Universidad de las Ciencias Informáticas

Facultad 6

Exploración y diseño de la herramienta de administración de bases de datos para PostgreSQL (HABD)

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

AUTORA

Ivette Rosa Teodosio Acosta

TUTORAS

Ing. Marianela Gutiérrez Rodríguez

Ing. Aislein Blanco González

CO-TUTOR

Ing. Dairon Domínguez Vega

La Habana, Junio de 2011

Año 53 de la Revolución

"Si el presente es de lucha, el futuro es nuestro."

Ernesto Ché Guevara

DECLARACIÓN DE AUTORÍA

Declaro ser autora de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los _____ del año 2010.

Ivette Rosa Teodosio Acosta

_____

Firma de la autora

Ing. Aislein Blanco González

_____

Firma de la tutora

Ing. Marianela Gutiérrez Domínguez

_____

Firma de la tutora

Ing. Dairon Domínguez Vega

_____

Firma del co-tutor

DATOS DE CONTACTO

Tutora:

Ing. Marianela Gutiérrez Domínguez
Universidad de las Ciencias Informáticas, Habana, Cuba
e-mail: mgrodriguez@uci.cu

Tutora:

Ing. Aislein Blanco González
Universidad de las Ciencias Informáticas, Habana, Cuba
e-mail: ablanco@uci.cu

Co-Tutor

Ing. Dairon Domínguez Vega
Universidad de las Ciencias Informáticas, Habana, Cuba
e-mail: ddvega@uci.cu

No resulta fácil agradecer a todas las personas que han estado conmigo brindándome su apoyo y más que siempre he buscado ayuda en un gran número de personas, pero de esta manera u otra quiero agradecerles de todo corazón por haber estado conmigo en los momentos buenos y malos.

Especialmente quiero agradecer a dos personas, fueron los que me dieron la vida, la esperanza y la fuerza en todo momento para seguir adelante. Les doy las gracias por no haber dudado ni un segundo, a ustedes Joaquín de Jesús Teodosio Macías y Rosario del Carmen Acosta Gómez, mis padres y las personas que más quiero en este mundo. Resumir en pocas palabras lo mucho que les agradezco me resulta difícil.

También quiero agradecer a mi hermano, mis sobrinos y mi cuñada. Al resto de la familia decirle que también los quiero y le agradezco su preocupación y atención durante todo este tiempo de estudio. A mi novio Yoe Laurencio que con cariño y paciencia me ha brindado su apoyo para continuar este camino.

Quiero agradecer también a mis tutoras Marianela o Nela como cariñosamente le digo y Aislein por su ayuda, les agradezco por brindarme sus conocimientos, así

como a Dairon mi co-tutor. Mis disculpas por molestar tanto, gracias a su ayuda esta tesis salió adelante.

Muchas personas me ayudaron no solo durante el desarrollo del presente trabajo de diploma sino a lo largo de toda la carrera, lo menos que pudiera hacer para agradecerle es dejar plasmado en este documento sus nombres: Alicia, Richard, negro viste ya soy ingeniera, Yoriangel o Yuri que me gusta más, Liván y demás compañeros del apto 48203, a Franklin Cantillo que se graduó hace mucho tiempo pero que me ayudó mucho.

Agradezco a todos mis amigos que me ayudaron tanto en lo profesional como en lo espiritual, a los que están en estos momentos, a los que por un motivo u otro no están aquí a mi lado, a mis amigas de Sancti Spíritus Iris y Tay las quiero mucho y gracias por apoyarme aun en la distancia. Al grupo de amigos más unido que he podido experimentar Teresa la refeita que la quiero y que la extraño mucho pues ahora no está junto a mí aunque sé que no me olvida, a Yudelki gordis gracias también a tí, así como a Raiko, el Yixo y Orisbel alias el salvaje.

Por último y no menos importante quiero darle gracias a esta revolución y al ingenioso creador de lo que es hoy la UCI el compañero Fidel Castro Ruz.

Dedico este trabajo a mi mamá y a mi papá, esas 2 personas que han sido capaces de darlo todo sin recibir nada a cambio. De quererme con todo su corazón sin condiciones ni pretextos. Las personas que depositaron en mí todo su amor, confianza y anhelo. A ustedes que han dado todo lo que han podido por mi educación, para transitar por la vida como una persona de bien. A ustedes dedico este trabajo, los quiero con todo mi corazón.

Gracias por existir.

RESUMEN

Los usuarios que actualmente trabajan con el gestor PostgreSQL deben realizar mayor esfuerzo para explotar sus funcionalidades, pues las herramientas libres con las que interactúan poseen deficiencias en las interfaces visuales. Además carecen de una variedad de servicios que les pueden facilitar el trabajo a las personas que interactúan con el gestor. Incorporarles nuevas funcionalidades a estos clientes libres resulta complicado por la extensión y poca flexibilidad que poseen. Sin embargo las herramientas propietarias son muy potentes pero a la vez son muy costosas, y adquirir las licencias de software privativo se dificulta en Cuba, por pertenecer este a la lista de países embargados.

El presente trabajo está enmarcado en la realización de una herramienta de administración de bases de datos (HABD) de ahí su nombre, la cual provee una interfaz amigable y una arquitectura que permite a los desarrolladores incorporar nuevos servicios de manera sencilla por la flexibilidad que posee. De esta forma se le pueden agregar gran número de funcionalidades evitando así que los usuarios necesiten de varias aplicaciones para resolver un único problema. La aplicación contribuye al intercambio entre el usuario y el gestor PostgreSQL y facilita el trabajo de estos.

Palabras claves: Herramientas de administración de base de datos, PostgreSQL, Front-End, HABD.

Índice

Introducción

Las tecnologías en el mundo de hoy presentan un gran auge, es por ello que la informatización de la sociedad ha crecido a nivel mundial y esto trae consigo el aumento de la generación y almacenamiento de la información. Estos grandes volúmenes de datos no pueden ser procesados de forma manual por el hombre, pues afectaría la consistencia de la información que ellos poseen. Es por ello que deben existir métodos que faciliten realizar un rápido almacenamiento y una consulta ágil sobre los datos almacenados. Para recopilar todo tipo de información y atender las necesidades de un grupo de usuarios surge la tecnología de bases de datos, considerada una de las más antiguas dentro de la ciencia de la informática, utilizada actualmente para manejar gran cantidad de datos.

Como resultado de la evolución de las bases de datos hoy en día estas son mucho más seguras, flexibles y manejables en la mayor parte de los casos. Esto es posible desde la creación de los Sistemas Gestores de Bases de Datos (SGBD); software que permite manejar los datos almacenados que posteriormente se convertirán en información. La utilización de un SGBD con aplicaciones dedicadas a convertir los datos almacenados en información de interés para los usuarios, facilita la toma de decisiones en las empresas y organismos. También son ampliamente utilizados en entornos científicos con el objetivo de almacenar la información experimental.

Las empresas cubanas al igual que cualquier organización del mundo necesitan de SGBD que le permitan la gestión de la información que poseen y además que les facilite anticiparse a los cambios futuros y adaptarse rápidamente a ellos. “La Industria Cubana del Software por su parte tiene como meta lograr que las empresas cubanas empleen en sus soluciones, tecnologías de bases de datos libres, de manera que se avance en el proceso de adquisición de la soberanía tecnológica. Un paso decisivo en aras de alcanzar este objetivo es contar con un sistema gestor de bases de datos propio, potente, flexible y que pueda ser personalizado según las necesidades de la empresa que lo utilice”. [1] Entre las instituciones que se dedican al desarrollo de soluciones informáticas en Cuba se encuentran: Transoft, Desoft, Infoservivi, Centro de desarrollo de software de Villa Clara-Universidad de las Ciencias Informáticas y la Universidad de las Ciencias Informáticas (UCI).

La UCI por su parte tiene entre sus misiones agilizar e impulsar la informatización cubana para incorporarla en los diferentes sectores sociales. Es una Universidad docente-productiva, de manera que sea posible producir software y servicios informáticos a partir de la vinculación entre el estudio y el trabajo como modelo de formación. Posee una infraestructura productiva conformada por varios centros uno de ellos es el Centro de Tecnologías de Gestión de Datos (DATEC), el cual está compuesto por varias líneas de producción dentro de las que se encuentra PostgreSQL.

La línea PostgreSQL tiene la tarea principal de llevar a cabo la idea de confeccionar un PostgreSQL Empresarial Cubano. “La solución se nutrirá del núcleo del PostgreSQL 8.4.1; además de un conjunto de herramientas en torno al gestor que facilitarán su utilización y de un grupo de módulos, que serán incorporados en el paquete Contrib del código del gestor de base de datos y que darán valor agregado al mismo”. [1]

Los SGBD poseen herramientas gráficas de diseño y administración que le proporcionan al usuario una vía más intuitiva para interactuar con ellos. Los usuarios que actualmente trabajan con el gestor PostgreSQL deben realizar mayor esfuerzo para obtener los beneficios que este provee y esto se debe a las deficiencias que presentan las herramientas de administración con las que interactúa.

Las herramientas de administración pueden ser propietarias y libres, siendo las propietarias las de mayor potencialidad por la variedad de funciones que poseen. En Cuba se dificulta la adquisición de las licencias de software privativo y esto se debe al bloqueo impuesto por los Estados Unidos de América (EUA), el cual impide el acceso a productos de software propietarios, así como el comercio con otros países que poseen empresas que tengan vínculos con él.

La mayoría de estos clientes de administración de bases de datos prestan servicios que son comunes entre ellos, pero existen algunos que no incluyen funcionalidades dedicadas a tareas específicas. Los usuarios necesitados de estas soluciones, muchas veces deben conseguir varias herramientas que puedan resolver un único problema, por lo que se produce cierto tiempo extra de adaptación al nuevo software. Además la necesidad de capital para la compra en caso de no ser gratis y otros aspectos relacionados con el costo de actualización del hardware y software.

Unido a este problema, se encuentra la necesidad de integrar nuevas funcionalidades en las que actualmente se encuentra trabajando el grupo de desarrolladores de la línea PostgreSQL tales como el CRUD-PG, Mask-PG, PgDataMerge y otra para el particionado de tablas del gestor PostgreSQL. Se dificulta la integración de dichas soluciones debido a que las herramientas gráficas de administración libres que existen presentan sus funcionalidades de una forma integrada, es decir en un único componente y resulta complicado en cuanto a tiempo y personal calificado la integración de estas nuevas aplicaciones que son desarrolladas.

Por la situación planteada con anterioridad surge la siguiente **interrogante**: ¿Cómo facilitar la interacción de los usuarios con el gestor de base de datos PostgreSQL?

Con el desarrollo de este trabajo se pretende dar solución al problema antes citado. Para la obtención de estos resultados se plantea como **objeto de estudio**: El proceso de desarrollo de las herramientas de administración de gestores de base de datos, enmarcado en el **campo de acción:** El proceso de exploración y diseño del Front-End y del plug-in “Explorador de Objetos de Base de Datos”.

El **objetivo general** del trabajo: Realizar el proceso de Exploración y Diseño el Front-End y el plug-in “Explorador de Objetos de Base de Datos” de la herramienta HABD, para lograr una mejor interacción entre los usuarios y el gestor PostgreSQL.

En correspondencia con el mismo se trazan los siguientes **objetivos específicos:**

- ✓ Realizar el proceso correspondiente a la fase de exploración del Front-End y del plug-in “Explorador de Objetos de Base de Datos” de la herramienta HABD.
- ✓ Diseñar el Front-End y el plug-in “Explorador de Objetos de Base de Datos” de la herramienta HABD.
- ✓ Realizar pruebas de validación de software al Front-End y al plug-in “Explorador de objetos de bases de datos” de la herramienta HABD.

Para dar cumplimiento a los objetivos específicos se plantean las siguientes **tareas de la investigación:**

- ✓ Revisión bibliográfica de las herramientas de administración de bases de datos de distintos gestores para la definición de la nueva herramienta.

- ✓ Definición de las historias de usuarios del Front-End y del plug-in “Explorador de Objetos de Base de Datos” para la herramienta HABD, facilitando el diseño de la misma.
- ✓ Definición de los requisitos funcionales y no funcionales del Front-End y del plug-in “Explorador de Objetos de Base de Datos” para la herramienta HABD, facilitando el diseño de la misma.
- ✓ Definición de la arquitectura de software del Front-End y del plug-in “Explorador de Objetos de Base de Datos” para la herramienta HABD.
- ✓ Confección de los artefactos del expediente de proyecto hasta la fase de desarrollo del Front-End y del plug-in “Explorador de Objetos de Base de Datos” para la herramienta HABD.
- ✓ Realización de pruebas de caja negra al Front-End y al plug-in “Explorador de objetos de bases de datos”.

El presente trabajo está compuesto de 3 capítulos que contiene la información relacionada con la herramienta de administración de bases de datos HABD, que utilizará el gestor PostgreSQL y que será parte del conjunto de herramientas que van a conformar el gestor PostgreSQL Empresarial Cubano. A continuación se muestra una descripción breve de cada capítulo.

Capítulo 1: Fundamento Teórico. En el capítulo se describe el marco teórico del trabajo, realizando un profundo análisis del estado del arte de las herramientas de administración de varios SGBD y sus características visuales. Contiene un análisis de los distintos conceptos que permiten un entendimiento claro y preciso de la presente investigación. Se brinda además una breve panorámica de las características fundamentales de este tipo de herramientas y se realiza un estudio de las tecnologías en las que se apoyará la realización de este software en función de un estudio de las tendencias actuales, encaminadas fundamentalmente hacia el software libre.

Capítulo 2: Características del sistema propuesto. El capítulo está integrado por las fases: Exploración, donde se describe la herramienta que se propone a través de las historias de usuarios, obteniéndose la lista de reserva del producto. Esta constituye una lista que recoge los requisitos funcionales y no funcionales del software. Para el negocio se establece un modelo de dominio el cual representa los conceptos más significativos que permiten entender con claridad el problema. Además la fase de Planificación donde se exponen las tareas de la ingeniería y el plan de iteraciones. También está

constituido por la fase Diseño que recoge las tarjetas CRC (clase, responsabilidad, colaboración) así como la arquitectura de software.

Capítulo 3: Validación del sistema propuesto. En el capítulo se describe el método que se utiliza para diseñar los casos de pruebas que guiarán la validación de la aplicación. El método que se utiliza es caja negra, la técnica es partición de equivalencia que rige en el proyecto PostgreSQL del centro DATEC. Además se muestran las no conformidades encontradas de la aplicación.

Capítulo 1: Fundamento Teórico

Introducción

En el capítulo se describe el marco teórico del trabajo, realizando un profundo análisis del estado del arte de las herramientas de administración de varios SGBD y sus características visuales. Contiene un análisis de los distintos conceptos que permiten un entendimiento claro y preciso de la presente investigación. Se brinda además una breve panorámica de las características fundamentales de este tipo de herramientas y se realiza un estudio de las tecnologías en las que se apoyará la realización de este software en función de un estudio de las tendencias actuales, encaminadas fundamentalmente hacia el software libre.

1.1 Conceptos asociados a la investigación

Bases de datos

Muchas personas han escrito acerca de las bases de datos, entre ellas se puede citar a Sergio Ezequiel Rozic cuando plantea: “Una base de datos es el lugar donde se guardan los datos en reposo y al cual acceden las diferentes aplicaciones (sistemas o programas) de una organización dada". [2]

EL Dr. Ramez Elmasri expresa que: “Una base de datos tiene una fuente de la cual se derivan los datos, cierto grado de interacción con los acontecimientos del mundo real y un público que está activamente interesado en el contenido de la base de datos”. [3]

“Es una colección de archivos interrelacionados, son creados con un DBMS (DataBase Manage System). El contenido de una base de datos engloba a la información concerniente (almacenadas en archivos) de una organización, de tal manera que los datos estén disponibles para los usuarios, una finalidad de la base de datos es eliminar la redundancia o al menos minimizarla “. [4]

Se puede definir como una base de datos una serie de datos organizados y relacionados entre sí, con la menor redundancia posible. Permite a los usuarios mantener los datos organizados facilitando la persistencia de los mismos durante mucho tiempo. Se convierten más útiles a medida que la cantidad de datos almacenados crece. Los datos recolectados son explotados por los sistemas de información de una

empresa o negocio, transformándolos en información de gran valor que le puede proporcionar ventaja competitiva en el mercado.

Sistema Gestor de Bases de Datos

Con el desmedido aumento de información y de personal que accede sobre ella, rápidamente surgió la necesidad de poseer un sistema de administración para controlar tanto los datos como los usuarios. Para la administración de bases de datos se utilizan los SGBD.

Adoración de Miguel lo define como “Conjunto coordinado de programas, procedimientos, lenguajes que suministra, tanto a los usuarios no informáticos como a los analistas, programadores o al administrador, los medios necesarios para describir, recuperar y manipular los datos almacenados en la base, manteniendo su integridad, confidencialidad y seguridad”. [5]

“Se trata de un conjunto de programas no visibles al usuario final que se encargan de la privacidad, la integridad, la seguridad de los datos y la interacción con el sistema operativo. Proporciona una interfaz entre los datos, los programas que los manejan y los usuarios finales. Cualquier operación que el usuario hace contra la base de datos está controlada por el gestor”. [6]

Con el apoyo de los conceptos antes expuestos se define un SGBD como un conjunto de aplicaciones de software para administrar bases de datos y que va a actuar como intermediario entre los usuarios, la base de datos y las aplicaciones que la utilizan. Permite acceder de manera organizada a los datos almacenados para que el personal calificado en determinado tema tenga en orden y con la menor duplicidad posible la información que necesita. El trabajo con los SGBD no resulta fácil y se puede convertir en un problema si no existen personas con el conocimiento requerido para administrarlos.

Interfaz de usuario

La comunicación entre el usuario y el ordenador comúnmente se realiza a través de una interfaz de línea de comandos o de una interfaz gráfica de usuario (GUI, siglas en inglés). Las interfaces de línea de comandos requieren que el usuario introduzca instrucciones breves mediante un teclado para dirigir las acciones de la computadora. Las GUI emplean ventanas para organizar archivos y aplicaciones

representadas por iconos (pequeñas imágenes) y menús que presentan una lista de instrucciones. El usuario manipula directamente estos objetos visuales en el monitor señalándolos, seleccionándolos y arrastrándolos o moviéndolos con el mouse.

“La interfaz de usuario (IU) es uno de los componentes más importantes de cualquier sistema computacional, pues funciona como el vínculo entre el humano y la máquina…el éxito de un programa frecuentemente se debe a qué tan rápido puede aprender el usuario a emplear el software, de igual importancia es que el usuario alcance sus objetivos con el programa de la manera más sencilla posible”. [7]

Los usuarios de aplicaciones computacionales disponen actualmente de una abundancia de información que no siempre resulta útil, y encontrar exactamente lo que se necesita puede ser difícil. Es por eso que todo producto de software debería aprovechar las capacidades que posee sin llegar a sobrecargar las interfaces de usuarios. Esto posibilita que el usuario se sienta satisfecho con el software pues además de proveerle las funcionalidades que necesita puede adaptarse al sistema con mayor facilidad.

1.2 Sistemas Gestores de Bases de Datos.

Los SGBD proporcionan a los usuarios la gestión del acceso a la base de datos por múltiples usuarios de manera simultánea y garantizan que no sucedan inconvenientes con la integridad de los mismos. Aunque poseen muchas ventajas es importante aclarar que el trabajo con SGBD no es una tarea fácil, pues contienen programas muy complejos que proveen grandes soluciones, y requieren de gran conocimiento en el tema de administración de bases de datos para poder obtener el mayor provecho de ellos. Para ayuda de los administradores de base de datos los gestores proveen interfaces y lenguajes de consulta que simplifican el trabajo con los datos.

En la actualidad se puede encontrar SGBD muy potentes que proporcionan muchas de las funciones estándar que el programador necesita para lograr grandes avances, por ejemplo: Oracle, DB2, PostgreSQL, SQL-Server, Fox Pro, Access y SQLite. A continuación se muestran algunas características de PostgreSQL ya que es el núcleo del gestor cubano, y al cual se le agregarán un conjunto de herramientas que facilitarán el trabajo de los usuarios que utilicen este gestor.

1.2.1 PostgreSQL

PostgreSQL es el gestor de base de datos relacional de código abierto más avanzado del mundo. Distribuido bajo licencia BSD (del inglés, Berkeley Software Distribution), lleva más de 15 años desarrollándose y su arquitectura despierta el interés de muchos por su fiabilidad e integridad de los datos. Es el gestor escogido como base para adoptarlo a las necesidades de las empresas cubanas y conformar el gestor PostgreSQL Empresarial Cubano.

PostgreSQL además de ser altamente adaptable a las necesidades del cliente, corre en casi todos los principales sistemas operativos: Linux, Unix, BSDs, Mac OS, Beos, Windows (34 plataformas). La documentación que posee se encuentra muy bien organizada, es pública y libre, con comentarios de los propios usuarios que pueden intercambiar problemas y soluciones, de esta manera existe un progresivo avance en el uso de este gestor. La comunidad que desarrolla los distintos servicios que provee el SGBD brinda soporte nativo para los lenguajes del medio como: PHP, C, C++, Perl y Python.

1.2.2 PostgreSQL Empresarial Cubano

El gestor PostgreSQL Empresarial Cubano está basado en PostgreSQL pero dirigido a solucionar las necesidades de las aplicaciones empresariales y es implementado para organismos y empresas cubanas. La figura 1 muestra la relación de los componentes que conforman el PostgreSQL Empresarial Cubano; “PostgreSQL como núcleo y en torno a él un conjunto de herramientas, materiales para el entrenamiento y soporte que conformen, como un todo, un producto a la altura de las necesidades de las empresas y organismos cubanos

Fig.1: Componentes que conforman el sistema de gestión de bases de datos cubano

Para el desarrollo del gestor, se definieron 3 fases, enfocadas a los componentes mostrados en la figura 1.

Fase 1: Desarrollo de materiales para el entrenamiento y soporte, donde se elaborarán materiales e instrumentos para simplificar el trabajo con el gestor.
Fase 2: Desarrollo de herramientas, donde se desarrollarán aplicaciones para explotar al máximo las potencialidades del gestor.
Fase 3: Desarrollo del núcleo, donde se añadirán funcionalidades al núcleo para incrementar la potencia y rendimiento del gestor." [8]

Entre las herramientas que se hacen necesarias y que conformarán el conjunto de aplicaciones que darán valor agregado al gestor PostgreSQL Empresarial Cubano se encuentra la herramienta HABD la cual brindará una interfaz entre el usuario y el gestor, proporcionando un mejor aprovechamiento de este por parte de los usuarios finales.

1.3 Herramientas de Administración de Bases de Datos. Interfaz Visual

Con el surgimiento de las Herramientas de Administración de Bases de Datos se hace más simple el proceso de trabajo al administrador de las bases de datos, esto se debe a las interfaces gráficas que proveen estas herramientas. La cantidad de errores que surgían en la interacción diaria con el gestor disminuyó considerablemente, pues se debían a que los usuarios tenían que memorizar muchas líneas de comandos, lo que traía consigo muchos conflictos a los administradores de bases de datos.

Para el presente trabajo de diploma se escogen 3 herramientas propietarias y 2 libres para estudiar sus características y funcionalidades. Las primeras son producidas por las compañías Microsoft y Oracle, ambas norteamericanas y que se encuentran en los primero lugares de la lista de países desarrolladores de software más importantes. Las libres son realizadas por el Grupo de Desarrollo Global de PostgreSQL (PGDG, por sus siglas en inglés) y constituyen las de más aceptación por los usuarios. En el Anexo 1 se puede ver una tabla que reúne aspectos muy importantes de las herramientas y que se van a analizar para la decisión de la concepción de la nueva herramienta.

1.3.1 EMS SQL MANAGER para PostgreSQL

“EMS SQL Manager para PostgreSQL está integrado por un conjunto de funcionalidades muy potentes entre las que se pueden encontrar el diseñador visual de base de datos, constructor visual o editor profesional de texto de SQL con la capacidad de crear y ejecutar consultas avanzadas, así como creación de bases de datos instantáneas en forma de SQL script. También da la posibilidad de ver, buscar y editar el contenido de la base de datos visualmente para que se pueda recuperar los datos más relevantes. Además tiene una impresionante exportación de los datos y capacidades de importación. Puede exportar datos a 17 formatos de archivo e importarlos.” [9]

Provee una herramienta muy potente para la creación de consultas complejas visualmente sin un profundo conocimiento de la sintaxis SQL que se nombra Visual Query Builder. Posee una interfaz intuitiva y fácil de manejar con un sistema de ayuda bien descrito, de modo claro en su uso, incluso un inexperto no debe complicarse con ella. Las herramientas le permiten crear y editar el texto SQL de una consulta, prepararlas y ejecutarlas así como ver los resultados de ejecución de la misma.

La interfaz visual de esta herramienta brinda muchas facilidades a los usuarios que interactúan con la misma. Ejemplo de estas son las barras de herramientas capaces de ser personalizadas para todas las ventanas y según las necesidades del usuario. También el explorador de base de datos que facilita la gestión de todos los objetos PostgreSQL de una manera más cómoda y amigable.

1.3.2 Dream Coder for Oracle DBA

Dream Coder for Oracle es una herramienta muy poderosa y potente que contiene diferentes utilidades y módulos para administrar y desarrollar en un servidor de base de datos Oracle por lo que no es necesario comprar varias aplicaciones independientes que dificultan y hagan más complejo el trabajo diario.

“Dream Coder for Oracle DBA ofrece una interfaz intuitiva que permite realizar rápida y fácilmente tareas con la base de datos. Cuenta características que le permiten realizar rápida y fácilmente todas las actividades de desarrollo y administración, gracias a su intuitiva interfaz, la cual incluye opciones para construir y depurar código SQL y PL/SQL. Facilitan el proceso de crear, editar, duplicar, exportar y borrar objetos, compilar y ejecutar procedimientos almacenados, exportar e importar datos, generar reportes,

monitorear la actividad de la base de datos, sincronizar la base de datos, construir y ejecutar consultas, formatear el código SQL y PL/SQL. “[10]

1.3.3 EMS SQL Management Studio for SQL Server.

“EMS SQL Management Studio es una solución integral para la administración y desarrollo de bases de datos. Es una plataforma de trabajo que provee herramientas esenciales para administrar bases de datos y gestionar sus objetos tales como la migración de bases de datos, extracción, importación, exportación y comparación de datos. Es un cliente rico, diseñado para satisfacer las necesidades del administrador de SQL Server. En esta herramienta las tareas administrativas se llevan a cabo utilizando el Explorador de objetos, que le permite conectarse a cualquier servidor de la familia SQL Server y gráficamente ver su contenido.” [11]

La interfaz visual de esta herramienta proporciona funcionalidades que les brindan una vía más intuitiva y fácil a los usuarios para intercambiar con el gestor, un ejemplo es la barra de herramientas para cambiar entre las ventanas con más facilidad, como en la barra de tareas de Windows. Además de poseer la ventaja de personalizar la interfaz y contar con un potente módulo de opciones visuales.

Luego de un estudio realizado a las anteriores aplicaciones descritas se llega a la conclusión que EMS PostgreSQL Manager, DreamCoder for Oracle DBA, EMS SQL Management Studio son HABD muy potentes. Aun así no es factible trabajar con ellas porque no se corresponde con las necesidades económicas de Cuba pues el precio de adquisición es muy elevado. Además el soporte y mantenimiento, elemento clave que le brinda seguridad e integridad a todo producto de software, es muy costoso. Otro factor que influye es que van contra la soberanía tecnológica que se está implementando en el país porque son propietarias, problema que impide consultar su código fuente y realizarle transformaciones. Además que no están disponibles para ninguna distribución de Linux por lo que para trabajar con ellas es necesario emularlas o utilizar máquinas virtuales con sistemas operativos windows.

Las herramientas antes descritas son herramientas propietarias pero a continuación se muestran dos clientes de administración libres que son las más usadas por los usuarios del gestor PostgreSQL, el pgAdmin y pgAcces destacándose el primero por el avanzado desarrollo y la cantidad de personas que lo utilizan.

1.3.4 pgAccess

”PgAccess es un software libre y está programado utilizando las librerías tcl/tk por lo que puede correr en cualquier plataforma a la que haya sido portado tcl/tk. Permite ejecutar VACUUM (comando utilizado para limpiar y opcionalmente analizar una base de datos) en una base de datos Postgres. Además posibilita la edición, adición y navegación de bases de datos, tablas, vistas, funciones, secuencias y usuarios, así como también el desarrollo gráfico de consultas.” [12] Esta herramienta no es muy efectiva porque no se le brinda soporte frecuentemente. Además carece de opciones para realizar operaciones sobre bases de datos en PostgreSQL, provocando que administradores y desarrolladores no queden satisfechos por ejemplo, no proporciona la posibilidad de revisar línea a línea el código en busca de errores y tampoco la creación y diseño de una base de datos de forma visual. Algo que decepciona a las personas que interactúan con pgAccess es que el ambiente visual que provee este cliente es pobre y poco amigable.

1.3.5 pgAdmin

“pgAdmin es una aplicación gráfica para trabajar con el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia Open Source. Está programada en C++ usando la librería gráfica multiplataforma wxWidgets. Se encuentra en los repositorios de las distribuciones de Ubuntu, aunque se puede descargar el código fuente de su última versión en su sitio oficial (www.pgadmin.org). La aplicación también incluye un editor de resaltado de sintaxis SQL, la conexión al servidor se puede hacer a través de TCP / IP o Unix Domain Sockets. PgAdmin es desarrollado por una comunidad de expertos de PostgreSQL en todo el mundo y está disponible en más de una docena de idiomas. Es un software libre publicado bajo la licencia de PostgreSQL.” [13]

Aunque es la herramienta más usada para el trabajo con PostgreSQL la interfaz visual de ella no le permite interactuar fácilmente con el gestor por ejemplo, el usuario no posee la opción de personalizar el ambiente de trabajo de manera que se muestren las ventanas de la aplicación según sus necesidades. Para diseñar bases de datos es preciso obtener nuevas herramientas dedicadas específicamente a esta tarea, debido que esta herramienta actualmente no dispone de un editor capaz de mostrar tablas donde se le agregarán sus atributos, llaves, crear relaciones entre ellas y generar el código automáticamente de toda la representación visual sin tener conocimientos previos de sentencias SQL. Además los usuarios

que interactúan con esta herramienta emplean mucho tiempo en resolver errores que surgen del trabajo con el gestor ya que este cliente carece de un depurador de consultas (debugger) para analizar el código SQL línea a línea permitiendo un análisis detallado del mismo y encontrar de manera sencilla el origen del error.

De las herramientas libres existentes para la administración de las bases de datos descritas anteriormente, se decide no tomar el código fuente de ninguna como base para realizarle modificaciones. La decisión antes tomada se debe a que la integración de nuevas soluciones se dificulta en ambas aplicaciones porque no poseen una arquitectura basada en componentes, que permita incorporarle las nuevas funcionalidades de una forma simple. Sucede que el tiempo asignado para la realización de nuevas aplicaciones es corto, provocando que muchas de las funcionalidades no puedan ser unidas a estos clientes quedando en desuso o solamente como uso independiente.

Los sistemas de hoy en día son cada vez más complejos, pues deben ser construidos en tiempo récord y además cumplir con los estándares más altos de calidad. Una estrategia para avanzar en este sentido es el desarrollo de software basado en plug-ins, lo que posibilita que una vez que sea creado un componente para solucionar una tarea específica exista un bajo acoplamiento o dependencia con otros componentes, de manera que permita realizar modificaciones al código sin necesidad de transformar y afectar el funcionamiento completo del sistema. Simplifica las pruebas pues admite que las mismas sean ejecutadas probando cada uno de los componentes de manera separada antes de probar el conjunto completo de componentes ensamblados. Provee mayor calidad en el software dado que un componente puede ser construido y luego mejorado continuamente por un experto u organización, lo que contribuye a que la calidad de este mejore constantemente con el paso del tiempo.

1.4 Metodologías de Desarrollo

El uso de una metodología de desarrollo que guíe el ciclo de vida del software es necesario para garantizar la satisfacción del cliente. No existe una metodología que abarque todo lo que respecta a la ingeniería de software y que permita hacer frente con éxito a cualquier proyecto de desarrollo de aplicaciones informáticas. Toda metodología debe ser adecuada al contexto del proyecto en el que se va a aplicar.

“La metodología de desarrollo de software en Ingeniería de Software es un marco de trabajo usado para estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. Es un conjunto de procedimientos, técnicas, herramientas, y un soporte documental que ayuda a los desarrolladores a realizar nuevo software. ” [14]

Estas se pueden enmarcar en dos grupos: tradicionales (pesadas) y ágiles (ligeras). Las metodologías tradicionales se centran especialmente en el control del proceso, estableciendo rigurosamente las actividades involucradas, los artefactos que se deben producir así como las herramientas y notaciones que se usarán. Las metodologías ágiles dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque está mostrando su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad.

1.4.1 Metodología RUP (Proceso Unificado de desarrollo de software)

Es una de las metodologías más utilizada en la actualidad para proyectos a medio y largo plazo. RUP divide en cuatro fases el desarrollo del software, en las cuales se realizan actividades que van dando lugar a los artefactos necesarios para el avance del proyecto. Estas actividades son conocidas como flujos de trabajo, los cuales tienen mayor o menor peso según la fase en que se encuentre el proyecto. Existen nueve flujos, de ellos los seis primeros son conocidos como flujos de ingeniería y los otros tres como flujos de apoyo.

“Los flujos de trabajo definidos por RUP son:

- ✓ Modelamiento del negocio: Describe los procesos de negocio, identificando quiénes participan y las actividades que requieren automatización.
- ✓ Requisitos: Define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen.
- ✓ Análisis y diseño: Describe cómo el sistema será realizado a partir de la funcionalidad prevista y las restricciones impuestas (requisitos), por lo que indica con precisión lo que se debe programar.
- ✓ Implementación: Define cómo se organizan las clases y objetos en componentes, cuáles nodos se utilizarán y la ubicación en ellos de los componentes y la estructura de capas de la aplicación.

- ✓ Prueba (Testeo): Busca los defectos a los largo del ciclo de vida.
- ✓ Instalación: Produce release del producto y realiza actividades (empaque, instalación y asistencia a usuarios) para entregar el software a los usuarios finales.
- ✓ Administración del proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
- ✓ Administración de configuración y cambios: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos y control de versiones.
- ✓ Ambiente: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización. ” [15]

1.4.2 Metodología Extreme Programming (XP)

“Este es el método que más popularidad ha alcanzado entre las metodologías ágiles. Esta se basa en la suposición de que es posible desarrollar software de gran calidad a pesar, o incluso como consecuencia del cambio continuo. Su principal concepción es que con un poco de planificación, un poco de codificación y unas pocas pruebas se puede decidir si se está siguiendo un camino acertado o equivocado, evitando así tener que echar marcha atrás demasiado tarde.” [16]

El uso de esta metodología proporciona muchas ventajas por ejemplo se consiguen productos fáciles de usar y con mayor rapidez, más fiables y robustos contra los fallos gracias al diseño de las pruebas de forma previa a la codificación. Por las continuas versiones que se ofrecen al usuario se atienden las necesidades del mismo con mayor exactitud. Además se reduce el número de errores debido a que los requisitos del software son fáciles de modificar obteniéndose el código más simple y entendible.

Existe una motivación por parte de los integrantes del equipo de desarrollo debido a que XP no permite excesos de trabajo (se debe trabajar 40 horas a la semana). Se concibe que la propiedad del código es colectiva, cualquiera puede desarrollar, mejorar, simplificar, cualquier necesidad del proyecto, siempre usando sistemas tipo CVS (Controlador de versiones) para evitar la duplicación de trabajo.

Para el desarrollo de la herramienta HABD se determina utilizar la metodología XP pues sus características se ajustan a las necesidades del presente trabajo. Pues se capturan los requisitos básicos y a partir de ahí se van añadiendo las funcionalidades que tanto el desarrollador como el cliente entiendan convenientes. A medida que el proyecto avanza pueden surgir nuevas expectativas o ideas que pueden ser incorporadas fácilmente permitiéndole mayor adaptabilidad al producto, con la metodología XP esto es completamente factible pues esta se adapta perfectamente a los proyectos cuyos requisitos cambian a menudo. El cliente se convierte en miembro del equipo, fortaleciendo aún más la relación entre este y el equipo de desarrollo. Se puede encontrar mayor información en el documento de la arquitectura del Proyecto PostgreSQL del centro DATEC.

Para una mejor definición de la herramienta a realizar se efectuará una combinación entre las metodologías XP y RUP. Esta combinación es permisible siempre que no altere o influya con la entrega final del producto. Para una descripción más detallada de la herramienta se han escogido artefactos definidos en la metodología de RUP tales como diagrama de clases y modelo de dominio. Se incluye además el rol de analista por el desempeño que tiene en la producción de un software de calidad y que XP no posee.

El analista se encargará de definir si hacer modelo de dominio o modelo de negocio. Será el encargado de realizar el diagrama de clases, definir las historias de usuarios y elaborar la lista de reserva del producto que va a estar compuesta por los requisitos funcionales y los no funcionales. Las tareas de la ingeniería, el plan de iteraciones también será un artefacto que debe elaborar este rol, así como el modelo de diseño que está compuesto por las tarjetas CRC. Además de la arquitectura de software que está representada por los patrones de diseño empleados.

1.5 Tecnologías y herramientas a utilizar

Para el desarrollo del cliente de administración de bases de datos que será parte del conjunto de de programas del gestor PostgreSQL Empresarial Cubano es necesario tener en cuenta las herramientas con las que se va a realizar el diseño y la implementación del mismo. A continuación se mencionan alguna de ellas: herramientas CASE (Computer-Aided Software Engineering), el lenguaje de modelado, framework de desarrollo, lenguaje de programación que se utilizará, así como el entorno de desarrollo.

1.5.1 Herramientas CASE (Computer-Aided Software Engineering) para el modelado

Visual Paradigm 6.4

Visual Paradigm for UML (Unified Modeling Language) es una herramienta CASE, multiplataforma de modelado UML, muy potente y fácil de utilizar. Posee una licencia Freeware. Aporta a los desarrolladores de software una plataforma de desarrollo para construir aplicaciones de gran calidad y rapidez. Te permite dibujar todo tipo de diagramas UML, revertir código fuente a modelos UML y generar código fuente desde dichos diagramas.

Soporta el ciclo de vida completo de desarrollo de un software, desde la fase de análisis hasta el despliegue del mismo. Además de la generación automática de informes en formato PDF, Word o HTML. Esta herramienta además se puede integrar con diversos IDE's como NetBeans (de Sun), Developer (de Oracle), Eclipse (de IBM), JBuilder (de Borland). Fácil de instalar, utilizar y actualizar.

Rational Rose

Rational Rose es una herramienta CASE muy potente para el modelado. Todos los productos Rational Rose brindan soporte al lenguaje de modelado UML el cual proporciona un lenguaje común para que la creación de software de calidad sea más rápida. Permite la generación de código Ada, ANSI C ++, C++, CORBA, Java y Visual Basic. Les proporciona a los usuarios que interactúan con este software la capacidad de análisis de calidad de código. Otra característica importante que posee es el modelado UML para trabajar en diseños de base de datos, con capacidad de representar la integración de los datos y los requisitos de aplicación a través de diseños lógicos y físicos.

El principal motivo por el cual se decide utilizar Visual Paradigm y no Rational Rose es primeramente porque esta última no se encuentra disponible para distribuciones del sistema Operativo Linux. En caso que se desee utilizar es necesario instalar y configurar una máquina virtual que posea instalado sistema operativo Windows. Esto trae consigo el consumo elevado de memoria RAM y de recursos del sistema. Además no incluye integración para el IDE Eclipse entorno de desarrollo que se utiliza para el desarrollo de la aplicación HABD. Visual Paradigm se ha actualizado rápidamente en correspondencia con el nuevo

desarrollo de técnicas del lenguaje de modelado UML 2.1. También brinda ayuda a los programadores pues garantiza la generación de código para lenguajes como C++, java, php, Ada y Python.

1.5.2 Lenguaje de Modelado

El lenguaje de modelado está formado por un conjunto de símbolos que permitirán modelar parte de un diseño de software orientado a objetos. Se utiliza extensivamente en combinación con una metodología de desarrollo de software para avanzar de una especificación inicial a un plan de implementación y para mostrar y comunicar dicho plan a todo un equipo de desarrolladores. El uso de un lenguaje de modelado es más sencillo que la auténtica programación, pues existen menos medios para verificar efectivamente el funcionamiento adecuado del modelo.

Lenguaje Unificado de Modelado (Unified Modeling Language o UML)

UML es un lenguaje para el desarrollo de software orientado a objetos y su principal propósito es visualizar, especificar, construir y documentar proyectos de software. Además es un lenguaje consolidado no solo en el mundo sino también en la UCI, puesto que es el lenguaje con el cual aprenden a describir el proceso de software los estudiantes y profesores. Su estrategia consiste en fragmentar cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Para trabajar con UML no es necesario conocer del lenguaje de programación ni de las características de los proyectos, ya que UML ha sido diseñado para modelar cualquier tipo de proyectos, tanto informáticos como de arquitectura, o de cualquier otra rama.

“UML también intenta solucionar el problema de propiedad de código que se da con los desarrolladores, al implementar un lenguaje de modelado común para todos los desarrollos se crea una documentación también común, que cualquier desarrollador con conocimientos de UML será capaz de entender, independientemente del lenguaje utilizado para el desarrollo.” [17]

1.5.3 Entorno de desarrollo

Eclipse 3.7

Eclipse es un armazón o caparazón que permite montar herramientas de desarrollo para cualquier lenguaje, mediante la implementación de plug-ins adecuados, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.

La arquitectura de plug-ins de Eclipse permite, además de integrar diversos lenguajes sobre un mismo IDE, introducir otras aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo como: Herramientas de programación, depuración y compilación. Eclipse provee al programador frameworks capaces para el desarrollo de aplicaciones gráficas, por ejemplo facilita la integración con Qt para en conjunto con otras herramientas se pueda compilar el código C++ y también se pueda obtener interfaces visuales con buena calidad.

NeatBeans 6.9

NetBeans 6.9 es un IDE de desarrollo gratuito y permite escribir, compilar, ensamblar y desplegar aplicaciones, está escrito en Java pero brinda soporte para toda clase de lenguajes de programación. Las plataformas para las cuales se encuentra disponibles son: Windows, Mac, Linux y Solaris. Es un proyecto de código abierto, esto le permite acceder al código fuente, a la base de datos de errores, así como a la información sobre la creación de módulos de NetBeans propios. Al igual que eclipse este IDE permite integrarse con el framework QT. NetBeans es un entorno de desarrollo integrado (IDE por sus siglas en inglés).

Este IDE de desarrollo permite administrar las interfaces (menús y barras de herramientas) y las configuraciones del usuario, el almacenamiento (guardando y cargando cualquier tipo de dato) y las ventanas de la aplicación a través de un framework basado en asistentes (diálogos paso a paso).

Para el desarrollo de la herramienta HABD se decide utilizar el IDE de desarrollo Eclipse, pues usa plug-ins para formar una aplicación con las necesidades que poseen los usuarios que trabajan con dicho entorno; mientras que el Netbeans trae todas sus funcionalidades integradas en un mismo paquete lo que crea mucha generación de código innecesario y por la misma razón consume muchos recursos en el ordenador. Una característica importante y que hace de Eclipse una herramienta más potente es que incluye editores que generan código de apoyo los cuales ayudan a agilizar el trabajo de los desarrolladores. Actualmente existen varias empresas y desarrolladores independientes que participan

activamente en la creación de componentes de software que puede utilizar este entorno de desarrollo y que serán proporcionados a los usuarios de forma gratuita. Eso equivale a millones de dólares en actividades de desarrollo ahorrado.

1.5.4 Lenguaje de Programación

C++

C++ es un lenguaje potente basado en el lenguaje C, al que se han añadido nuevos tipos de datos, clases, plantillas, mecanismo de excepciones, sistema de espacios de nombres, funciones online, sobrecarga de operadores, referencias, operadores para manejo de memoria persistente, y algunas utilidades adicionales de librería (en realidad la librería estándar C es un subconjunto de la librería C++).

Es un lenguaje procedural (orientado a algoritmos) y orientado a objetos. Se puede decir que como lenguaje procedural se asemeja al C y es compatible con él. Como lenguaje orientado a objetos se basa en una filosofía completamente diferente, que exige del programador un completo cambio de mentalidad. El C++ depende del hardware, es uno de los lenguajes más potentes porque nos permite programar a alto y a bajo nivel. Además, ha eliminado algunas de las dificultades y limitaciones del lenguaje C.

Java

Java es un lenguaje de programación de alto nivel, moderno y orientado a objetos, entre sus características se muestra que es un lenguaje robusto, ya que proporciona numerosas comprobaciones en compilación y en tiempo de ejecución. Java es un lenguaje que no se compila en un lenguaje máquina nativo, sino en un código máquina intermedio que luego hay que interpretarlo.

Para trabajar con este lenguaje es necesario utilizar la máquina virtual de java (JVM) lo que hace que los programas compilen más lento, trabajar con esta lenguaje implica que se tenga que usar el framework de programación JNI (Java Native Interface). Este permite que que permite que un programa escrito en Java ejecutado en la JVM pueda interactuar con programas escritos en otros lenguajes como C, C++ y ensamblador. Utilizar JNI y a su vez la JVM traería consigo un mayor empleo de los recursos del sistema y uso de la memoria, por lo que disminuye el rendimiento del ordenador.

La herramienta será programada en C++ y la principal razón es porque las librerías de PostgreSQL son programadas en dicho lenguaje y de esta manera se puede lograr una integración apropiada. Además de los inconvenientes que representaría trabajar con la JVM que se explica anteriormente y la JNI principalmente porque no resulta fácil de aprender a trabajar con este framework y que pequeños errores en el uso de la JNI pueden desequilibrar completamente la máquina virtual de Java, de formas muy difíciles de corregir.

1.5.5 Framework

"Un framework, en el desarrollo de software es una estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto. "[18]

wxWidgets son bibliotecas multiplataforma y software libre, que se utilizan para el desarrollo de interfaces gráficas programadas en lenguaje C++. Están publicadas bajo licencia pública general limitada de GNU (LGPL), similar a la Licencia pública general de GNU (GPL) con la excepción de que el código binario producido por el usuario a partir de ellas, puede ser propietario, permitiendo desarrollar aplicaciones empresariales sin coste. Proporcionan una interfaz gráfica basada en las bibliotecas ya existentes en el sistema, con lo que se integran de forma óptima y resultan muy portables entre distintos sistemas operativos.

QT 4.7 es una biblioteca multiplataforma para desarrollar interfaces gráficas de usuario y también para el desarrollo de programas sin interfaz gráfica como herramientas de la consola y servidores. Utiliza el lenguaje de programación C++ de forma nativa, adicionalmente puede ser utilizado en varios lenguajes de programación a través de bindings. Funciona en todas las principales plataformas, y tiene un amplio apoyo. La API (La interfaz de programación de aplicaciones) de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL, así como uso de XML, gestión de hilos, soporte de red, una API multiplataforma unificada para la manipulación de archivos y una multitud de otros para el manejo de ficheros, además de estructuras de datos tradicionales. Esta biblioteca se encuentra distribuida bajo los términos de GNU Lesser General Public License, Qt es software libre y de código abierto.

Se decide trabajar con QT 4.7 porque presenta un buen diseño orientado a objetos o diferencia de wxWidgets. Además permite una mayor reutilización de código para agilizar el proceso de desarrollo de aplicaciones visuales, lo que trae consigo que sea más fácil el trabajo diario del programador. Responde con una rápida velocidad ya que está escrito en lenguaje C++. El API de la biblioteca cuenta con métodos para acceder a bases de datos mediante SQL que se utilizan para realizar operaciones sobre estas de una forma más sencilla.

1.5.6 Controlador de Versiones

Subversion

Un sistema de control de versiones es un sistema centralizado que se utiliza generalmente para compartir información. Permite la gestión de archivos y directorios, y sus cambios a través del tiempo, además de la conexión de múltiples usuarios al repositorio para leer o escribir.

La diferencia que existe entre CVS (Concurrent Versions System) y Subversion se puede encontrar en que CVS es uno de los paquetes tradicionales y más utilizados y Subversion es un software de sistema de control de versiones diseñado específicamente para reemplazar al popular CVS pues soluciona varias deficiencias de este.

Al Subversion también se lo conoce como SVN por ser ese el nombre de la herramienta de línea de comandos. Una característica importante de él es que, a diferencia de CVS, los archivos con versiones no tienen cada uno un número de revisión independiente. En cambio, todo el repositorio de subversion tiene un único número de versión que identifica un estado común de todos los archivos del repositorio en cierto punto del tiempo.

RapidSVN es un cliente gráfico que permite manipular los repositorios de subversion. Además es una de las alternativas más conocidas para los sistemas GNU/Linux, muy intuitivos y fáciles de utilizar.

1.6 Conclusiones del capítulo

Durante el desarrollo de este capítulo se realizó una búsqueda de información relacionada con las principales herramientas de administración que utilizan distintos SGBD, y se decide realizar una nueva herramienta basada en componentes. Se concluye realizar un híbrido entre las metodologías XP y RUP para una mejor definición del Front-End y del Plug-in “Explorador de objetos de bases de datos”. Se determinó utilizar el lenguaje de Modelado: UML, Herramientas CASE para el modelado: Visual Paradigm 6.4, lenguaje de programación: C++, Framework: QT 4.7, y el entorno de desarrollo Eclipse 3.6.

Capítulo 2: Características del sistema propuesto

Introducción

El capítulo está integrado por las fases: Exploración, donde se describe la herramienta que se propone a través de las historias de usuarios, obteniéndose la lista de reserva del producto. Esta constituye una lista que recoge los requisitos funcionales y no funcionales del software. Para el negocio se establece un modelo de dominio el cual representa los conceptos más significativos que permiten entender con claridad el problema. Además la fase de Planificación donde se exponen las tareas de la ingeniería y el plan de iteraciones. También está constituido por la fase Diseño que recoge las tarjetas CRC (clase, responsabilidad, colaboración) así como la arquitectura de software.

2.1 Identificación del problema

Todo SGBD debe proporcionar una serie de aplicaciones o programas de software que permitan administrar la base de datos de modo efectivo. Algunas de estas herramientas se utilizan para importar y exportar datos, otras para monitorear, el uso y el funcionamiento de la base de datos. Además de programas de análisis estadístico para examinar las prestaciones o las estadísticas de utilización, otros que permiten la reorganización de índices, herramientas para aprovechar el espacio dejado en el almacenamiento físico por los registros borrados y que consoliden el espacio liberado para re-utilizarlo cuando sea necesario. Cada una de estas pueden ser incluidas en los diferentes clientes de administración como el Pgadmin, EMS PostgreSQL Manager y Dream Coder for Oracle DBA. Todas son destinadas a facilitarle al usuario la interacción con el gestor y así obtener mayores beneficios de este.

Para lograr una herramienta con una buena calidad y que influya positivamente en la adopción del gestor PostgreSQL se ha realizado un estudio de varias aplicaciones que existen en el mundo para el trabajo con SGBD. De estas se toman las características comunes y diferentes de cada una de ellas y se conformará un cliente de administración de base de datos más completo e íntegro.

Es necesario un cliente que sea libre, que además de tener las características comunes que poseen los demás como la realización de consultas SQL, manipulación de los objetos de las bases de datos y analice estadísticamente el uso que se les da a estas también sea capaz debe permitir el diseño visual de las

bases de datos para que el usuario no necesite comprar o adquirir otra herramienta que brinde esta solución. La depuración de código es un factor que influye positivamente en la calidad de tal herramienta por lo que se hace indispensable que exista un profundo análisis en cuanto a este tema, así como en las herramientas visuales y de texto para elaboración de consultas. Tener presente la exportación y la capacidad de importación, la personalización de todas las ventanas del programa. Son varias las funcionalidades que debe contener para lograr una mayor aceptación. En los epígrafes posteriores se abordarán todos los temas referentes al desarrollo de los procesos de negocio, los requisitos y arquitectura de la herramienta.

2.2 Modelo de dominio

La metodología XP no define una técnica específica para definir el negocio, es así que se puede realizar este proceso de comprensión de la manera más cómoda y entendible para aquellas personas que van a trabajar durante el desarrollo de la herramienta de administración. El modelo de dominio es una técnica muy utilizada, que facilita la realización de todo software y se considera como uno de los artefactos más necesarios que debe contener cualquier proyecto.

El objetivo principal del modelo de dominio es ayudar a comprender los conceptos que utilizan los usuarios, también los conceptos con los que trabajan y con los que deberá trabajar la aplicación. Las principales razones que conllevaron a que se realizara un modelo de dominio para el análisis de la herramienta de administración fueron:

- ✓ No se logra determinar el proceso del negocio con fronteras bien establecidas donde se puedan ver claramente quiénes son las personas que desarrollan las actividades, quiénes son las que la inician y quiénes son los beneficiados con cada uno de estos procesos.

- ✓ Existe un solapamiento de roles lo que significa que no hay una clara diferencia entre los trabajadores y los actores del negocio, ya que el cliente es el propio equipo de desarrollo.

- ✓ La herramienta constituye una solución genérica que puede desplegarse en cualquier entidad que la necesita y no es para una en específico.

En el diagrama se muestra cómo el usuario interactúa con el Front-End el cual está compuesto por varias opciones de apariencia que el usuario será capaz de configurar según sus necesidades y preferencias. Además tendrá un agregado de plug-ins que irá conformando la aplicación y brindándole funcionalidades a la misma. Los plug-ins utilizan una clase conexión que es la calificada para conectarse a la librería Libpq++. Esta librería es la encargada de trasladar las consultas al SGBD que manipula y maneja las bases de datos.

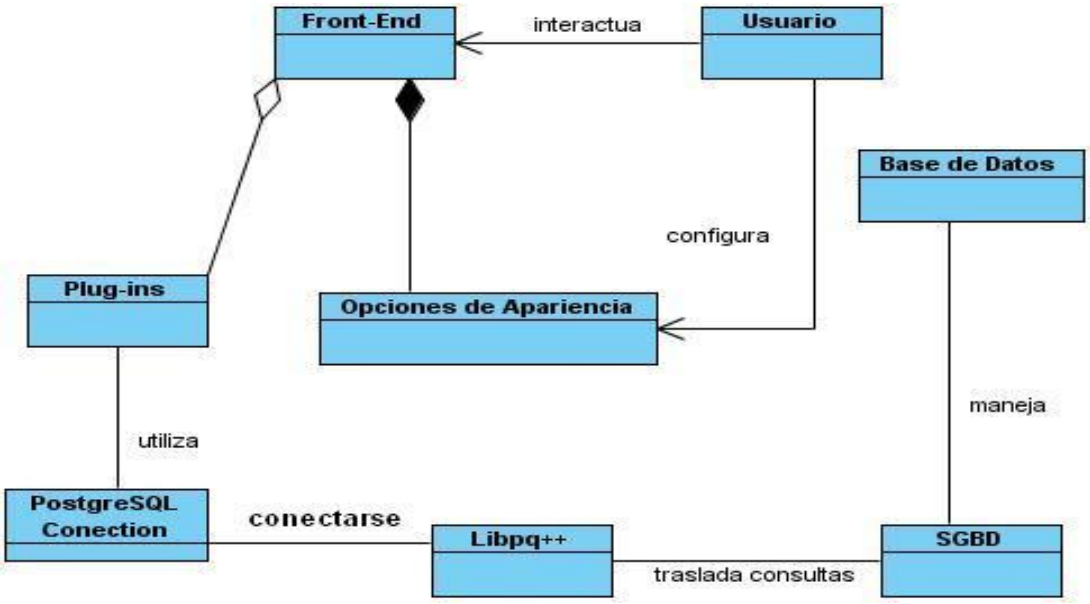


Fig. 2: Modelo de dominio del Front-End

Usuario: persona que interactúa con la herramienta.

Front-End: armazón que será capaz de cargar y manipular los plug-ins que sean adicionados por el usuario.

Plug-ins: es un módulo que le va a añadir nuevas característica y funcionalidades al Front-End y de esa manera conformará un sistema más grande.

LibPq++: es un conjunto de rutinas de biblioteca que permiten a los programas cliente trasladar consultas al servidor de Postgres y recibir el resultado de esas consultas.

SGBD: es donde se almacenan las disitintas bases de datos y las operaciones que se realizan sobre ellas.

Base de Datos: contienen los datos que serán almacenados y sobre los cuales se podrán realizar consultas para obtener la información que se necesite procesar para mostrar en la interfaz de usuario.

2.3 Descripción del sistema propuesto.

El sistema propuesto es una herramienta de administración de base de datos, que va a permitir a los usuarios trabajar con el gestor PostgreSQL, en el futuro formará parte del conjunto de programas que va a contener el PostgreSQL Empresarial Cubano. La solución estará enfocada en realizar una aplicación basada en componentes o plug-ins de forma tal que la persona que interactúa con ella pueda agregar tantas funcionalidades como necesite. Para la primera versión se desarrolla el Front-End que va a permitir administrar los plug-ins que se van desarrollando y que irán conformando la herramienta HABD y también se implementa el plug-in “Explorador de objetos de bases de datos” como muestra que la aplicación es capaz de incorporar nuevas funcionalidades.

La arquitectura basada en componentes ofrece muchas ventajas no solo al programador por el bajo acoplamiento que poseen sus componentes, sino también al usuario pues tiene y adquiere de la herramienta las funcionalidades que necesite, de esta forma garantiza la reducción de costos y mantenimiento. Esto permite además personalizar el ambiente de trabajo a sus necesidades. En caso que solicite nuevas funcionalidades o requisitos no va a resultar difícil por la flexibilidad que permite la arquitectura de la aplicación.

Se realiza primeramente el Front-End que no es más que una interfaz dinámica que le va a permitir a los usuarios interactuar directamente con las funciones del sistema. Este Front-End servirá de armazón para incorporarle plug-ins que estén en función de la herramienta de administración que se realiza. Su interfaz visual será capaz de personalizar el entorno de trabajo tanto como el usuario desee, permitiendo configurar las preferencias de las ventanas a sus necesidades. También se implementa el plug-in “Explorador de objetos” aunque no todas sus funcionalidades pero si las fundamentales para demostrar que este se incorpora al Front-End e interactúa con el gestor PostgreSQL.

2.4 Estándares de codificación

XP enfatiza la comunicación de los programadores a través del código, con lo cual es indispensable que se sigan ciertos estándares de codificación (del equipo, de la organización u otros estándares reconocidos para los lenguajes de programación utilizados). Los estándares de programación mantienen el código legible para los miembros del equipo, facilitando los cambios.

Las siguientes pautas de programación están enfocadas a la estructura y apariencia física del código que se va a generar en la creación de la HABD. Este estándar va a servir para facilitar la lectura, comprensión y mantenimiento del código para la programación en lenguaje C++.

Para los comentarios

Cada programa deberá comenzar con un comentario que incluya: autor, fecha, objetivo, o problema que resuelve el programa. También debe contener fecha de creación y bitácora de versiones con las dos últimas fechas de modificación y el algoritmo.

Cada función debe tener un encabezado que contenga: objetivo de la función y no descripción del procedimiento, comentarios de apoyo a variables, llamadas a función o inclusión de archivos que no sean obvios al proceso. Además de la explicación de uso de argumentos (parámetros) no obvios y explicación de uso de valores devueltos (de retorno).

Para los nombres de identificadores.

Se considera como identificador a los nombres de variables (arreglos, matrices, apuntadores), funciones, así como cualquier tipo de dato definido por el usuario (estructura, clase). Dichos identificadores deberán seguir las siguientes normas, además de las definidas por el propio lenguaje.

Estos deberán tener un nombre significativo que se pueda conocer su función después de una simple lectura. Para nombres que se usen con frecuencia o para términos largos, se recomienda usar abreviaturas estándar para que éstos tengan una longitud razonable. Si usa abreviaturas deben manejar

la misma lógica en todo el programa. Evitar identificadores que comiencen con uno o dos caracteres de subrayado para prevenir que se confundan con los que el compilador selecciona.

Los identificadores de variables comenzarán siempre con la primera letra minúscula correspondiente a su tipo de dato y para distinguir palabras dentro del nombre deberá emplearse un guión bajo (_). El nombre de los identificadores de punteros (apuntadores) deberá comenzar con la letra p. El nombre de los identificadores de variables dimensionadas (arreglos, matrices) deberá comenzar con las letras ar. Los identificadores de datos constantes serán declarados en letras mayúsculas. La primera letra de identificadores de funciones deberá ser mayúscula.

2.5 Estándares de diseño.

Las interfaces de usuario representan la vía por la cual se puede interactuar con las aplicaciones informáticas por tanto es necesario que sean diseñadas de manera que faciliten la comunicación USUARIO - SISTEMA y SISTEMA - USUARIO.

Las **ventanas de ingreso al sistema** son aquellas que permiten al sistema autenticar y autorizar a los usuarios para utilizar un sistema de información determinado. La barra de título deberá mostrar una imagen (icono) alusiva a la funcionalidad que tiene esta interfaz en la parte izquierda. Además debe contener un título para la ventana. También debe estar conformada por una barra de estado que muestre, de izquierda a derecha: el logo de la institución y los diferentes mensajes de ayuda al usuario.

Las **ventanas de menú principal** son aquellas que se muestran una vez que se ha completado el inicio de sesión en el sistema de forma satisfactoria, es decir, el sistema ha autenticado y autorizado al usuario para que pueda operar la aplicación. La **barra de título** debe tener incorporado en su parte izquierda un ícono representativo de la aplicación. Debe mostrar un título, el cual estará conformado por: NOMBRE DE LA APLICACIÓN COMPLETO.

Las **ventanas de procesos** son todas aquellas que posibilitan al usuario la ejecución de acciones en la aplicación. La barra de título debe mostrar como título: SIGLAS DE LA APLICACIÓN + UN GUIÓN (“-“) + NOMBRE DEL PROCESO O FUNCIONALIDAD DE LA VENTANA. El nombre del proceso se define a

criterio del diseñador. Deben contener además una barra de estado que permita mostrar, de izquierda a derecha: el logo de la institución y mensajes al usuario como medio de información o ayuda. El uso de los botones minimizar, maximizar y cerrar se mostrarán a la derecha barra de título, todas las ventanas deben ser consistentes en estas características.

2.6 Fase de exploración. Definición

El ciclo de vida de XP enfatiza en el carácter iterativo e incremental del desarrollo. Una iteración de desarrollo es un período de tiempo en el que se realiza un conjunto de funcionalidades determinadas, que en el caso de XP corresponden a un conjunto de historias de usuarios.

La metodología de desarrollo XP comienza con la fase de exploración, en esta fase los clientes plantean a grandes rasgos las historias de usuario mediante un proceso de identificación que son de interés para la primera entrega del producto. Al mismo tiempo el equipo de desarrollo se familiariza con las herramientas, tecnologías y prácticas que se utilizarán en el proyecto.

Se prueba la tecnología y se exploran las posibilidades de la arquitectura del sistema construyendo un prototipo. La fase de exploración toma de pocas semanas a pocos meses, dependiendo del tamaño y familiaridad que tengan los programadores con la tecnología.

2.6.1. Historias de Usuario

Uno de los artefactos más importantes que genera la metodología XP son las historias de usuario (HU). Estas tienen el mismo propósito que los casos de uso y son escritas por el propio cliente, tal y como ven ellos las necesidades del sistema, por tanto son descripciones cortas y escritas en el lenguaje del usuario sin terminología técnica. Para el presente trabajo de diploma se obtiene un total de 8 HU que serán realizadas en 4 iteraciones, aquí se muestra la HU Administrar Plug-ins, en los anexos se encuentran 3 más que son las implementadas y el resto en el documento Plantilla de Historias de usuarios del Expediente de proyecto.

Historia de Usuario	
Número: 1	**Nombre de la Historia de Usuario:** Administrar Plug-ins.
Cantidad de modificaciones a la Historia de Usuario: Ninguna.	
Puntos estimados: 2 semanas	**Iteración asignada:** 1ra Iteración.
Descripción: El usuario tendrá la posibilidad de instalar plug-ins a la aplicación según sus necesidades y una vez instalados se le permite activar o desactivar. También podrá eliminar los plug-ins que ya no quiera tener. El plug-in se mostrará en el Front-End en dependencia del estado en que se encuentre(activado o desactivado)	
Observaciones: El plug-in solo realiza las operaciones para las que se ha creado si se encuentra activado.	

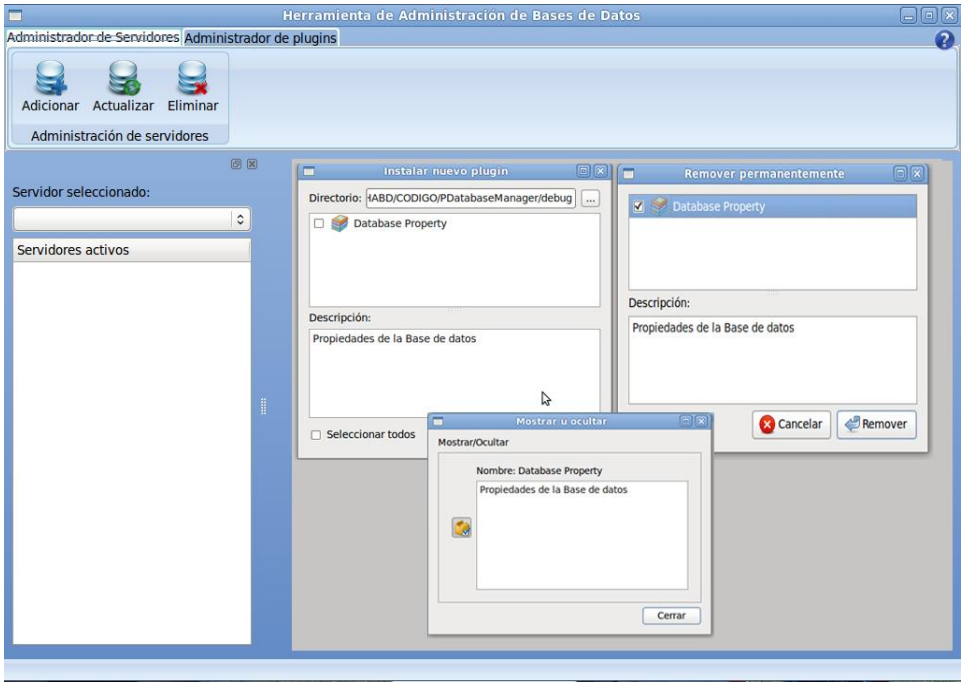


Tabla 1: Historia de Usuario "Administrar Plug-ins".

2.6.2 Lista de reserva del producto

La lista de reserva del producto es una tabla que contiene los reuisitos funcionales que debe cumplir la aplicación que se desea realizar, ordenados según la prioridad de implementación, ubicados en Muy Alta, Alta, Media y Baja, en esta última aparecen los requisitos de menor complejidad además de los requisitos no funcionales del sistema a desarrollar. Indica además la estimación de cada uno de los ellos y su implementación por semanas además del rol que hizo la estimación.

Esta lista de reserva recoge los requisitos correspondientes al Front-End que permitirá cargar los plug-ins y también algunos del plug-in "Explorador de objetos de base de dato" que es de vital importancia para la manipulación de las bases de datos. El plug-in "Explorador de objetos de la base de datos" para el presente trabajo de diploma tendrán implementado algunas funcionalidades que demuestren que él es capaz de interactuar con el Front-End e integrarse.

Número	Descripción	Estimación	Estimado por
Prioridad: Alta			
1	**Administrar Plug-ins.**	2 semanas	Analista
1.1	Instalar Plug-ins.		
1.2	Desintalar Plug-ins.		
1.3	Activar Plug-ins.		
1.4	Desactivar Plug-ins.		
2	**Personalizar ambiente de trabajo**	2 semanas	Analista
2.1	Cambiar apariencia.		
2.2	Cambiar ubicación de los servicios.		
2.3	Embeber ventanas en la plataforma.		
2.4	Extraer ventanas de la plataforma.		
3.0	**Gestionar base de datos.**	2 semanas	Analista
3.1	Registrar base de datos.		
3.2	Eliminar base de datos.		

3.3	Modificar base de datos.		
3.4	Conectar base de datos.		
3.5	Desconectar base de datos.		
4	**Gestionar Servidor.**	2 semanas	Analista
4.1	Registrar servidor.		
4.2	Eliminar Servidor registrado.		
4.3	Conectar Servidor.		
4.4	Desconectar Servidor.		
4.5	Actualizar Servidor.		
4.6	Mostrar propiedades del servidor conectado.		
5	**Gestionar Script.**	8 semanas	Analista
5.1	Crear Script.		
5.2	Cargar Script.		
5.3	Ejecutar Script.		
5.4	Guardar Script.		
5.5	Modificar Script.		
6	**Gestionar usuarios.**	2 semanas	Analista
6.1	Crear usuario.		
6.2	Eliminar usuario.		
6.3	Modificar usuario.		
7	**Gestionar grupos de usuarios.**	2 semanas	Analista
7.1	Crear grupo de usuarios.		
7.2	Eliminar grupo de usuarios.		
7.3	Modificar grupo de usuarios.		
8	**Mostrar objetos generales**	1 semana	Analista
8.1	Mostrar base de datos.		
8.2	Mostrar usuarios.		
8.3	Mostrar grupos de usuarios.		
8.4	Mostrar tablespaces.		

8.5	Mostrar lenguajes.		
Prioridad: Baja. Requisitos no funcionales.			
1	**Apariencia o interfaz externa:** Interfaz intuitiva, amigable, organizada, con una navegabilidad flexible y de fácil comprensión, de forma que el objetivo del sistema para con los clientes, se pueda conseguir rápidamente. Debe estar diseñada para verse en cualquier resolución igual o inferior a 1024x768. El diseño gráfico debe estar acorde con las pautas de diseño de la Universidad.		Analista
2	**Facilidad de uso:** Para utilizar el sistema es necesario poseer conocimientos elementales de computación, así como de base de datos.		Analista
3	**Soporte**: Se dispondrá de documentación técnica que describa todas las funcionalidades del sistema, de modo que existirá un manual para el uso de la aplicación orientada a clientes y usuarios finales. Sistema Multiplataforma con licencia GPL/, PostgreSQL 8.4.x o versiones.		Analista
4	**Software**: Sistema Operativo: Multiplataforma. Librerías QT. Servidor de Bases de datos: SGBD PostgreSQL 8.4.x o versiones superiores.		Analista
5	**Hardware:** Se necesita 100 MB de memoria RAM mínimo, 1GB de espacio libre en el disco duro para su instalación y el micro a 300 MHz.		Analista
6	**Seguridad:** Garantizar que cada conexión que se realice al servidor de base de datos solicite autenticación. Permitir que las funcionalidades del sistema se muestren de acuerdo al tipo de usuario que esté activo. Permitir		Analista

	hacer salvas y restauración cuando se caiga la conexión con el servidor.		

Tabla 2: Lista de reserva del producto.

2.6.3 Tareas de la Ingeniería.

El equipo de desarrollo evalúa cada escenario, entiéndase por escenario historias de usuarios o requisitos y lo divide en tareas donde cada una de estas representa una característica del sistema. A continuación aparece una tarea de la ingeniería (TI) donde se muestra el encargado de programarla y una descripción breve de lo que se necesita. La siguiente tabla muestra la TI Instalar plug-in y el resto de ellas se encuentran en el documento correspondiente a las TI del expediente de proyecto.

Tarea de ingeniería	
Número de tarea: 1	**Número de tarea:** 1
Nombre de tarea: Instalar plug-in.	
Tipo de tarea : Desarrollo	**Puntos estimados:** 3 semanas
Programador responsable: Dairon Domínguez Vega	
Descripción: El usuario tendrá la posibilidad de instalar nuevos plug-ins los cuales no se mostrarán inmediatamente sino que van a estar almacenados para que el usuario utilice el que necesite.	

Tabla 3: TI Instalar Plug-in.

2.7 Fase de Planificación. Definición.

La metodología XP plantea la planificación como un diálogo continuo entre las partes involucradas en el proyecto, incluyendo al cliente, a los programadores y a los coordinadores o gerentes. Es una fase corta, en la que el cliente, los gerentes y el grupo de desarrolladores acuerdan el orden en que deberán implementarse las historias de usuario, y asociadas a estas, las entregas. Comúnmente esta fase consiste en una o varias reuniones grupales de planificación.

2.7.1 Plan de iteraciones.

Después de tener ya definidas las historias de usuario es necesario crear un plan de iteraciones, en inglés "Release plan", donde se indiquen las historias de usuario que se crearán para cada versión del programa. Un "Release plan" se utiliza para la planificación, donde los desarrolladores y clientes establecen los tiempos de implementación ideales de las historias de usuario, además de determinar la prioridad con la que serán implementadas y las historias que serán implementadas en cada versión del programa. La siguiente tabla es el plan de release elaborado para el Front-End y el plug-in de la herramienta HABD.

Liberación	Descripción de la iteración	Orden de la HU a implementar	Duración total
1	El objetivo de esta iteración es crear el armazón que va a permitir cargar los distintos plug-ins. Además este debe permitir la administración de ellos y personalizar el ambiente de trabajo incorporándole un .qss a la aplicación.	1 y 2	4 semanas
2	El objetivo de esta iteración es que el plug-in “Explorador de objetos de bases de datos” se integre al front-end o caparazón. Permitiendo gestionar la conexión a través de la Gestión de los servidores y mostrando los objetos generales que posee este. Además de corregir errores encontrados en la iteración anterior.	3 y 4	4 semanas
3	El objetivo de esta iteración es conseguir que el plug-in explorador de objetos de base de datos permita Gestionar el script de las bases de datos. Además de corregir errores encontrados en la iteración anterior	5	8 semanas

4	El objetivo de esta iteración es que el plug-in pueda Gestionar los usuarios, los grupos de usuarios y las bases de datos. Además, de corregir errores o disconformidades de las HU implementadas en la iteración anterior.	6, 7 y 8	5 semanas

Tabla 4: Plan de Release.

2.8 Fase de diseño. Definición.

En XP la idea es que el diseño final sea la forma más simple de agrupar todos los requisitos. Un diseño complejo puede influir negativamente en el avance del desarrollo del producto de software debido a que los requisitos pueden cambiar constantemente. Hay que diseñar lo que las necesidades del problema requieren, no lo que se cree que deberá ser el diseño.

2.8.1 Modelo de diseño. Tarjetas CRC.

Las tarjetas CRC están divididas en 3 partes:

Clase: representa una colección de objetos similares.

Responsabilidades: es aquello que la clase sabe o hace. En ocasiones la misma no tiene toda la información para hacerlo. Esto hace que deba interactuar con otras clases.

Colaboración: toman 2 formas un pedido de información o un pedido a que se realice una acción.

La tabla que se muestra a continuación se corresponde con la tarjeta CRC Plug-in, el resto de ellas se encuentran en el documento Modelo de diseño del Expediente de Proyecto.

Tarjeta CRC	
Clase: Plug-in	
Responsabilidades	**Colaboraciones**
Instalar plug-ins. Desinstalar plug-ins. Adicionar plug-ins. Eliminar plug-ins.	

Tabla 5: Tarjeta CRC “Plug-ins”.

2.9 Arquitectura de Software

Una definición reconocida de arquitectura de software(AS) es la que plantea Clements:"La AS es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión…" [19]

2.9.1 Consideraciones generales de la solución

El diseño de la solución se basa en un estilo de arquitectura basado en componentes, donde el sistema final se basará en varios componentes o plug-ins que exponen interfaces bien especificadas y que van a colaborar entre sí para resolver el problema. A continuación se muestra una definición general de cómo se va a estructurar la herramienta de administración que se va a realizar.

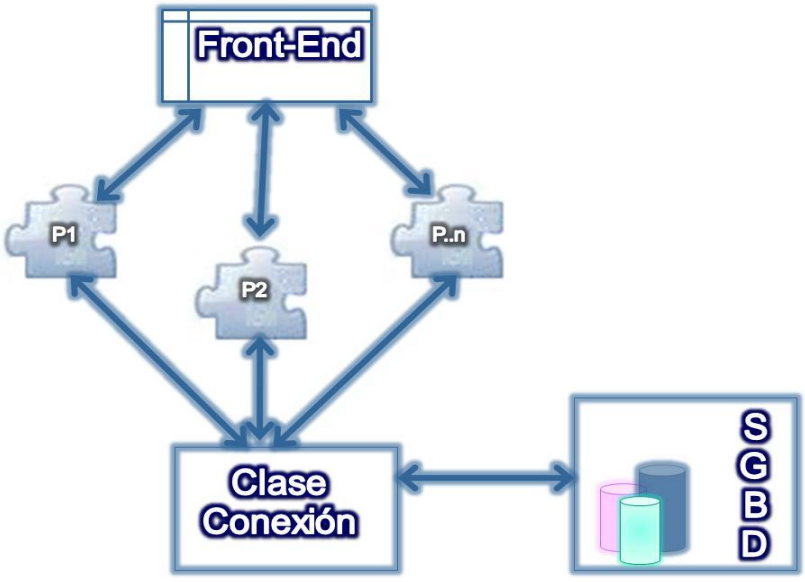


Fig 3. Consideraciones generales de la herramienta HABD

2.9.2 Arquitectura Cliente-Servidor

"La arquitectura Cliente/Servidor es aquella en la que una serie de aplicaciones cumplen funciones diferentes (una requiere servicios y la otra los brinda) pero que a la vez, pueden realizar tanto actividades en forma conjunta como independientemente. Esas dos categorías son justamente cliente y servidor." [20]

PostgreSQL se basa en una arquitectura cliente-servidor. El programa servidor se llama **postgres** y entre los muchos programas cliente se encuentran, por ejemplo, pgAdmin (un cliente gráfico) y **psql** (un cliente en modo texto).

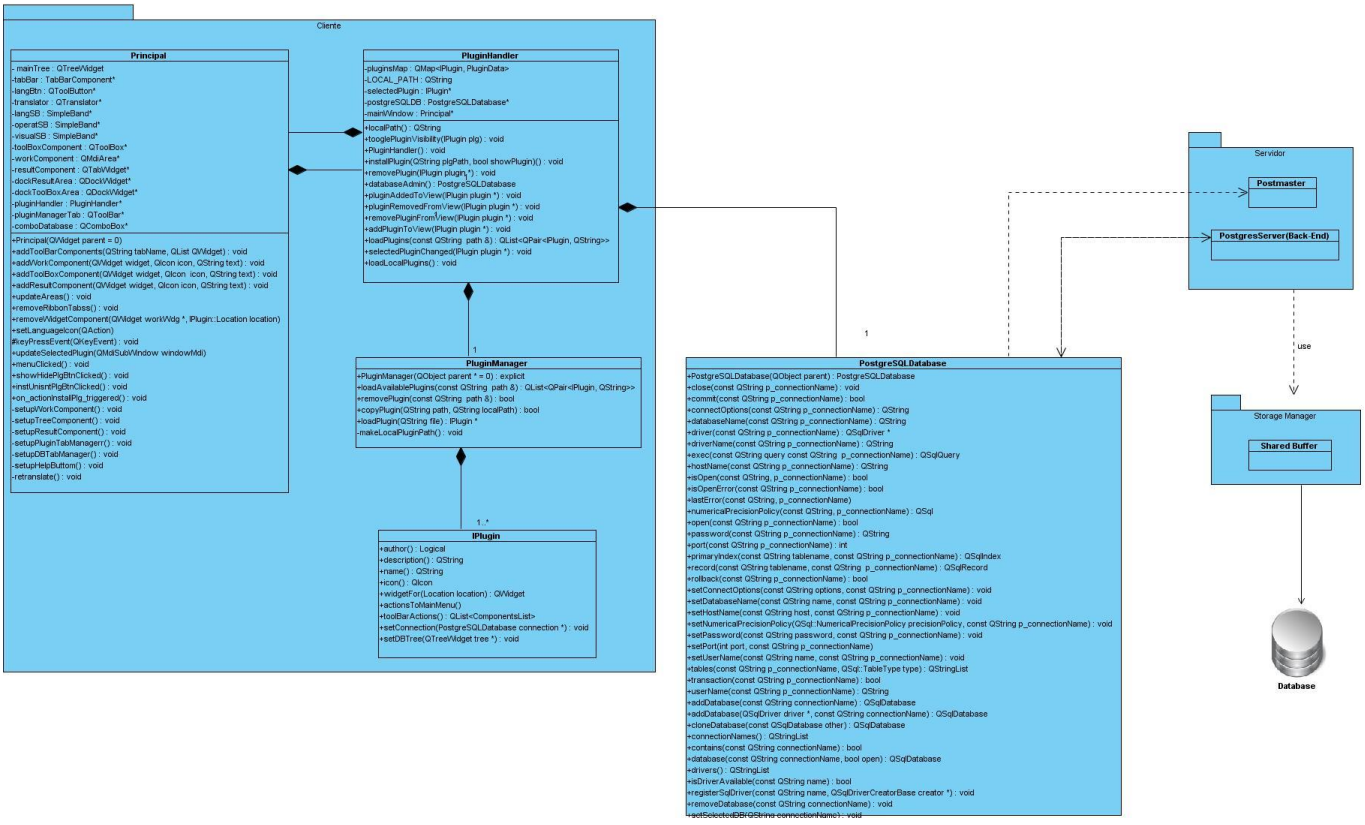


Fig.4 Empleo de arquitectura cliente-servidor

PostgreSQLDatabase es responsable de manejar la comunicación con los procesos del cliente: establecer la conexión al postmaster permitiendo la obtención de la conexión con el servidor postgres.

Servidor se compone de dos subsistemas: el postmaster y el servidor de postgre. El Postmaster es responsable de aceptar la petición de conexión entrante del cliente, de realizar control de la autenticación y de acceso en la petición. El servidor de Postgre maneja todas las consultas y comandos entrantes del cliente.

Storage Manager es responsable de la gestión de la memoria externa general.

En el cliente se muestran las clases pertenecientes al Front-End, se explica el objetivo principal de cada una de ellas en el epígrafe 2.9.4 luego del diagrama de clases correspondiente al mismo.

2.9.3 Patrones de diseño empleados

Los patrones de diseños brindan gran ayuda a la hora de resolver problemas de diseños ya que sugieren clases y objetos. Proponen además interfaces entre objetos ejemplo de esto son las clases y operaciones abstractas. Permiten la reutilización del código. Los patrones tienen la facilidad de implementarse de varias formas siempre que se mantengan estables las interfaces.

Un concepto más definido expone que: “El patrón describe un problema que ocurre infinidad de veces en el entorno, así como la solución al mismo, de tal modo que se pueda utilizar esta solución un millón de veces más adelante sin tener que volver a pensarla otra vez, de forma reutilizable.“ [21]

Patrón View-Handler

El patrón de diseño View Handler ayuda a administrar todas las vistas que proporciona un sistema de software. El mismo propone 4 componentes que son fundamentales view handler, abstract view, suppliers y la specific view. El view handler es el componente central del patrón. Es responsable de abrir nuevas ventanas, y los clientes pueden especificar la vista que quieren. En este caso el view handler es la clase Principal, el componente abastract view define una interfaz que es común a todas las vistas, esta se corresponde con la clase abstracta Iplugin la cual define el conjunto de funcionalidades que las clases que la extiendan deben implementar. El supplier proporciona los datos que son presentados por los componentes vista, en la solución que se presenta la clase PostgreSQLDatabase se corresponde con lo antes planteado puesto que se encarga de proporcionarle los datos que los plug-ins necesita mostrar. Los

componentes specifc view son derivados de abstract view e implementan esta interfaz, en este caso es la clase PDatabaseManager.

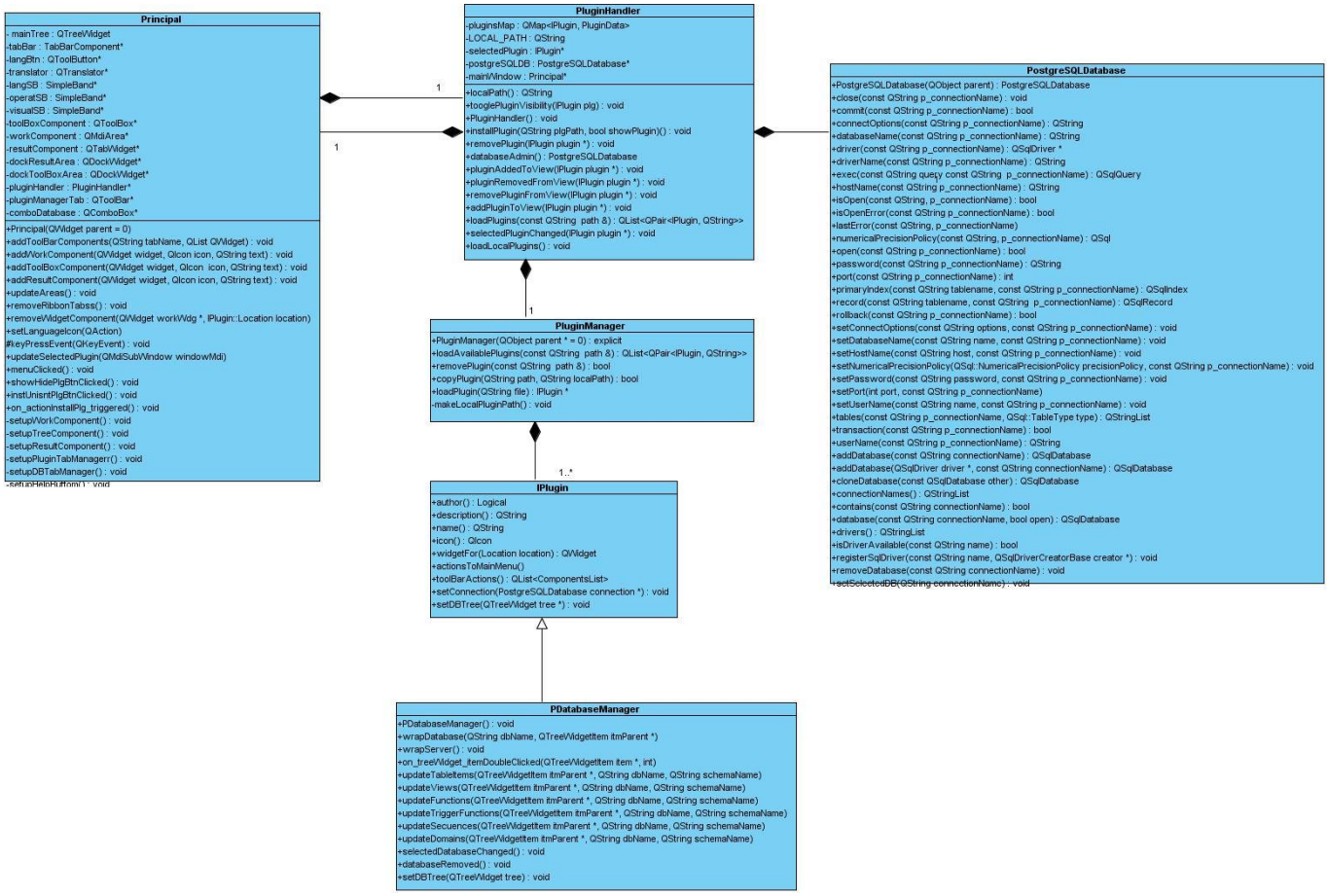


Fig. 5 Ejemplo del patrón view handler.

Patrón Creador

El patrón creador nos ayuda a identificar quién debe ser el responsable de la creación de nuevos objetos o clases. La nueva instancia deberá ser creada por la clase que tiene la información necesaria para realizar la creación del objeto, o usa directamente las instancias creadas del objeto. También si almacena o maneja varias instancias de la clase, contiene, agrega la clase y se evidencia en las clases Principal y PluginHandler.

Patrón Experto

El patrón Experto ofrece como solución, asignar las responsabilidades a las clases que tienen la información necesaria para cumplir con la responsabilidad. Ejemplo de clase que cumplan este patrón de diseño se encuentra PostgreSQLDatabase.

2.9.4 Diagrama de clases del Front-End

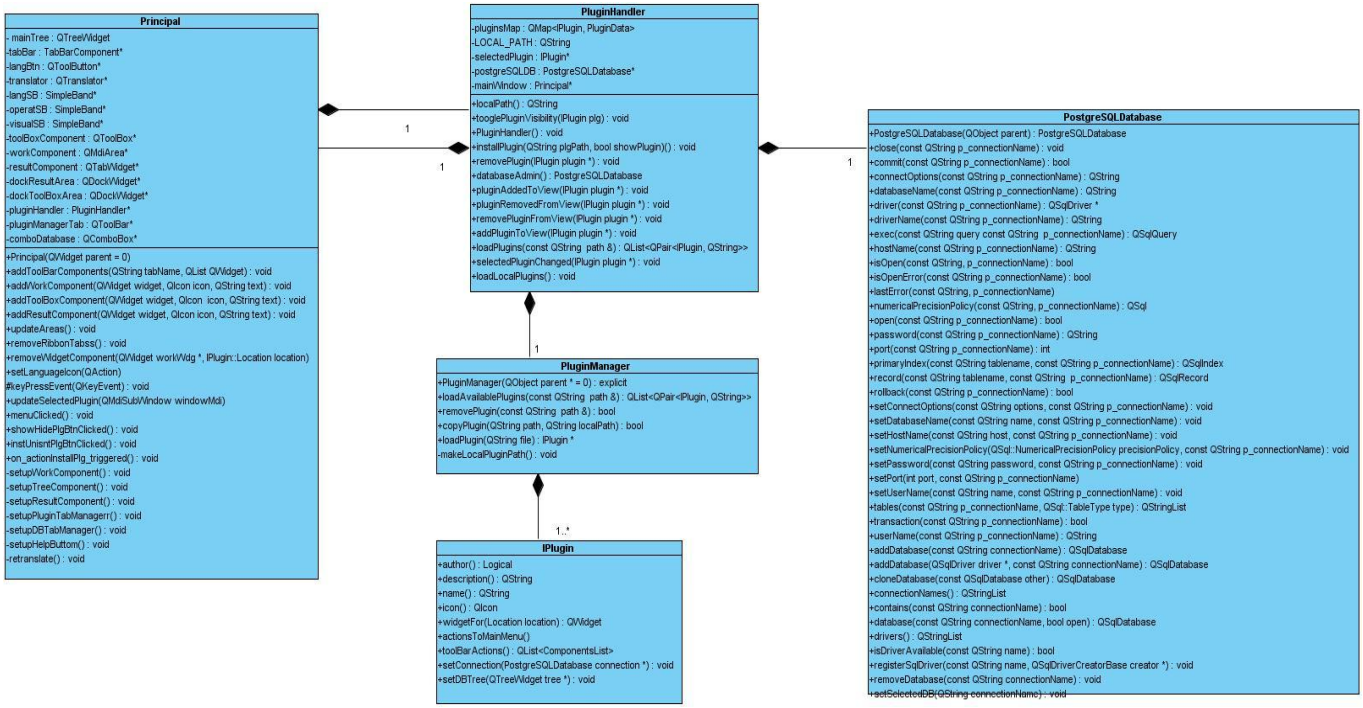


Fig.6 Diagrama de clases del Front-End.

Principal: define el área donde se colocarán las interfaces de usuario definidas, como el menú, el explorador de objetos y el área de trabajo.

PluginHandler: se encarga de administrar el conjunto de plug-ins que serán proporcionados a Principal para que los represente y fija la conexión que será utilizada por los plug-ins.

PluginManager: se encarga de cargar los plug-ins definidos guardándolos en una lista y se los proporciona a la clase PluginHandler para que esta los maneje.

IPlugin: es una clase abstracta que define el conjunto de funcionalidades que los plug-ins que se quieran incorporar al Front-End deben implementar.

2.10 Conclusiones del capítulo

En el capítulo han sido analizados los procesos fundamentales que están relacionados con la solución informática que se debe desarrollar. Se elaboró una propuesta con las perspectivas de solucionar el problema identificado y para eso fueron desarrollas las fases de exploración y diseño propuestas por la metodología utilizada en las que se definieron un total de 8 historias de usuarios, las cuales agrupan un total de 35 requisitos funcionales. También se muestra el plan de iteraciones el cual contiene el orden en que ellas deben ser implementadas durante el desarrollo y se describe la arquitectura a través de los patrones de diseño empleados.

Capítulo 3: Validación del sistema propuesto

Introducción

En el capítulo se describe el método que se utiliza para diseñar los casos de pruebas que guiarán la validación de la aplicación. El método que se utiliza es caja negra, la técnica es partición de equivalencia. Además se encuentra la tabla de las no conformidades encontradas en la aplicación las cuales fueron resueltas en un período de 10 días.

El desarrollo de un producto de software implica la realización de una serie de actividades encaminadas a encontrar errores. Incorporar acciones que evalúen la aplicación que se está desarrollando es de vital importancia para lograr un producto de software con la calidad requerida. El flujo de trabajo de prueba, es mediante el cual se puede validar que las suposiciones hechas en el diseño y los reuisitos se estén cumpliendo satisfactoriamente, por lo que se encarga de verificar que el producto funcione como se diseñó y que los requisitos se cumplan cabalmente.

3.1 Enfoques de diseños de pruebas

“Existen tres enfoques principales para el diseño de casos de pruebas:

- ✓ El enfoque estructural o de caja blanca. Se centra en la estructura interna del programa (analiza los caminos de ejecución).
- ✓ El enfoque funcional o de caja negra. Se centra en las funciones entradas y salidas.
- ✓ El enfoque aleatorio consiste en utilizar modelos (en muchas ocasiones estadísticos) que representen las posibles entradas al programa para crear a partir de ellos los casos de prueba. “[22]

Para este trabajo de diploma se estudia el método de caja negra y en el posterior epígrafe se analizan diversas técnicas que aplica este método.

3.1.1 Método de Caja Negra

El método de caja negra se centra en los requisitos funcionales y se lleva a cabo sobre la interfaz del software. Tiene como objetivo demostrar que las funciones del software son operativas, que las entradas

se acepten de forma adecuada y se produzca un resultado correcto, teniendo en cuenta que la integridad de la información externa se mantenga. A continuación se citan diversas técnicas de dicho método.

- ✓ "**Técnica de prueba basada en grafos:** En esta técnica se debe entender los objetos que se modelan en el software y las relaciones que conectan a estos, tales como objetos de datos, objetos de programa como módulos o colecciones de sentencias del lenguaje de programación." [23]
- ✓ "**Partición equivalente:** La técnica de prueba partición equivalente divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. En otras palabras, este método intenta dividir el dominio de entrada de un programa en un número finito de clases de equivalencia. De tal modo que se pueda asumir razonablemente que una prueba realizada con un valor representativo de cada clase es equivalente, a una prueba realizada con cualquier otro valor de dicha clase.
- ✓ **Análisis de Valores Límite:** El análisis de valores límite (AVL) es una técnica de diseño de casos de prueba que completa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el AVL lleva a la elección de casos de prueba en los extremos de la clase." [24]

Para verificar que el sistema desarrollado cumple con las funciones específicas para las que se ha creado se utiliza el enfoque funcional o de caja negra y empleando la técnica de partición equivalente. La principal entrada es el plug-in "Explorador de Objetos de Base de Datos" que para la salida el sistema tiene que mostrar instalado el plug-in en el Front-End. Al comenzar el proceso de realización de pruebas lo primero que se concibe es el diseño de los casos de pruebas, que se definen según las funcionalidades descritas en las HU. Se parte de las descripciones efectuadas en las HU como apoyo para las revisiones.

Para probar las funcionalidades se diseñaron un total de 4 casos de pruebas correspondientes a las 4 primeras historias de usuarios de las 8 identificadas y descritas en la fase de exploración. Se prueban 4 HU pues son las que han sido implementadas por el grupo de desarrolladores. A continuación se muestra un ejemplo de casos de prueba para la HU Administrar plug-in.

Condiciones de ejecución: HU Administrar Plug-in

El usuario debe establecer la dirección del plug-in en el campo directorio.

Escenario	Descripción	Variable 1	Variable 2	Respuesta del sistema	Flujo central
EC 1.1 Instalar Plug-in.	El usuario decide instalar un plug-in el cual va estar disponible para activarse.	V	V	El plug-in se instala en el sistema.	1. El usuario selecciona la pestaña Administrador de Plug-in. 2. El usuario se dirige al menú operaciones y selecciona el botón Instalar. 3. El sistema muestra una interfaz para establecer la dirección donde se encuentra el plug-in. 4. El usuario selecciona el plug-in a instalar, activa el chekbox y oprime el botón Aceptar. 5. El sistema instala el plug-in.
EC 1.2 Desinstalar Plug-in.	El usuario decide desinstalar un plug-in de la aplicación.	N/A	V	El plug-in se desinstala del sistema	1. El usuario selecciona la pestaña Administrador de Plug-in. 2. El usuario se dirige al menú operaciones y selecciona el botón Remover. 3. El sistema muestra el listado de plug-ins que se encuentran instalados. 4. EL usuario selecciona el plug-in que desea eliminar. 5. El sistema elimina el plug-in.

EC 1.3 Activar Plug-in.	El usuario decide activar un plugin de la lista de plugin disponibles para activarse o desactivarse.	N/A	V	El sistema activa el plug-in.	1. El usuario selecciona la pestaña Administrador de plug-in. 2. El usuario se dirige al menú Visual y selecciona el botón Activar/Desactivar. 3. El sistema muestra una interfaz que contiene los plug-ins instalados. 4. El usuario selecciona el plug-in que desea activar. 5. El sistema activa el plug-in.
EC 1.4 Desactivar Plug-in.	El usuario decide desactivar un plugin de la lista de plugin disponibles para activarse o desactivarse.	N/A	V	El sistema desactiva el plug-in.	1. El usuario selecciona la pestaña Administrador de plug-in. 2. El usuario se dirige al menú Visual y selecciona el botón Activar/Desactivar. 3. El sistema muestra una interfaz que contiene los plug-ins instalados. 4. El usuario selecciona el plug-in que desea desactivar. 5. El sistema desactiva el plug-in.

Tabla 6. Sección a probar de la HU Administrar plug-in.

La tabla que se muestra a continuación describe las variables que se encuentran asociadas a las interfaces de la HU Administrar plug-in.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
Variable 1	Directorio	Campo de selección	No Nulo	El directorio es un campo de selección en el cual se escribe la dirección donde se
Variable 2	Plug-in de prueba	Chekbox	No Nulo	El plug-in es un archivo que tiene los formatos .so para Linux y .dll para Windows.

Tabla 7. Descripción de las variables.

Los resultados de las pruebas que no fueron satisfactorios pasaron a ser no conformidades y se emitieron en el registro de defectos y dificultades detectados. La tabla que se muestra a continuación muestra las no conformidades asociadas a la HU Administrar Plug-in que se encontraron, además de las fechas en la que se detectaron y en la que se resolvieron. El resto de los diseños de casos de pruebas, las variables asociadas a estos y las no conformidades encontradas luego de aplicarlos se encuentran en los documentos del expediente de proyecto.

Elemento	No	No conformidad	Aspecto correspondiente	Etapa de detección	Clasificación	Estado NC	Respuesta del Equip Desarrollo
La aplicación	1	La HU Administrar plug-in presenta problemas en los requisitos Activar y desactivar plug-in.	La aplicación permite realizar operaciones propias del plug-in Explorador de objetos de bases de datos como registrar, eliminar, actualizar un servidor sin estar este previamente instalado y activado.	Desarrollo	S	PD 15/11/2011 RA 25/11/2011	Procede
La aplicación	2	La HU Administrar plug-ins presenta	Interfaz que muestra al acceder a la Pestaña	Desarrollo	S	PD 15/11/2011 RA 25/11/2011	Procede

		errores ortográficos en el requisito Instalar plug-in.	"Administrador de plug-in", menú Operaciones y oprimir botón Instalar. La palabra con error es: Descripción.				
La aplicación	3	La HU Administrar plug-ins en el requisito Instalar plug-in presenta errores de validación en el campo descripción.	Interfaz que muestra al acceder a la Pestaña "Administrador de plug-in", menú Operaciones y oprimir botón Instalar. El campo descripción permite la entrada de texto.	Desarrollo	S	PD 15/5/2011 RA 25/5/2011	Procede
La aplicación	4	La HU Administrar plug-ins presenta errores ortográficos en el requisito Desinstalar plug-in.	Interfaz que muestra al acceder a la Pestaña "Administrador de plug-in", menú Operaciones y oprimir botón Remover.	Desarrollo	S	PD 15/5/2011 RA 25/5/2011	Procede
La aplicación	5	La HU Administrar plug-ins en el requisito Instalar plug-in presenta errores de validación en el campo descripción.	Interfaz que muestra al acceder a la Pestaña "Administrador de plug-in", menú Operaciones y oprimir botón Instalar. El campo descripción permite la entrada de texto.	Desarrollo	S	PD 15/5/2011 RA 25/5/2011	Procede
La aplicación	7	La HU Administrar	Interfaz que se muestra al	Desarrollo	S	PD 15/5/2011 RA 25/5/2011	Procede

		plug-in en el requisito Instalar plug-in utiliza palabras del idioma inglés.	acceder a la Pestaña "Administrador de plug-in", menú Operaciones y oprimir el botón Instalar. (Botones OK y Cancel).				
La aplicación	8	La HU Administrar plug-in en el requisito Desinstalar plug-in utiliza palabras del idioma inglés.	Interfaz que se muestra al acceder a la Pestaña "Administrador de plug-in", menú Operaciones y oprimir el botón Remover. (Botones OK y Cancel).	Desarrollo	S	PD 15/5/2011 RA 25/5/2011	Procede
La aplicación	9	Los botones de la aplicación no contienen tool tip	Cuando te paras encima de un botón que no contiene el nombre de lo que hace debe ponerse la acción que realiza.	Desarrollo	R	PD 15/5/2011 RA 25/5/2011	Procede

Tabla 8. No conformidades encontradas y resueltas.

3.2 Conclusiones del capítulo

Se analiza el método de caja negra y sus técnicas, como resultado de este estudio se diseñaron un total de 4 casos de pruebas basados en las HU que se obtuvieron en la fase de exploración de la aplicación. Se verificó el funcionamiento del Front-End aplicándole los casos de pruebas a la interfaz de este, realizándose de la misma manera al plug-in "Explorador de Objetos de ase de Datos". De las pruebas ejecutadas al sistema se obtiene un total de 16 no conformidades de ellas 3 fueron de interfaz, 2 funcionales, 3 de validación y 8 de ortografía. Todas fueron resueltas en un período de 10 días.

Conclusiones generales

Se decidió realizar una nueva herramienta basada en componentes de manera que el usuario instale en la aplicación las funcionalidades que necesita, para esto se estudiaron las herramientas y tecnologías necesarias para el desarrollo.

Se diseñó el Front-End que servirá de contenedor para cargar los plug-ins de la herramienta HABD. Esta solución va a beneficiar la interacción entre los usuarios y el gestor postgreSQL por la facilidad de uso que provee. Además le facilitará el trabajo a los desarrolladores interesados en integrar sus soluciones a esta herramienta por la flexibilidad que posee su arquitectura.

Se obtuvieron los artefactos correspondientes a las fases exploración y diseño del Front-End y del plug-in, que propone la metodología que se utilizó y otros como beneficio de la combinación entre las metodologías XP y RUP.

El equipo de desarrollo de la línea PostgreSQL implementó el Front-End y varias funcionalidades del plug-in “Explorador de Objetos de Bases de Datos” para demostrar que el Front-End es capaz de incorporar nuevos componentes o plug-ins.

Se realizaron pruebas de validación de software al Front-End y al plug-in “Explorador de objetos de bases de datos” de la herramienta HABD, para verificar que la aplicación funciona como se diseñó y que los requisitos se cumplen cabalmente.

Recomendaciones

1. Regirse por el diseño planteado en el presente trabajo para conocer como se pueden integrar nuevas funcionalidades a la aplicación.
2. Continuar con la implementación de nuevos plug-ins con el objetivo de ampliar la potencialidad de la aplicación y lograr además una mejor interacción entre el usuario y el gestor de bases de datos.
3. Incorporarle más funcionalidades al plug-in “Explorador de objetos de bases de datos” de manera que permita la manipulación de objetos lo mejor posible.

Referencias bibliográficas

[1] **Yudisney Vazquez Ortíz, Marcos Luis Ortíz Valmaseda, Aldo Cristiá Álvarez y Yeni Morgado.** Propuesta del gestor de bases de datos cubano basado en PostgreSQL. La Habana : Grupo Editorial Ediciones Futuro, 2010.

[2] **Sergio Ezequiel Rozic.** Bases de Datos. s.l. : MP Ediciones.

[3] **Ramez Elmasri, Shamkant B Navigate.** Sistemas De Bases De Datos Conceptos fundammentales Segunda Edición version en español. Arlintong, Georgia : s.n.

[4] **Medrano, Ing. Lourdes Arlín Campoy.** Tutorial Base de Datos I. [En línea] [Citado el: 29 / 11 / 2010.] http://sistemas.itlp.edu.mx/tutoriales/basedat1/tema1_1.htm

[5] **Adoracion De Miguel Castano, Mario G Piattini.** Fundamentos y Modelos de Bases de Datos. Madrid : RA-MA, 1997.

[6] **Alvares, Sara.** Sistemas gestores de bases de datos. desarrolloweb. [En línea] 31 / 7 / 2007. [Citado el: 29 de 11 de 2010.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html

[7] Facultad de ciencias matemáticas-físicas. [Online] [Citada: 25/2/2011.] http://www.fismat.umich.mx/~crivera/tesis/node11.html

[8] **Ing. Yudisney Vazquez Ortíz.** Gestor Cubano basado en PostgreSQL. Primeros resultados de su uso. [En Línea] [Citado el: 4 / 12 / 2010.] http://www.informaticahabana.cu/?q=es/node/640

[9] SQLManager.net. EMS Database Management Solutions. [En línea] [Citado el: 28 /11 / 2010.] http://download2.sqlmanager.net/download/datasheets/postgresql/manager/en/pgmanager.pdf

[10] SQLDeveloper. [En línea] 2010. [Citado el: 1 / 12 / 2010.] http://www.sqldeveloper.net/herramientas-base-datos/oracle/vision-general.html

[11] Free Download Manager. [En línea] 18 de 10 de 2006. [Citado el: 7 de 12 de 2010.]

[12] **Fredi Palominos Villavicencio.** PgAccess - a Tcl/Tk interface for PostgreSQL. [En línea] [Citado el: 7/12/2010] http://palomo.usach.cl/docshtml/postgresql/tutorial-pgaccess/index.html#tut

[13] PgAdmin PostgreSQL tools. [En línea] [Citado el: 8 de 12 de 2010.] http://pgadmin.org/.

[14] SCRIBD. [En línea] [Citado el: 2 / 12 / 2010.] http://www.scribd.com//doc/2050925/metodologias-de-desarrollo-software.

[15] Entorno Visual de Aprendizaje. [En línea] 2007. http://eva.uci.cu/mod/resource/view.php?id=34099.

[16] **José Carlos Cortizo Pérez, Diego Expósito Gil, Miguel Ruiz Leyva.** [En línea] [Citado el: 1 / 12 / 2010.] http://www.esp.uem.es/jccortizo/xp.pdf

[17] Multimania. [En línea] [Citado el: 10 de 12 de 2010.] http://usuarios.multimania.es/oopere/

[18] CodeBox. [En línea] 2008. [Citado el: 6 /12 / 2010.] http://www.codebox.es/glosario

[19] Paul Clements. “A Survey of Architecture Description Languages”. Proceedings of the International Workshop on Software Specification and Design, Alemania, 1996.

[20] **Marcos Guglielmetti, Analia Lanzillotta, Guillem Alsina, David Yanover.** [En línea] 10 de 02 de 2005. [Citado el: 11 de 04 de 2011.] http://www.mastermagazine.info/termino/4294.php.

[21] GUNNAR ÖVERGAARD, K. P. Use Cases Patterns and Blueprints, Addison Wesley Professional

[22] **Eduardo Fernández, Medina Patón.** Alarcos. *Alarcos.* [En línea] [Citado el: 12 de 5 de 2011.] http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema09.pdf

[23] **CALISOFT.** *Curso Introducción a las pruebas de software.* 2009

[24] **Camejo, Onaysi Vasallo Artigas Yislén Dolores Ramírez.** *Proceso de Pruebas de Liberación al Sistema de Manejo de Datos de Ensayos Clínicos Cubano.* Ciudad de la Habana: Universidad de las Ciencias Informáticas., 2009.

Bibliografía

1. **A., Ernesto Quiñones.** PostgreSQLPE. [En línea] [Citado el: 2 / 12 / 2010.] http://postgresql.org.pe/articles/introduccion_a_postgresql.pdf.
2. **Adoracion De Miguel Castano, Mario G Piattini.** Fundamentos y Modelos de Bases de Datos. Madrid : RA-MA, 1997.
3. **Alfonso Rodríguez, Eduardo Fernandez, Medina Mario.** Documents for Samll Business & Professionals. [En línea] 2010. [Citado el: 5 de 4 de 2011.] http://www.docstoc.com/docs/40346277/Introduccion-a-BPMN.
4. **Altamirano, Ing. Alfonso Valdez.** Comparativo de Entornos de Desarrollo Integrados. [En línea] [Citado el: 6 de 12 de 2010.] www.ubicuos.com.
5. **Alvares, Sara.** Sistemas gestores de bases de datos. desarrolloweb. [En línea] 31 / 7 / 2007. [Citado el: 29 de 11 de 2010.] http://www.desarrolloweb.com/articulos/sistemas-gestores-bases-datos.html
6. CodeBox. [En línea] 2008. [Citado el: 6 /12 / 2010.] http://www.codebox.es/glosario
7. **Estrada, Manuel Alejandro Cerón.** Gluc. *Un pequeño análisis de wxWidgets. Un framework para desarrollar aplicaciones multiplataforma .* [En línea] 15 de 6 de 2004. [Citado el: 10 de 6 de 2011.] http://gluc.unicauca.edu.co/wiki/index.php/Un_peque%C3%B1o_an%C3%A1lisis_de_wxWidgets._Un_framework_para_desarrollar_aplicaciones_multiplataforma.
8. EclipseSource. [En línea] [Citado el: 27 de 4 de 2011.] http://eclipsesource.com/en/eclipse/.
9. Entorno Visual de Aprendizaje. [En línea] 2007. http://eva.uci.cu/mod/resource/view.php?id=34099.
10. [En línea] [Citado el: 15 de 12 de 2010.] http://www.zator.com/Cpp/E1_2.htm#[1]
11. El CoDiGo K. [En línea] [Citado el: 1 de 3 de 2011.] http://www.elcodigok.com.ar/2008/10/rapidsvn-un-cliente-grafico-para-repositorios-svn/
12. Facultad de ciencias matemáticas-físicas. [Online] [Cited: 2 25, 2011.] http://www.fismat.umich.mx/~crivera/tesis/node11.html.
13. **García, Alejandro Pérez.** Adictos al trabajo. [En línea] 26 de 6 de 2010. [Citado el: 1 / 12 /2010.] http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=eclipseHelios

14. GSINNOVA. Grupo Soluciones Innova. [En línea] 2011. [Citado el: 5 de 4 de 2011.] http://www.rational.com.ar/herramientas/roseenterprise.html.

15. GUIA UBUNTU. [En línea] 6 de 6 de 2010. [Citado el: 28 / 11 / 2010.] http://www.guia-ubuntu.org/index.php?title=WxWidgets

16. **Hermes Alexy Marañón Rodríguez, Pedro David Duharte Montero.** Planificación y Diseño del Portal para la Comunidad Técnica Cubana de PostgreSQL. La Habana : s.n., 2010.

17. **José Carlos Cortizo Pérez, Diego Expósito Gil, Miguel Ruiz Leyva.** [En línea] [Citado el: 1 / 12 / 2010.] http://www.esp.uem.es/jccortizo/xp.pdf

18. **Jiménez, Andrés Bernal.** [En línea] http://www.cs.cinvestav.mx/Estudiantes/TesisGraduados/2010/tesisAndresBernal.pdf

19. **Medrano, Ing. Lourdes Arlín Campoy.** Tutorial Base de Datos I. [En línea] [Citado el: 29 / 11 / 2010.] http://sistemas.itlp.edu.mx/tutoriales/basedat1/tema1_1.htm

20. MERCADO MEDIA NETWORK. Los líderes mundiales del mercado del software. [En línea] 2011. [Citado el: 7 de 7 de 2011.] http://www.revistamercado.do/2011/02/los-lideres-mundiales-del-mercado-del-software/

21. Multimania. [En línea] [Citado el: 10 de 12 de 2010.] http://usuarios.multimania.es/oopere/

22. NetBeans. NetBeans. [En línea] 2011. [Citado el: 5 de 4 de 2011.] http://netbeans.org/community/releases/69/relnotes_es.html#known_issues-cnd.

23. **Ortíz, Ing. Yudisney Vazquez.** Comunidad cubana de PostgreSQL. [En línea] 2009.

24. [Citado el: 4 de 12/ de 2010/.] http://www.informaticahabana.cu/?q=es/node/640.

25. PgAdmin PostgreSQL tools. [En línea] [Citado el: 8 de 12 de 2010.] http://pgadmin.org/.

26. **Ramez Elmasri, Shamkant B Navigate.** Sistemas De Bases De Datos Conceptos fundammentales Segunda Edición version en españo. Arlintong, Georgia : s.n.

27. SCRIBD. [En línea] [Citado el: 2 / 12 / 2010.] http://www.scribd.com//doc/2050925/metodologias-de-desarrollo-software.

28. **Sergio Ezequiel Rozic.** Bases de Datos. s.l. : MP Ediciones.

29. SQLManager.net. EMS Database Management Solutions. [En línea] EMS. [Citado el: 28 /11 / 2010.] http://download2.sqlmanager.net/download/datasheets/postgresql/manager/en/pgmanager.pdf

30. SQLDeveloper. [En línea] 2010. [Citado el: 1 / 12 / 2010.] http://www.sqldeveloper.net/herramientas-base-datos/oracle/vision-general.html

31. Software Libre - Catamarca. [En línea] 5 de 9 de 2010. [Citado el: 30 / 11 / 2010.] http://infogeekcatamarca.wordpress.com/2010/08/05/%C2%BFque-es-la-biblioteca-qt

32. **Yudisney Vazquez Ortíz, Marcos Luis Ortíz Valmaseda, Aldo Cristiá Álvarez y Yeni Morgado.** Propuesta del gestor de bases de datos cubano basado en PostgreSQL. La Habana : Grupo Editorial Ediciones Futuro, 2010.

ANEXOS

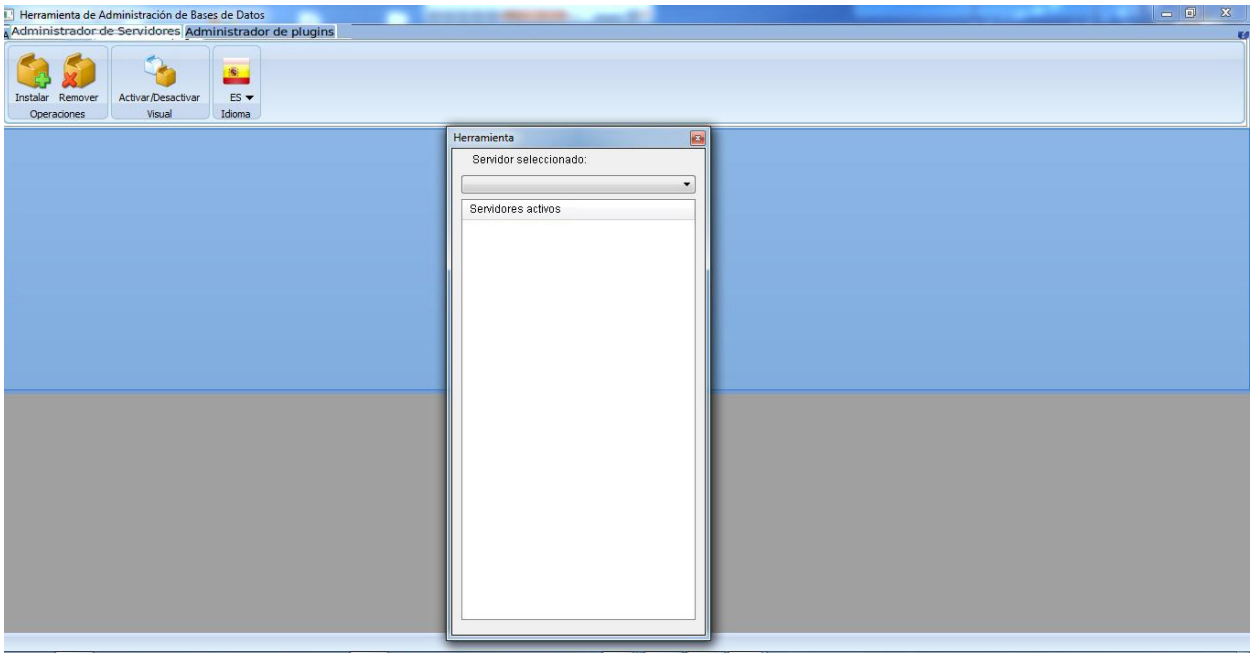
Anexo#1 Tabla aspectos importantes sobre las HABD.

HABD	SGBD	Plataforma	Licencia o forma de distribución	Precio	Tamaño
Administrador EMS SQL 2005 para PostgreSQL 3.8	Postgre SQL	Windows	Shareware	$135.00 USD	14937 K
Dream Coder for Oracle DBA	Oracle	Windows	Licencia Comercial	Varian de $6000 USD $300 USD $6500 USD	14848 K
EMS SQL Management Studio for SQL Server	SQL Server	Windows	Shareware	$518.00 USD	46725 K
PgAccess	Postgre SQL	Linux/UNIX/ otros	BSD Berkeley Software Distribution License	$ 0.0	1639.4 K
pgAdmin	Postgre SQL	Linux, FreeBSD, Solaris, Mac OSX y Windows	Licencia de PostgreSQL v1.10. Licencia Artística para versiones anteriores.	$0.0	14745.6

Anexo#2 Historia de usuario “Personalizar ambiente de trabajo”

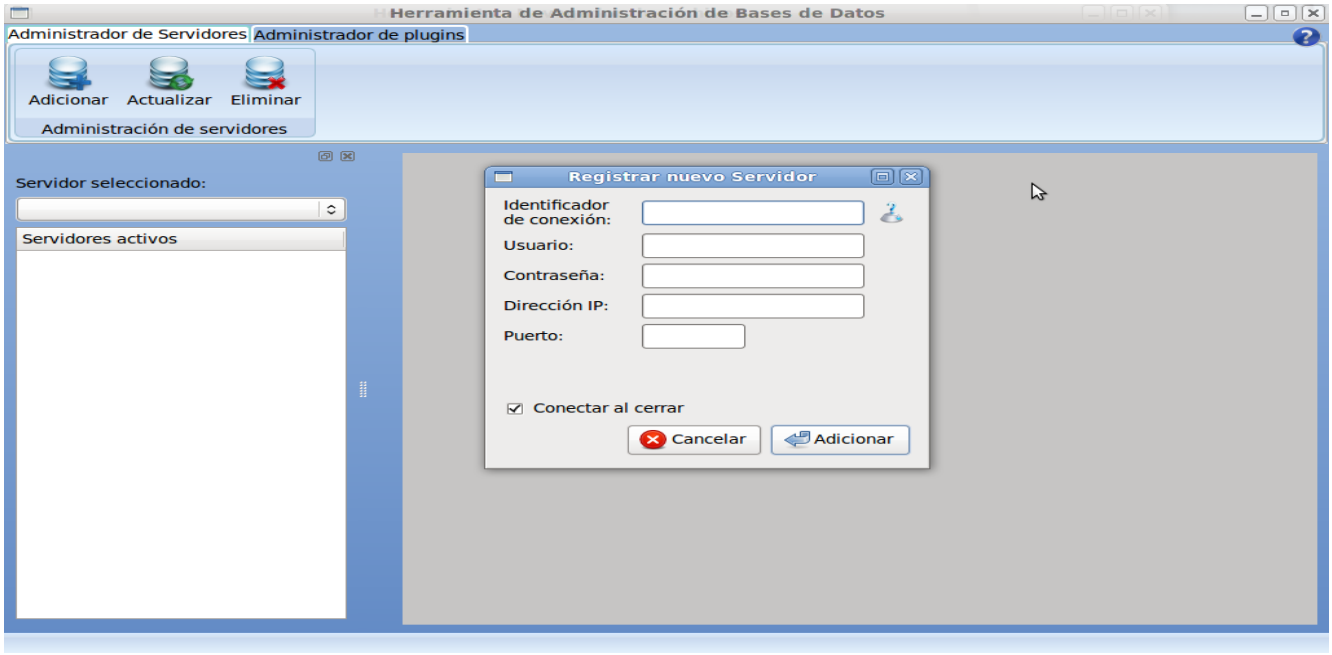
	Historia de Usuario
Número: 2	**Nombre de la Historia de Usuario:** Personalizar ambiente de trabajo

Cantidad de modificaciones a la Historia de Usuario: Ninguna.	
Puntos estimados: 2 semanas	**Iteración asignada:** 1ra Iteración.
Descripción: El usuario tendrá la posibilidad de Cambiar apariencia y ponerla el tema con los colores que prefiere. Además la aplicación debe permitirle cambiar la ubicación de los servicios entiéndase por servicios el área de resultados, el área del árbol de objetos de bases de datos. Además de dar la posibilidad de mover el área de trabajo hacia arriba, hacia abajo, a la derecha, a la izquierda. También embeber ventanas en la plataforma y extraer ventanas de la plataforma.	
Observaciones: El usuario debe ser capaz de identificar a través de los iconos como realizar cada acción que plantea esta historia de usuario que haga.	



Anexo#3 Historia de usuario “Administrar Servidor”

Historia de Usuario	
Número: 3	**Nombre de la Historia de Usuario:** Gestionar Servidor.
Cantidad de modificaciones a la Historia de Usuario: Ninguna.	
Puntos estimados: 2 semanas	**Iteración asignada:** 2da Iteración.
Descripción: El usuario tiene la posibilidad de registrar un servidor y mostrar los objetos que este tiene almacenado. Para registrarlo es necesario conocer el servidor al cual se va a realizar la conexión, el usuario, la contraseña y el nombre que se le va a dar a este servidor. Además es necesario que se actualicen los datos que se muestran al listar los servidores y eliminar el servidor que el usuario especifique.	
Observaciones: Una vez se realice la conexión debe ser mostrado en la lista de servidores.	



Anexo#4 Historia de usuario “Obtener Objetos”

	Historia de Usuario
Número:4	**Nombre de la Historia de Usuario:** Mostrar Objetos generales.
Cantidad de modificaciones a la Historia de Usuario: Ninguna.	
Puntos estimados: 4 semanas.	**Iteración asignada:** 2da Iteración.
Descripción: El sistema debe obtener los objetos generales del gestor, entiéndase por esto los roles existentes, los grupos de roles, los lenguajes, los tablespaces y las bases de datos.	
Observaciones: Estos datos se obtienen una vez que se realiza la conexión con el servidor.