PL/Scheme.
- Lenguaje para aplicaciones estadísticas R a través PL/R.

1.5. Conclusiones.

Importante destacar que una vez culminado este capitulo se han obtenido resultados avalados principalmente por la selección de las tecnologías a emplear para el desarrollo de nuestra aplicación, ya que son las tecnologías, los lenguajes de programación los que permiten el desarrollo de los procesos. Resultados que se obtuvieron luego de un estudio detallado de cada una de estas tecnologías, así como del análisis de sus principales ventajas y potencialidades frente a otras que brindan similares prestaciones.

De aquí la decisión de desarrollar nuestra aplicación empleando como lenguaje de programación PHP, Java Script para la implementación del lado del cliente, así como AJAX, como gestor de bases de datos el PostgreSQL y el navegador Mozilla Firefox.

CAPITULO II: DESCRIPCION Y ANALISIS DE LA SOLUCION PROPUESTA

Introducción

En este capítulo se aborda el análisis del diseño propuesto por los analistas. Se analizarán las posibles implementaciones de módulos y componentes, así como la reutilización de los existentes. Se harán descripciones de las clases, los tipos de datos y las operaciones implementadas para dar solución al problema.

2.1. Valoración crítica del diseño propuesto en el análisis.

Del diseño propuesto por los analistas se pudo extraer de las clases y las funciones que deben tener las mismas, dando una idea clara de lo que se debe implementar, que explican la colaboración que existe entre las clases y cómo son llamados los funciones y sentencias dentro de cada una, de los cuales se obtuvo la información necesaria para conocer el orden de las acciones a implementar.

Se pudo comprobar después de un análisis que es necesario realizar cambios en el diseño de la base de datos ya que la misma no satisfacía completamente las necesidades del sistema, se agregaron nuevas tablas y campos a los ya existentes, debido a que en el proceso de implementación se hace necesario para facilitar el trabajo al recuperador o sea al usuario, además para conservar en un mayor grado la integridad del sistema ante la mirada del usuario que no siempre es confiable. En cuanto a la capa de presentación de la aplicación, se había decidido desarrollarla en DHTML, pero por las necesidades y la nueva forma en la que se están desarrollando las aplicaciones en el centro esta capa se desarrolló en Ext 2.0, por ser mas fácil su manejo al ya tener componentes implementados los cuales solo deberían ser adaptados a la aplicación en cuestión, no necesita estilo, ya lo trae predefinido, es mas agradable a la vista del usuario, etc.

El diseño propuesto fue creado siguiendo patrones, que de manera general constituyen soluciones simples y elegantes a problemas específicos y comunes del diseño orientado a objetos, permitiendo llevar a cabo la implementación clara y limpia del módulo bajo patrones como los GRASP y los GOF.

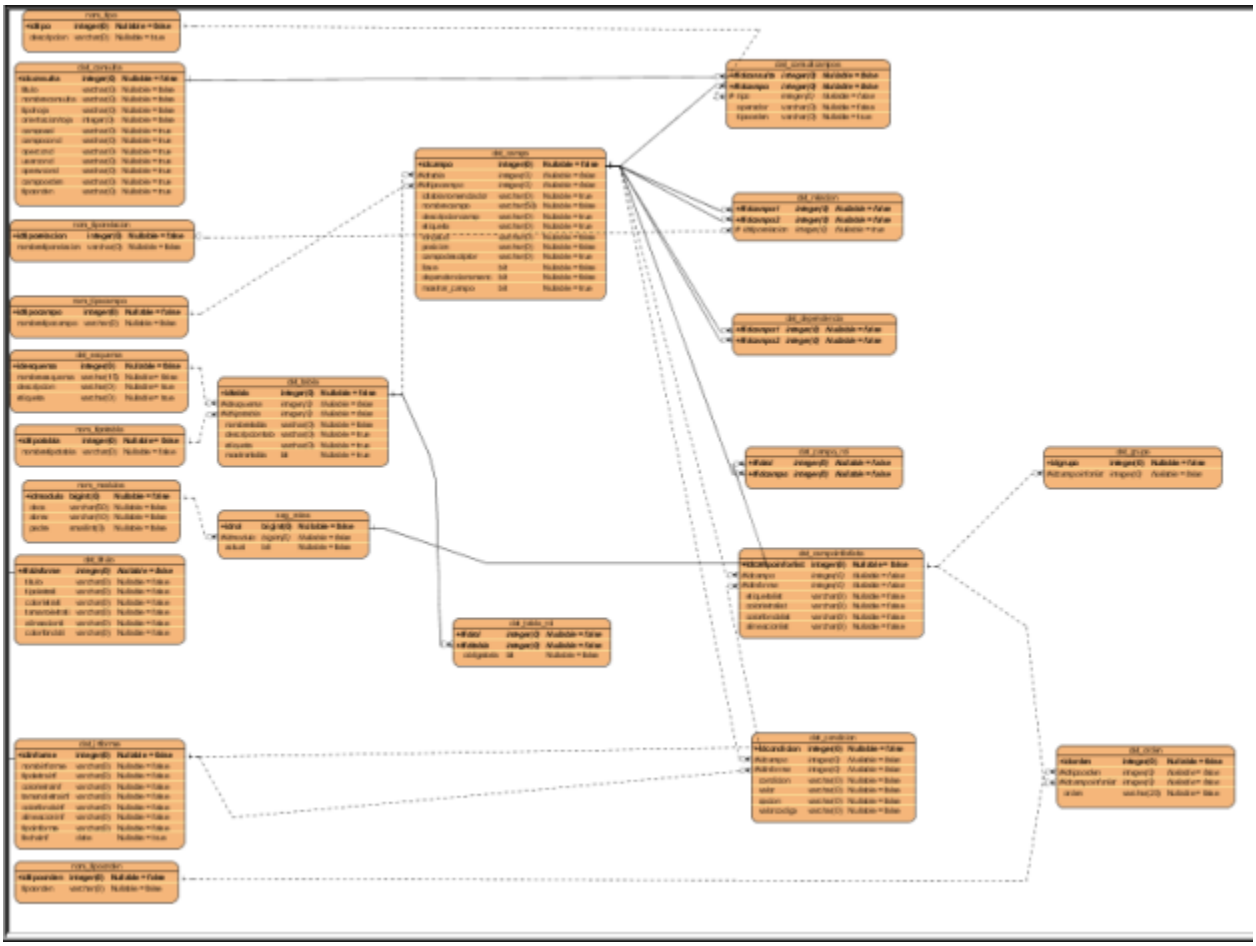
Los patrones GRASP se utilizan con el objetivo de asignar responsabilidades a las diferentes clases que se definen en el diseño. Dentro de este grupo se identifican cinco patrones muy utilizados: Experto, Creador, Alta Cohesión, Bajo Acoplamiento y el Controlador. Estos patrones se les aplican a las clases definidas en el diseño, distribuyendo responsabilidades entre las mismas de forma tal que no existan muchas relaciones, que no se sobrecargue de métodos a una clase en específico pudiendo acomodarlos en otras, entre otras mejoras que brinda el uso de este grupo de patrones.

Los patrones GOF generalmente se evidencian en clases que son creadas debido al uso de un patrón en específico. Existe un grupo de patrones de este tipo definidos para el diseño de clases, con el propósito de crear una arquitectura robusta para el sistema a desarrollar. Para el acceso a datos la capa de Lógica de Negocio se relaciona con una clase que implementa la interfaz del modelo de persistencia que responde a un patrón llamado Factory la cual es una puerta de enlace entre la capa de Acceso a Datos y la capa de Lógica de Negocio.

En la capa de acceso a datos, como es la capa encargada de establecer la conexión con la base de datos se implementa en ella el patrón de diseño Singlenton cuyo principal objetivo y para el cual está diseñado es restringir la creación de objetos pertenecientes a una clase, logrando que una clase sólo tenga una instancia.

2.1.1. Modelo físico de la Base de datos

Modelo físico de la Base de Datos después de hacer cambios para satisfacer las necesidades del sistema.



2.2. Análisis de posibles implementaciones, componentes o módulos ya existentes que puedan ser rehusados.

En este caso se pudiera argumentar que esta aplicación formará parte del sistema de gestión que esta desarrollando el MINFAR en nuestra universidad, por lo que muchos de los componentes de otros módulos ya desarrollados anteriormente nos sirven como base a nuestro proyecto, además se

cuenta con una bien elaborada arquitectura en tres capas, lo cual permite que todos los diferentes módulos sean de cierto modo semejantes.

Principalmente se uso el sistema de Generador de Típicas que de una forma u otra tiene el mismo principio que nuestra aplicación.

Se usan como componentes: Trees y grids componentes implementados en algunos otros módulos del centro por ejemplo Planificación, Recursos humanos.

2.3 Descripción detallada de la implementación.

El eje fundamental de nuestra aplicación esta en que es una aplicación flexible, cuyo principal objetivo es adaptar el recuperador dinámico a cualquier Base de datos, con el fin de cumplir este objetivo al estar la arquitectura del centro basada en PostgreSQL esta aplicación se basa en generar consultas a los metadatos o las tablas propias de PostgreSQL (pgClass, pgCatalogo, information schema, etc.), para obtener la estructura de la Base de Datos de la que se quiere recuperar, separando los datos y guardándolos en diferentes tablas (dat_esquema, dat_tabla, dat_campo ,dat_relaciones) las cuales tienen diferentes campos que le permiten al grafo de recuperación formar la consulta con la cual obtendrá la información de la Base de Datos de la que se quiere recuperar ,además para mayor seguridad y mayor entendimiento del usuario esta aplicación permite crear una mascara para la interacción con el usuario y así este no ve campos y tablas ,las cuales no les brindan ninguna información útil para el usuario .

2.3 Estrategias de integración.

Todo el código dentro un mismo componente se comunica mediante llamadas a métodos o funciones de forma directa. La información que es transmitida debe cumplir con los estándares que hay establecidos en el centro para adaptarse y facilitar la integración a nuevos componentes y otros sistemas desarrollados bajo las mismas concepciones. La base de datos es accedida mediante clases de acceso a datos como son las clases Consultas y Típicas (generadas por el Generador de Típicas, anteriormente mencionado), las cuales son a la ves instanciadas por las clases de lógica de negocio mediante la clase Factoría Típica .Esta secuencia es cumplida por todos los módulos del centro lo que por supuesto favorece a la integración, además de que todas

estas clases tienen un denominador común que y es que heredan de clases que son utilizadas por todos los módulos.

2.4. Descripción de los algoritmos no triviales a implementar.

Cuando se hace un análisis del código, se puede observar que no hay algoritmos extremadamente complicados, solo existen dos que destacan por encima de los demás:

Funciones a tener en cuenta por su complejidad:

Insertar campos:

Este paso se realiza posterior a la inserción de las tablas en dat_tabla, para ello se definió el siguiente algoritmo:

- Obtener las tablas almacenadas dentro de **dat_tabla**.

- Fijar una tabla y se obtiene los campos de la misma.

- Se verifica si el campo es llave primaria de la tabla.

- Se determina si el campo depende de un nomenclador.

-Se pone por defecto que el campo puede ser visible, dejando que el configurador lo defina según sea el caso.

-Llevar los resultados al formato requerido para su inserción dentro de **dat_campo**.

Insertar relaciones:

El último paso en la configuración del recuperador es la captura de las relaciones de la Base de Datos, para ello se definió el siguiente algoritmo:

- Obtener las tablas almacenadas dentro de **dat_tabla**.

- Obtener los ids asignados internamente por el gestor de BD de todas esas tablas.
- Se fija una tabla y se obtiene las tablas relacionadas con ella utilizado el id del paso anterior.
- Verificar que existan tablas relacionadas y la cantidad.
- Obtener el id de los campos relacionados según **dat_campo**.
- Determinar el tipo de relación (uno a muchos, etc.)
- Llevar los resultados al formato requerido para su inserción dentro de **dat_relacion**.

2.4.1. Selección de las estructuras de datos para la implementación. Descripción de los casos de uso desarrollados.

No se hizo necesario utilizar ninguna estructura de datos compleja, para el almacenamiento momentáneo y procesamiento de datos se utilizó arreglos, que se adaptaban perfectamente a las necesidades de implementación. Además de los arreglos se tienen a los árboles (Trees) que si el trabajo con ellos no es tan común como con los arreglos, se usan principalmente en la capa de presentación para mostrar datos de interés para el usuario, lo que permite una mejor comprensión y fácil manejo de la información que gestiona el sistema.

2.4.1.1. Descripción de Casos de uso.

Nombre: Configuración _ inicial

Atributo	Tipo
$host	
$nombreBD	
$usuario	
$contrasenna	

Para cada responsabilidad:	
Nombre:	construct
Descripción:	Constructor de la clase.
Nombre:	Getdatos
Descripción:	Funcion para
Nombre:	Control
Descripción:	Funcion para hacer las llamadas a las otras funciones según la opcion que se entre por la interfaz
Nombre:	GenFicheroConf($host,$nombreBD,$usuario,$contrasenna)
Descripción:	Funcion utilizada para actualizar los elementos de la tabla dat_esquemas
Nombre:	depNomenclador($campo)
Descripción:	Funcion utilizada para chequear si un campo tiene o no dependencia de un nomenclador
Nombre:	getTipocampo($tipo)
Descripción:	Funcion utilizada para comparar el tipo de campo que entra con uno de los tipo de campos que se encuentra en la tabla nom_tipocampo
Nombre:	
Nombre:	Modificar()
Descripción:	Funcion que se utiliza para insertar la etiqueta que se mostrara en el arbol y para seleccionar los campos y tablas

	que se mostraran en el arbol

Tabla 1 Clase Lógica de Negocio "*Configuración Inicial*"

Este Caso de uso (Configuración Inicial) es el eje central de la aplicación se pondrán a su disposición las diferentes consultas que se encargan de obtener los datos de los metadatos o las clases propias de PostgreSQL

Nombre: Consulta	
Atributo	Tipo
$sql	
$ndb	
$user	
$pass	
$ip	
$con	
Para cada función:	
Nombre:	Construct($ndb,$user,$pass,$ip)
Descripción:	Constructor de la clase.
Nombre:	ChequearConexion($ndb,$user,pass,$ip)
Descripción:	Función utilizada para conectarse y verificar si se ha podido conectar correctamente a la base de Datos
Nombre:	generarEsquema()
Descripción:	Funcion para generar el esquema en la base de datos

	de la que se quiere recuperar la estructura
Nombre:	queryAsArray($result)
Descripción:	Funcion utilizada para convertir lo que devuelve la consulta en un arreglo
Nombre:	getEsquemas()
Descripción:	Funcion que se encarga de obtener todos los esquemas de la base de datos para insertarlos en la tabla dat_esquema
Nombre:	getTablas()
Descripción:	Funcion encargada de obtener todos las tablas de la base de datos para insertarlos en la tabla dat_tabla
Nombre:	getCampos($tabla)
Descripción:	Funcion utilizada para obtener todos los campos de la Base de Datos para insertarlos en la tabla dat_campo
Nombre:	getCampoDescriptor($nom)
Descripción:	Funcion utilizada para obtener uno de los campos de la tabla dat_campo (campodescriptor)
Nombre:	getTipocampo($campo)
Descripción:	Funcion utilizada para obtener el tipo del campo que voy a insertar en la tabla dat_Campo
Nombre:	lastIdcampo()
Descripción:	Funcion para obtener el id con el que voy a insertar el nuevo campo a la tabla dat_campo
Nombre:	getNomencladores()
Descripción:	Funcion utilizada para obtener todos los nomencladores de la tabla dat_tabla para mas adelante poder llenar el campo dependencianomenc

	de la tabla dat_campo
Nombre:	getTablasArray()
Descripción:	Funcion utilizada para devolver las tablas de la tabla dat_tabla para obtener el id de la tabla y insertarlo en la tabla dat_campo
Nombre:	Insert($var)
Descripción:	Funcion utilizada para insertar todos los valores en la tabla dat_campo
Nombre:	InsertarRelaciones($var)
Descripción:	Funcion utilizada para insertar las relaciones que se recuperaron de la estructura de la Base de datos a recuperar informacion en la tabla dat_relacion
Nombre:	existeRelacion($id1,$id2)
Nombre:	Funcion utilizada para comprobar si la relacion ya existe en la tabla dat_relacion

Tabla 2 Clase consulta "*Consulta*"

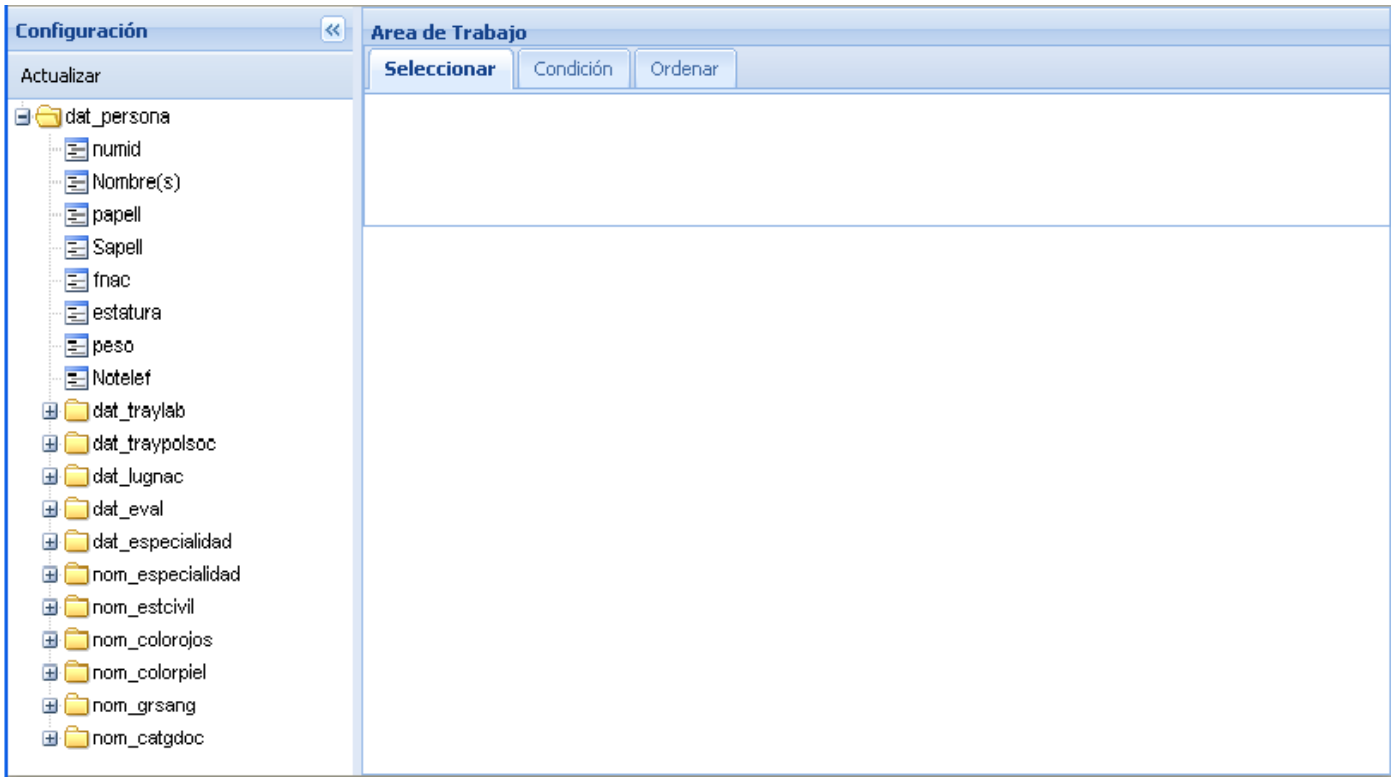
Nombre: lnListarDatos	
Atributo	
nombre	
id	
Para cada función:	
Nombre:	construct ($post)
Descripción:	Constructor de la clase.

Nombre:	Getdatos()
Descripción:	Funcion para
Nombre:	Control()
Descripción:	Funcion para hacer las llamadas a las otras funciones según la opcion que se entre por la interfaz
Nombre:	getArray()
Descripción:	Funcion utilizada para capturar la informacion que viene de la interfaz ,almacenarla en un arreglo y procesarla en la logica de negocio.
Nombre:	mensaje()
Descripción:	Funcion utilizada para comprobar si se encuentran los registros solicitados
Nombre:	genPDF($datos,$column,$titulo,$,$thoja,$orientacion)
Descripción:	Funcion que genera un documento pdf a partir de un array
Nombre:	getCampos()
Descripción:	Funcion utilizada para capturar los datos entrados por la interfaz para ser almacenados en la tabla dat_Consulta
Nombre:	insertar()
Descripción:	Funcion para insertar una consulta en la tabla dat_Consulta
Nombre:	getId($str)
Descripción:	Funcion que pasandole el arreglo que viene de la interfaz lo trata Como un arreglo con el esquema en la posicion 0 ,la tabla en la 1 y el campo en la 2
Nombre:	filtrarTipo($tipo, $campo)
Descripción:	Funcion utilizada para filtrar la reuperacion creando el
Nombre:	buscarNomSimples()

Descripción:	Funcion utilizada para obtener los nomencladores simples para llenar el combobox de las condiciones
Nombre:	crearArbol()
Descripción:	Funcion utilizada para llamar a las consultas y generar el arbol con todas las tablas y los campos
Nombre:	getNombreCampo()
Descripción:	Funcion llamada por la funcion getdatos() para componer la consulta

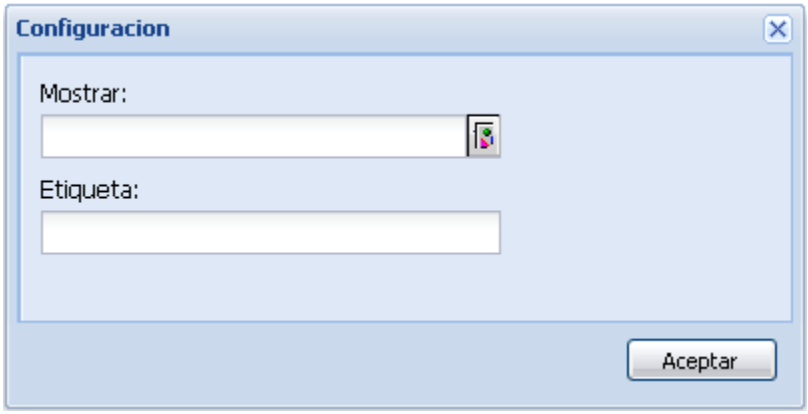
Tabla 3 Clase Lógica de negocio "*Listar personas*"

2.4.2. Interfaces de usuario



Interfaz 1 "*Configuración*"

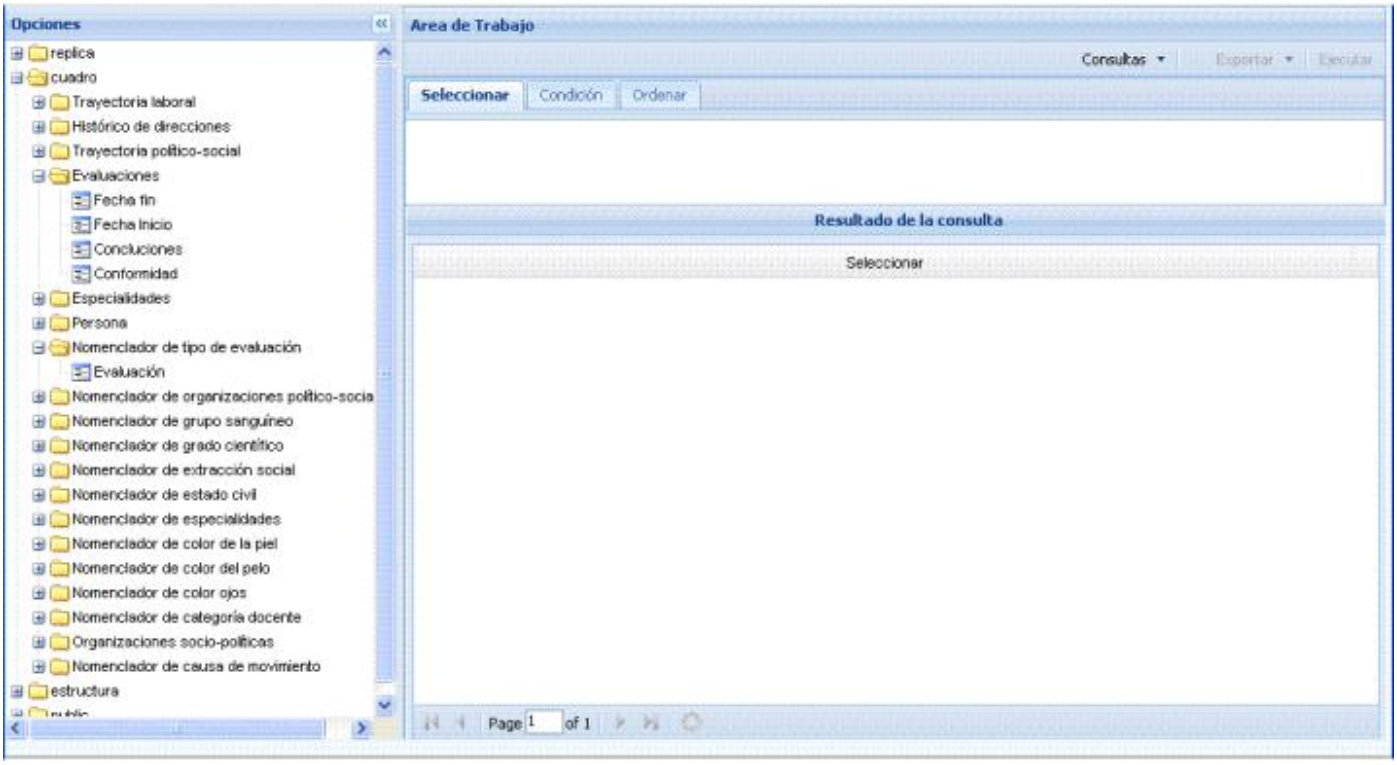
En esta interfaz se mostraran todas las tablas y campos de la base de datos para así poder configurar a lo que el usuario tiene acceso



Interfaz 2"*Modificar*"

En esta interfaz o ventana el usuario podrá especificar que campo y tabla mostrarle al usuario y que etiqueta tendrá el campo o la tabla.

Etiqueta: nombre con el cual el cual se les mostraran los campos y las tablas a los usuarios, esto es a decisión del configurador, sirve con propósitos de seguridad.



Interfaz 3"*ListarDatos*"

Interfaz que se le mostrara al usuario y en la que podrá trabajar después de haber terminado el trabajo el configurador de la aplicación.

2.5. Estándares de codificación.

Con el objetivo de lograr una aplicación con calidad y que reduzca la posibilidad de errores al mínimo, se establecen los estándares o reglas de codificación, además permite un mejor entendimiento del código escrito por los desarrolladores de las aplicaciones, la reutilización del código, el mantenimiento de forma ágil y con el mínimo de esfuerzo.

Para el desarrollo del sistema se emplean estándares para el código, para la estructura que debe tener cada sitio que se desarrolle, se crearon estándares para los nombres de las tablas de la base de datos, así como para los campos y atributos de las mismas.

Por ejemplo:

Los nombres de las clases empiezan de la forma siguiente:

Lógica de Negocio: ln

Consultas: c

Típicas: t

Los nombres de las tablas empezaran según el tipo de tabla:

Nomencladores: nom_

Informativas: dat_

2.6. Conclusiones

Como se ha podido observar este capitulo se ha centrado en la descripción de cómo trabaja esta aplicación, las diferentes clases que conforman la aplicación y sus funciones, para lograr cumplir su objetivo satisfactoriamente que no es mas, que recuperar la información que el cliente o el usuario necesite, de forma eficiente y optima.

CAPÍTULO III VALIDACION DE LA SOLUCION PROPUESTA.

3.1 Introducción

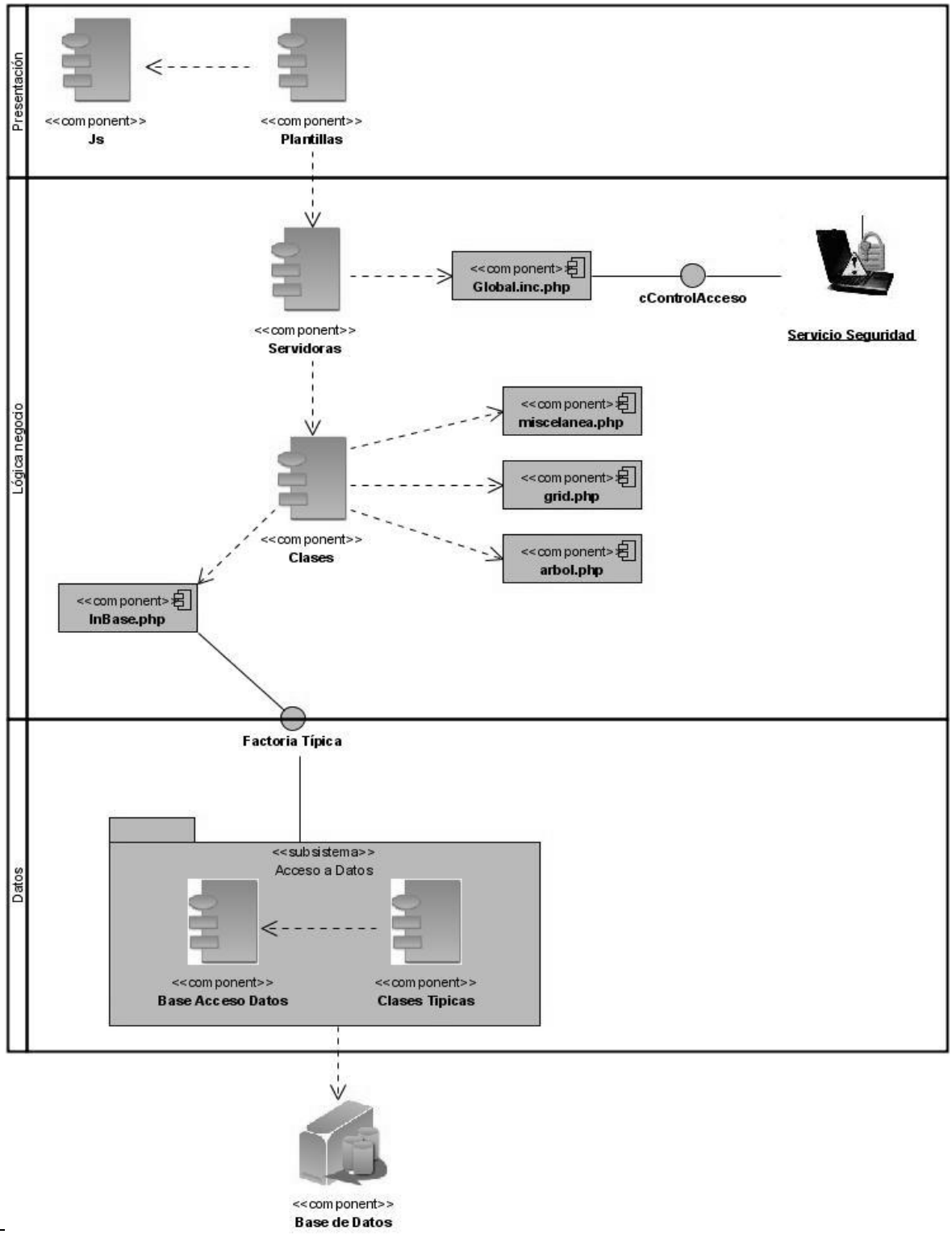
Las pruebas constituyen un flujo de trabajo de vital importancia para lograr entre otros aspectos la calidad e integridad de toda aplicación, el mayor grueso de las pruebas se llevan a cabo cuando se ha obtenido un resultado luego de la implementación. Por lo tanto en esta capitulo se tratara el tema de las pruebas de software y las respuestas del sistema ante las diferentes pruebas que se le realicen.

3.2. Diagrama de componentes

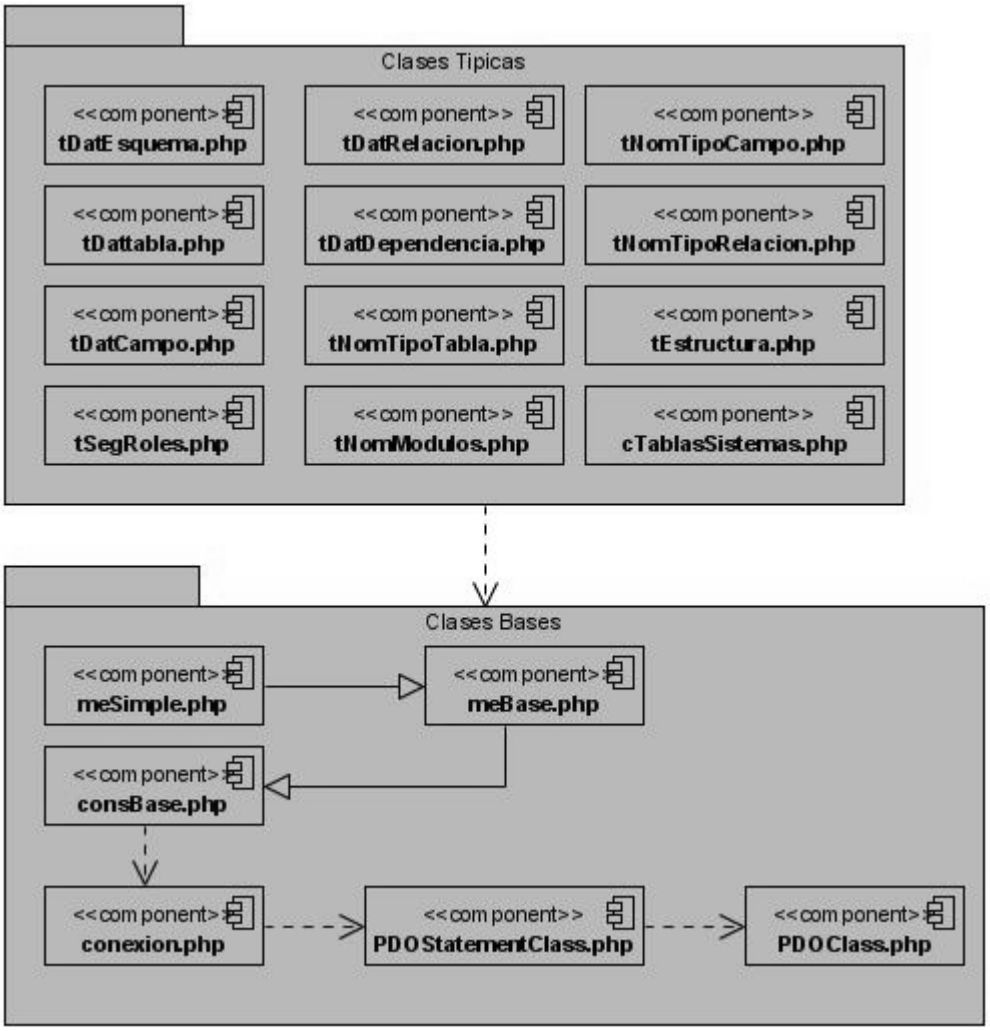
Los diagramas de componentes describen la organización y dependencias lógicas entre componentes *software*, siendo éstos, componentes de código fuente, ejecutables, librerías, tablas, entre otros.

En las siguientes figuras se muestra los diagramas de componentes por paquetes de nuestro módulo.

3.2.1. Diagrama de componentes (General):


Presentación
Lógica negocio
Datos
<<component>>
Js
<<component>>
Plantillas
<<component>>
Servidoras
<<component>>
Global.inc.php
cControlAcceso
Servicio Seguridad
<<component>>
miscelanea.php
<<component>>
grid.php
<<component>>
arbol.php
<<component>>
Clases
<<component>>
InBase.php
Factoria Típica
<<subsistema>>
Acceso a Datos
<<component>>
Base Acceso Datos
<<component>>
Clases Tipicas
<<component>>
Base de Datos

3.2.2. Diagrama de componentes (Acceso a Datos)



3.3. Diagrama de despliegue

En el siguiente diagrama de despliegue se representa la distribución física del sistema en términos de cómo se distribuirán la funcionalidades entre los nodos, cada nodo representa un recurso de cómputo, siendo estos procesadores o dispositivos hardware que se necesitarán para el despliegue del sistema.

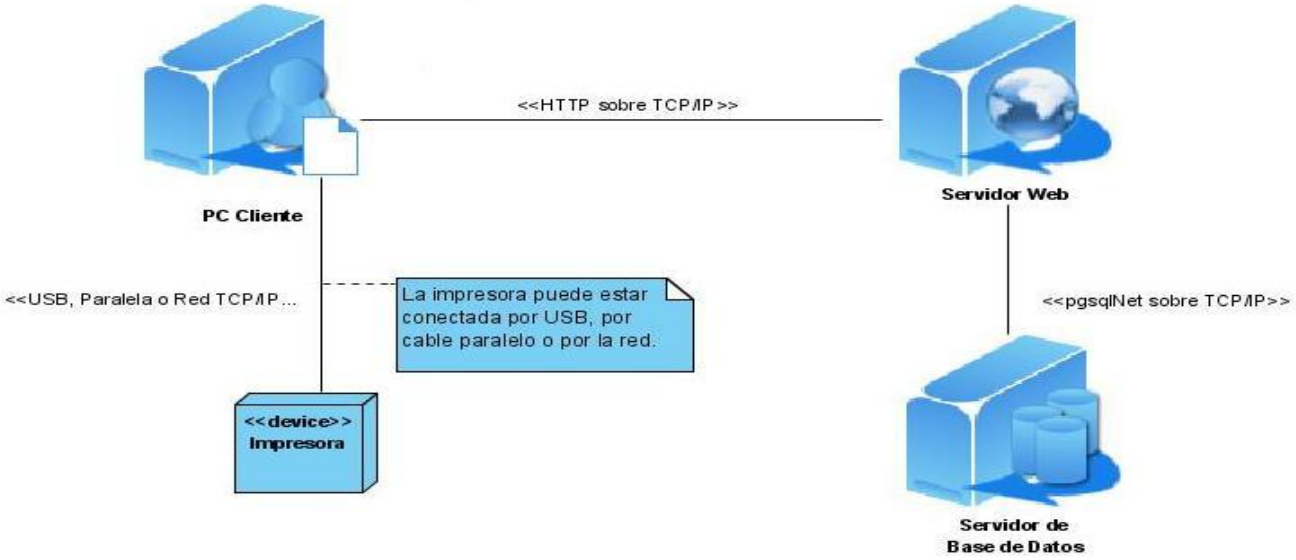


Diagrama de Despliegue.

3.4 Modelo de prueba

En esta sección se realizarán las pruebas de Caja Negra a aquellos casos de uso que tienen una alta probabilidad de introducir errores, para esto se definirán los casos de pruebas, las entradas y las condiciones, de aquí se obtendrán resultados que serán evaluados según los requisitos definidos y los objetivos del software.

CU Configuración Inicial.

Casos de Pruebas:

1. Introducir los datos para realizar la conexión.
2. Verificar si el esquema “recuperación” ya existe en la Base de Datos.

CU Configuración Inicial.		
Entrada	**Resultados**	**Condiciones**
	El sistema muestra una interfaz para entrar los datos de la Base de Datos a conectar.	
Fecha: “018/06/2008”	El sistema muestra un mensaje de alerta pidiendo que se llenen todos los campos correctamente	El sistema cheque el objeto conexión
Estado: “Elaboración”	El sistema muestra un mensaje de alerta mostrando que el esquema recuperación ya existe en la Base de Datos.	Existe en information_schema este esquema.

3.5 Conclusiones

En este capítulo se obtuvieron importantes resultados sobre el comportamiento del sistema a partir de la realización de varios procedimientos de pruebas, que como se ha dicho anteriormente es necesario para comprobar el buen funcionamiento del software, además Como se ha podido apreciar, en nuestro software el funcionamiento principal pasa fuera de la vista del usuario por lo que no es necesario validar muchas cosas en las interfaces de la aplicación en caso de ocurrir algún error seria por medio del configurador quien seria el único con acceso a este configurador

CONCLUSIONES

Este aplicación como se ha podido ver es bastante necesaria para facilitar el desarrollo y avance en los módulos del MINFAR y el ERP, por ser estos los que la utilizaran en menor periodo de tiempo. La aplicación como se ha querido expresar cumple su función para cualquier sistema que use PostgreSQL como gestor de base de datos, además tratar de generalizar en el desarrollo de los software es beneficioso porque estandariza el trabajo, por citar un ejemplo se tiene un software que fue desarrollado en el centro: el Generador de Típicas, que ahorra trabajo al programador al generar las típicas y las consultas; nuestro software se basa en el mismo principio, el ahorro de tiempo y esfuerzo de los programadores del centro, además que favorece a la integración y estandariza los reportes.

Para el desarrollo del sistema se realizó un estudio profundo de las tecnologías, herramientas y tendencias actuales empleadas para desarrollar aplicaciones Web, de aquí que se decidiera emplear para la implementación el lenguaje PHP, como gestor de base de datos PostgreSQL y el Apache como servidor Web. Además se emplearon otras herramientas de vital importancia para el desarrollo como lo son el PgAdmin y el PgManager para la administración de la base de datos, el Framework Ext2.0 para la capa de presentación del sitio.

RECOMENDACIONES

Se recomienda especialmente llevar este software de recuperación dinámica a un nivel más allá de lo que se había previsto:

Primeramente desarrollarlo como un software independiente como se tenía previsto, de forma tal que sea adaptable a cualquier módulo, no solo del MINFAR y del ERP, sino de otros proyectos en la UCI que usen como Gestor de Base de Datos PostgreSQL, este trabajo ratifica y coincide con la recomendación de la tesis que precedió a esta con el diseño de que se hace necesario y seria muy útil, un recuperador dinámico de información que sea configurable para cualquier gestor de Base de Datos, además se llama la atención acerca de la necesidad de investigar otras librerías para el acceso a datos como ADO DB, porque pudiera facilitar y brindar otras ventajas al desarrollo de este recuperador .

BIBLIOGRAFÍA

[1]- Recuperaciones Dinámicas .Universidad de Las Ciencias Informáticas .2006.Raidel Muñoz . Vidal y Michel López

[2]- Modulo de Configuración del Generador de Recuperaciones Dinámicas para aplicaciones Web.

Yanet Pérez Valcárcel y BelkisY.Torres González 2007.

[3] Torres, Ariel. *Que podemos hacer con PHP?,* 2007. Disponible en:
http://www.articuloweb.com/category.php?cat_id=3

[4] -de la Torre, Aníbal. *Lenguajes del lado del Servidor o Cliente,* 2006.Disponible en:
http://www.adelat.org/media/docum/nuke_publico/lenguajes_del_lado_servidor_o_cliente.html

[5] - Manual de PHP por:
Stig Sæther Bakken, Alexander Aulbach, Egon Schmid, Jim Winstead, Lars Torben Wilson, Rasmus Lerdorf,
Zeev Suraski, Andrei Zmievski, y Jouni Ahto

[6] - http://php.ciberaula.com/noticia/PHPpopularidad/
Manuales de Lenguajes de programación (PHP, Java), trabajo con CSS

[7]- http://www.desarrolloweb.com/manuales/
PHP manual Stig Sæther Bakken, por:
Alexander Aulbach, Egon Schmid, Jim Winstead, Lars Torben Wilson, Rasmus Lerdorf, Andrei Zmievsk, Jouni Ahto .Grupo de Documentación de PHP, 2004.

[8]- http://www.php.net/docs.php

[9]- Introducción al Lenguaje Estructurado de Consultas (SQL – Structured Query Language), por el Prof. Víctor Cherubini U. INACAP – Santiago Centro

[10]- Ejercicios Resueltos de SQL, por Borja Sotomayor 2002

[11]- Tutorial de PostgreSQL, por: El equipo de desarrollo de PostgreSQL y Editado por Thomas Lockhart

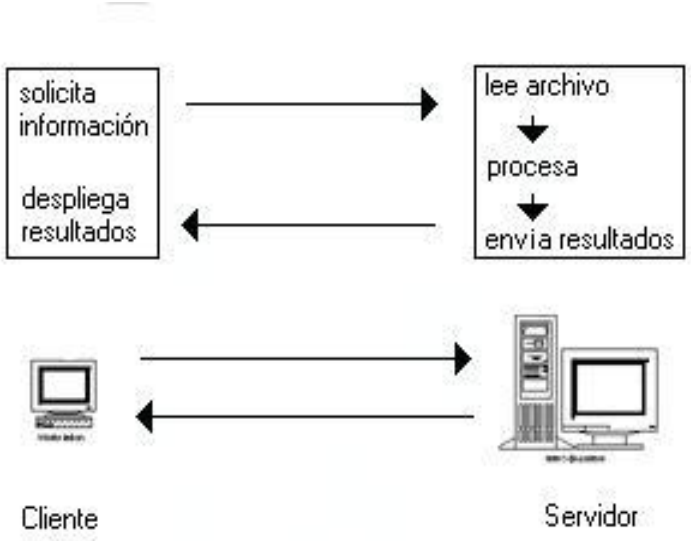
[12]- Practical PostgreSQL, por: John Worsley, Joshua Drake

[13]- pgAdmin Ayuda

[14]- Manual de Php, también un buen manual de PostgreSQL
http//www.php-es.com/ref.pgsql.html

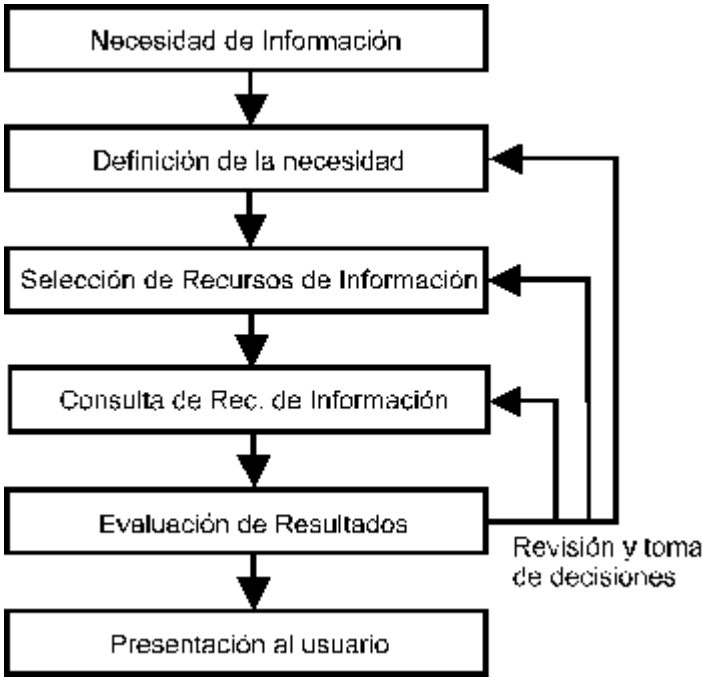
ANEXOS

Anexo 1 Arquitectura Cliente / Servidor.



Anexo 2. Arquitectura de *3 capas*.

Anexo 3. Proceso genérico de recuperación de información.


Necesidad de Información
Definición de la necesidad
Selección de Recursos de Información
Consulta de Rec. de Información
Evaluación de Resultados
Revisión y toma de decisiones
Presentación al usuario

GLOSARIO DE TERMINOS

Recuperación de información: es el conjunto de tareas mediante las cuales el usuario localiza y accede a los recursos de información que son pertinentes para la resolución del problema planteado. En estas tareas desempeñan un papel fundamental los lenguajes documentales, las técnicas de resumen, la descripción del objeto documental, etc.
En principio, la recuperación de información engloba las acciones encaminadas a identificar, seleccionar y acceder a los recursos de información útiles al usuario.

JSON: acrónimo de "Java Script Object Notation", es un formato ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de Java Script que no requiere el uso de XML. La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX. Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos en este contexto es que es mucho más sencillo escribir un analizador semántico de JSON