Universidad de las Ciencias Informáticas

Facultad 2

Título: Complementos para evaluar resultados de auditorías a SGBD Oracle y SQL Server.

Trabajo de Diploma para optar por el Título de
Ingeniero en Ciencias Informáticas.

Autores:

Yaisel Hurtado González.

Luis Felipe Ramírez Martínez.

Tutores:

Ing. Leslye Bravo García.

Ing. Ariel Díaz Rodríguez.

Ing. Antonio Hernández Domínguez.

“Año 54 de la Revolución”

Declaración de Autoría

Declaramos que Yaisel Hurtado González y Luis Felipe Ramírez Martínez son los únicos autores de este trabajo y autorizamos al Departamento de Seguridad Informática del Centro Telemática para que haga el uso que estime pertinente del mismo.

Para que así conste firmamos la presente a los 18 días del mes de junio del 2012.

_____	_____
Firma del Autor	Firma del Autor
Yaisel Hurtado González	Luis Felipe Ramírez Martínez

_____	_____	_____
Firma de la Tutora	Firma del Tutor	Firma del Tutor
Ing. Leslye Bravo García.	Ing. Ariel Díaz Rodríguez.	Ing. Antonio Hernández Domínguez.

Datos de Contacto

Tutora: Ing. Leslye Bravo García.

Graduado del 2010 de Ingeniero en Ciencias Informáticas en la UCI, La Habana, Cuba. Actualmente es Adiestrado. Ha participado en varios eventos y cuenta con varias publicaciones en eventos científicos de carácter nacional e internacional. Fungió como Analista principal del proyecto AuditBD y actualmente se desempeña como Líder del proyecto.

Correo electrónico: lbravo@uci.cu.

Tutor: Ing. Ariel Díaz Rodríguez.

Graduado en el año 2008 de Ingeniero en Ciencias Informáticas en la UCI, La Habana, Cuba. Categoría Docente Instructor. Fue arquitecto por varios años del proyecto Informatización de la Contraloría General de la República. Ha participado y cuenta con publicaciones en varios eventos científicos de carácter nacional e internacional. Actualmente se desempeña como como Jefe de las asignaturas Programación II y Programación III del Departamento de Programación de la Facultad 2.

Correo electrónico: adrodriguez@uci.cu.

Tutor: Ing. Antonio Hernández Domínguez.

Graduado en el año 2009 de Ingeniero en Ciencias Informáticas en la UCI, La Habana, Cuba. Categoría Docente Instructor. Lideró por varios años el proyecto Informatización de la Contraloría General de la República. Ha participado y cuenta con publicaciones en varios eventos científicos de carácter nacional e internacional. Fungió como Jefe en funciones del Departamento de Seguridad Informática del Centro de Telemática.

Correo electrónico: ahdominguez@uci.cu.

"La única profesión que no necesita preparación, es la de idiota, para lo demás hay que estudiar."

Joseph Pulitzer.

AGRADECIMIENTOS

Dicen que el camino más largo comienza con el primer paso, y… es cierto. Hace casi 19 años, el 1 de septiembre de 1993, cuando no pasaba por mi cabecita ni remotamente la idea de que un día sería ingeniero, comencé mi recorrido por el camino del conocimiento en busca del tesoro del saber. Hoy, he llegado al final de este camino y en mi han quedado marcadas huellas profundas de éste recorrido. Hoy, puedo decir que soy un profesional. Y si hoy puedo decirlo, es gracias a la orientación de muchas personas, pues ya lo dice la Biblia en Proverbios 24:6: *"Porque con ingenio harás la guerra, y en la multitud de consejeros está la victoria"*.

Por eso, como una muestra de gratitud y eterno reconocimiento, por el apoyo que me han brindado y con el cual he logrado terminar mi carrera profesional, agradezco:

- A mis tutores Leslye Bravo García, Antonio Hernández Domínguez y Ariel Díaz Rodríguez por ayuda brindada, la confianza depositada y la paciencia que tuvieron para con nosotros.
- A mi madre Arelys por ser madre y padre a la vez, por apoyar y respetar mis decisiones, por confiar en mí y estar siempre presente en mi vida.
- A mi hermanito Yoenny por sentirse orgulloso de mí y tenerme como ejemplo.
- A mi novia Liamne por su apoyo, por su paciencia, por hacerme feliz y darme la oportunidad de amar. A sus padres, mis suegros, Pura y Peñalver, a su abuela Cándida, a su tía Marlene, a Jorge Chala y a Magaly su madrina.
- A mi padrastro por ser mucho más padre que mi padre.
- A mi tía Magdalena por tenerme como uno de los hijos que nunca tuvo y a su esposo Orlando.
- A mis abuelos Martha y Andrés por brindarme su ayuda sincera, muchas veces sin poder.
- A mis tíos Rolando, Aida, Aracelys y a su esposo Ramón.
- A mis primos Anisleydis, Asiel y Yamilsy.
- A mis hermanos Yordanys, Yuriesky Y Osmani.
- A mis amigos y hermanos: Willy, Luis Yandy (Louis) y Yasiel (El Kimbu) y a sus familias.
- A mis vecinos Siria y Blas.
- A mis compañeros de guerrilla: Yannier, David Delisle, Peña y Eduardo.

- A los que hemos compartido grupo, apartamento, escenario, laboratorio, proyecto y algún que otro Ranchón, a los que me iniciaron en el mundo del Software Libre, a los que creyeron y creen en mí, a los que un día me tendieron su mano.
- Al tribunal y a los oponentes (que por falta de uno tuve dos) por las sugerencias y críticas.
- A mi compañero de tesis.
- A todos los presentes.

¡Muchas gracias!

Yaisel Hurtado González.

En primer lugar quisiera agradecer a mis tutores Leslye, Antonio y Ariel por brindarme su apoyo durante la realización del Trabajo de Diploma.

A mi madre y mis abuelos por ayudarme a ser una mejor persona.

A mi compañero de tesis.

Especialmente a mi padre por enseñarme con su ejemplo, que aunque la vida te trate de la forma que no esperas debes seguir adelante.

¡Gracias!

Luis Felipe Ramírez Martínez.

DEDICATORIA

Dedico este trabajo de diploma a mi Dios, a mi madre, a mi hermanito, a mi novia y a mi familia, la de toda una vida y la nueva.

Yaisel Hurtado González.

A mis padres Aracelys y Juan Carlos y a mis niñas Eliannet y Elizabeth.

Luis Felipe Ramírez Martínez.

RESUMEN

Las auditorías de seguridad informática son de vital importancia para el Departamento de Seguridad Informática de la Empresa de Telecomunicaciones de Cuba S.A. (ETECSA), debido a que permiten determinar las posibles vulnerabilidades existentes en sus sistemas informáticos.

Por tal motivo, las auditorías a los Sistemas Gestores de Bases de Datos (SGBD) son parte sustancial de las actividades llevadas a cabo por el departamento, siendo Oracle y SQL Server dos de los principales SGBD a tener en cuenta en dichas auditorías.

En la actualidad, el departamento realiza estas auditorías de forma manual, auxiliándose de varios scripts y documentos. Parte importante durante la ejecución de las auditorías es realizar las evaluaciones de los resultados obtenidos, las cuales son realizadas por el especialista, auxiliándose de una guía. Este es un proceso engorroso debido a que en muchos casos la información obtenida es redundante y la interpretación de los resultados depende de la capacidad y experiencia del especialista. Además de que el departamento posee un reducido personal calificado para la ejecución de estas tareas.

Razones por las cuales surge la idea de desarrollar dos complementos para realizar la evaluación de los resultados de auditorías a gestores de bases de datos Oracle y SQL Server, que tienen como objetivo principal, facilitar la ejecución de la evaluación de los resultados de dichas auditorías.

En el presente trabajo de diploma se recoge la descripción del problema, se explican los conceptos relacionados, se muestra la arquitectura, herramientas y patrones de diseño a usar en el desarrollo de los complementos y se dejan algunas recomendaciones para el mejoramiento futuro de los mismos.

PALABRAS CLAVE: Auditoría, Bases de Datos, Oracle, SQL Server.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

El surgimiento de los ordenadores trajo consigo un creciente desarrollo de las Tecnologías de la Información y las Comunicaciones (TIC). A raíz de este auge, los volúmenes de información que se generaban fueron creciendo por lo que fue necesario almacenarlos y protegerlos, surgiendo así, las Bases de Datos (BD) y los Sistemas Gestores de Bases de Datos.

Una BD puede considerarse como un conjunto de datos relacionados entre sí, entendiéndose por datos los hechos conocidos, que pueden registrarse y que tienen significado implícito.
Un SGBD es un conjunto de programas que permite a los usuarios crear y mantener una BD, por lo tanto, el SGBD es un software de propósito general que facilita el proceso de definir, construir y manipular la BD para diversas aplicaciones. Estos pueden ser de propósito general o específico.[1]

Los SGBD poseen mecanismos para garantizar la integridad, la confidencialidad y disponibilidad de la información. Estos mecanismos, no inmunizan a los SGBD de los efectos de la ciberguerra llevada a cabo por piratas informáticos, en una era en donde la información, mueve el mundo y quién la controla, tiene el poder. Motivo por el cual, se ha incrementado la búsqueda por parte de las empresas, de soluciones eficientes, que brinden la posibilidad de auditar la seguridad de sus servidores de datos, con el objetivo de conocer sus puntos débiles y tratar de erradicarlos.

Existen diferentes SGBD, entre ellos Oracle y SQL Server. Estos son SGBD relacionales, capaces de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea.

Cuba a pesar de ser un país del tercer mundo ha realizado grandes esfuerzos por informatizar su sociedad. Este accionar se ve reflejado en los planes operativos de cada uno de sus ministerios, empresas e instituciones. Una de las empresas que ha logrado un mayor avance en la informatización de

[1] **Mato García, Rosa María.** *Diseño De Bases de Datos.* 2nd. 2005. pág. 2.

sus servicios es la Empresa de Telecomunicaciones de Cuba S.A. (ETECSA), la cual tiene como objetivo: Lograr en el período 2010-2015 una gestión efectiva que permita brindar servicios de telecomunicaciones que satisfagan las necesidades de los usuarios y la población, así como respaldar los requerimientos de la defensa y del desarrollo socio-económico del país con resultados económicos que de la empresa demanda y espera el país.[2]

ETECSA está integrada por varios departamentos, entre ellos se encuentra el Departamento de Seguridad Informática, el cual posee entre sus principales responsabilidades garantizar y mantener la integridad, confidencialidad y disponibilidad de los sistemas informáticos que soportan las telecomunicaciones en Cuba.

Para darle cumplimiento a lo expresado anteriormente, se realizan de forma periódica auditorías a los sistemas informáticos de ETECSA, entre los cuales se encuentran, los SGBD, especialmente Oracle y SQL Server. Estas auditorías, usando buenas prácticas y la metodología COBIT, comprueban una serie de parámetros definidos por los especialistas de ETECSA. En ellas se chequea el cumplimiento de las políticas de seguridad y se emiten reportes que reflejan el estado de la seguridad de sus servidores y se ofrecen posibles soluciones a problemas encontrados.

Actualmente las auditorías se realizan de forma manual, apoyándose en el uso de varios scripts y documentos de texto, lo que provoca demora en la culminación de dichas auditorías. El especialista obtiene los resultados luego de realizar un análisis exhaustivo con la guía, la cual posibilita la interpretación de cada uno de los mencionados resultados. Esta forma de realizar auditorías deja espacio para posibles errores debido a que la interpretación parte del criterio de los especialistas, además de que en muchos casos la información obtenida está redundante. Por otra parte una vez concluida la interpretación de los resultados obtenidos se procede a completar una matriz que contiene las vulnerabilidades encontradas en la base de datos auditada.

[2] **ETECSA.** Misión y Visión. *Portal Web de la Empresa de Telecomunicaciones de Cuba.* [En línea] ETECSA. [Citado el: 1 de Diciembre de 2011.

Las auditorías requieren gran cantidad de tiempo debido a que son realizadas por un especialista. Es importante mencionar que este proceso requiere de una rápida ejecución para solventar los posibles errores existentes.

Como se observa es un proceso lento, engorroso, en el que puede haber errores y cuya calidad depende de la capacidad y experiencia del supervisor.

En aras de resolver estos problemas se realizó un acuerdo entre el Departamento de Seguridad Informática de ETECSA y el Departamento de Seguridad Informática del Centro de Telemática de la Universidad de las Ciencias Informáticas (UCI) para desarrollar el Sistema para la realización de Auditorías a SGBD (SASGBD).

SASGBD proporciona al supervisor mejoras en cuanto a errores y tiempo en el proceso de auditoría. Sin embargo, esta herramienta carece de un mecanismo para evaluar los resultados de las auditorías a SGBD Oracle y SQL Server por lo que el proceso de evaluación continúa realizándose de la misma forma.

Por todo lo antes expuesto se plantea como **problema a resolver**: ¿Cómo proveer un mecanismo al SASGBD que permita disminuir el tiempo y los errores en la evaluación de los resultados de auditorías a los SGBD Oracle y SQL Server?

A partir del problema a resolver se puede inferir como **objeto de estudio** para esta investigación: Las evaluaciones de resultados de auditorías a los SGBD, delimitando el **campo de acción**: Las evaluaciones de resultados de auditorías a los SGBD Oracle y SQL Server en ETECSA.

La presente investigación tiene como **objetivo general**: Desarrollar una solución informática que sea capaz de integrarse con el SASGBD y que permita reducir el tiempo y los errores en las evaluaciones de resultados de las auditorías a los SGBD Oracle y SQL Server respectivamente.

De dicho objetivo general se derivan los siguientes **objetivos específicos**:

- Diseñar dos complementos que permitan evaluar los resultados de las auditorías a los SGBD Oracle y SQL Server.
- Implementar dos complementos que permitan evaluar los resultados de las auditorías a los SGBD Oracle y SQL Server.

Una vez definido el objetivo general y en aras de darle cumplimiento, se plantea la siguiente **idea a defender**: El desarrollo de los complementos para los gestores Oracle y SQL Server le brindará al SASGBD un mecanismo para evaluar los resultados de las auditorías.

Para darle cumplimiento con lo antes descrito en la presente investigación se han trazado las siguientes **tareas:**

- Realización de encuentros y entrevistas con el personal del Departamento de Seguridad Informática de la empresa cubana ETECSA, para conocer cómo se realizan los procesos relacionados con las auditorías de la seguridad informática a los SGBD Oracle y SQL Server.
- Caracterización de los sistemas informáticos existentes en el mundo que realizan auditorías de seguridad informática a SGBD Oracle y SQL Server.
- Análisis de sistemas informáticos basados en complementos.
- Realización de un estudio y selección de las herramientas, lenguajes de programación, arquitectura y patrones de diseño a utilizar.
- Análisis de los conceptos relacionados a la arquitectura basada en componentes.
- Identificación y descripción de los requerimientos funcionales de los complementos.
- Modelación del diseño de los complementos.
- Implementación de los requerimientos funcionales de los complementos.
- Realización de pruebas de caja blanca.
- Realización de pruebas de caja negra.

Para realizar las tareas de la investigación se emplearán los siguientes **métodos científicos**:

Métodos Teóricos:

- **Histórico lógico:** En la primera parte de la investigación se hará un estudio del estado del arte de la problemática y se analizarán las ventajas y desventajas de cada una de las herramientas utilizadas actualmente por el Departamento de Seguridad Informática de ETECSA para la realización de auditorías informáticas a SGBD Oracle y SQL Server.
- **Analítico – Sintético:** Se utilizará para captar y resumir varios documentos y procedimientos por los cuales se rige el Departamento de Seguridad Informática de ETECSA para hacer auditorías informáticas a SGDB Oracle y SQL Server. De ellos se extraerán las ideas fundamentales y se detallará la información necesaria para el modelado del negocio.
- **Modelación:** Se utilizará para abstraer la realidad mediante diagramas, permitiendo comprender y conocer la respuesta de los procesos sin tener que ejecutar los mismos en el mundo real. Por ejemplo los diagramas obtenidos de la ejecución de la metodología RUP en la etapa de diseño.

Métodos Empíricos:

- **Entrevista:** Se utilizará la entrevista como una conversación formal con los clientes, para obtener información sobre el problema en cuestión. Su uso es un medio para el conocimiento cualitativo de las características particulares de un proceso y puede influir en el posterior análisis y diseño del producto de software.

El presente extenso, está estructurado en cuatro capítulos, a continuación se muestra una breve descripción de cada uno de ellos:

Capítulo 1: “Fundamentación Teórica”, contiene una base teórica para entender el problema que se plantea, un estado del arte del tema a nivel nacional e internacional y una descripción de los conceptos fundamentales así como las tendencias, técnicas, tecnologías, metodologías y software usados.

Capítulo 2: “Características del Sistema”, contiene el análisis de la situación problemática, la construcción del modelo de negocio; la especificación de los requisitos que constituirán las bases de la propuesta del sistema, así como la elaboración de los diagramas de casos de uso del sistema y prototipos no funcionales de interfaces.

Capítulo 3: “Diseño del Sistema”, representa la base de la futura implementación del sistema; donde se realiza la elaboración de diagramas de clases del Diseño, la descripción de la arquitectura del sistema y de los patrones de diseño a utilizar.

Capítulo 4: “Implementación y Prueba”, muestra cómo será implementado el sistema en término de componentes, así como los tipos de pruebas, métodos y técnicas utilizadas para realizar las mismas.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

El presente capítulo contiene una base teórica para entender el problema que se plantea, un estado del arte del tema a nivel nacional e internacional y una descripción de los conceptos fundamentales así como las herramientas, tecnologías y metodología usadas en la actualidad o en las que se apoya para la solución del problema en cuestión.

1.2 Conceptos fundamentales asociados al problema.

1.2.1 Auditoría

El concepto proviene del latín audire ("oír"), que hace referencia a la forma en que los primeros auditores cumplían con su función (escuchaban y juzgaban la verdad o falsedad de lo que era sometido a su verificación).

La auditoría es en resumen un examen crítico y sistemático que realiza una persona o grupo de personas independientes del sistema auditado.[3]

1.2.2 Auditoría informática

La auditoría informática es el proceso de recoger, agrupar y evaluar evidencias para determinar si un sistema de información trabaja adecuadamente con los parámetros establecidos, mantiene la integridad y seguridad de los datos llevando eficazmente los fines de la organización.[4]

[3] **Definicion.De.** Definición de Auditoría. *Definicion.de.* [En línea] 2010. [Citado el: 18 de Enero de 2012.] http://definicion.de/auditoria/.
[4] **Hoy Digital.** Tecnología ¿Qué es auditoría informática? *Hoy Digital.* [En línea] 5 de Enero de 2009. [Citado el: 18 de Enero de 2012.] http://www.hoy.com.do/investigacionTecnologiaQue-es-auditoria-informatica.

1.2.3 Auditoría en un SGBD

La auditoría a un SGBD está basada en políticas que indican qué, cómo y cuándo se va a auditar. Estas políticas están concebidas de manera que sea posible auditar, un objeto de la BD, en determinadas circunstancias, bajo el uso de una o varias operaciones definidas por el auditor.[5]

En fin, la auditoría de un SGBD, es una búsqueda exhaustiva de vulnerabilidades realizada directamente en el gestor, ejecutada por personal capacitado en el tema y capaz de valorar y emitir una evaluación al concluir la auditoría.

1.2.4 Complemento o Plugin

Un complemento es una aplicación informática que añade funcionalidades específicas a un programa principal.[6]

Estos plugins permiten que los desarrolladores colaboren con la aplicación principal extendiendo sus funciones y permiten reducir el tamaño de la aplicación.

En la presente investigación los plugins se relacionan estrechamente con los gestores de bases de datos. Es decir, cada gestor hace referencia a un plugins, el cual contendrá drivers, consultas y funcionalidades.

1.2.5 Script

Un script es un programa simple, un conjunto de órdenes contenidas en un archivo de texto plano, que facilita la automatización de tareas a través de la creación de pequeñas utilidades.[7]

Para el presente trabajo un script se definirá como un contenedor de consultas SQL que al ser ejecutado sobre un servidor de base de datos, construye un fichero que contendrá el resultado de las consultas

[5] **Delgado Picazo, Mario**. E-Archivo. Repositorio Institucional de la Universidad Carlos III. [En línea] [Citado el: 18 de Enero de 2012.] http://e-archivo.uc3m.es/bitstream/10016/6247/1/PFC_MDP_v1.0.pdf.
[6] **Saberia.Com.** Qué es un plugin. *Saberia.com.* [En línea] 2010. [Citado el: 18 de Enero de 2012.] http://www.saberia.com/2010/01/que-es-un-plugin/.
[7] **DefinicionABC.Com**. definicionabc.com. [En línea] [Citado el: 31 de Enero de 2012.] http://www.definicionabc.com/general/script.php.

ejecutadas. Las consultas contenidas en el script serán las encargadas de comprobar elementos, configuraciones o datos en la base de datos especificada.

1.2.6 El SGBD Oracle

Oracle es un SGBD con características objeto-relacionales desarrollado por Oracle Corporation. Sus características principales son las siguientes:

- Entorno cliente/servidor.
- Gestión de grandes bases de datos.
- Usuarios concurrentes.
- Alto rendimiento en transacciones.
- Sistemas de alta disponibilidad.
- Disponibilidad controlada de los datos de las aplicaciones.
- Adaptación a estándares de la industria, como SQL-92.
- Gestión de la seguridad.
- Autogestión de la integridad de los datos.
- Opción distribuida.
- Portabilidad.
- Compatibilidad.
- Conectabilidad.
- Replicación de entornos. [8]

1.2.7 El SGBD SQL Server

Microsoft SQL Server es un Sistema de Gestión de Bases de Datos Relacionales (SGBDR) basado en el lenguaje Transact-SQL, y específicamente en Sybase IQ, fabricado por Microsoft, capaz de poner a disposición de muchos usuarios grandes cantidades de datos de manera simultánea. Constituye la

[8] **Velasco, Roberto Hernando.** El SGBDR Oracle. rhernando.net. [En línea] [Citado el: 08 de 12 de 2011.] http://www.rhernando.net/modules/tutorials/doc/bd/oracle.html.

alternativa de Microsoft a otros potentes sistemas gestores de bases de datos como son Oracle, Sybase ASE, PostgreSQL, Interbase, Firebird o MySQL.[9]

Microsoft SQL Server ofrece las siguientes características:

- Facilidad de instalación, distribución y utilización.
- Posee una gran variedad de herramientas administrativas y de desarrollo que permiten mejorar la capacidad de instalar, distribuir, administrar y utilizar SQL Server.
- Almacenamiento de datos.
- Incluye herramientas para extraer y analizar datos resumidos para el Proceso Analítico en Línea (OLAP). SQL Server incluye también herramientas para diseñar gráficamente las bases de datos y analizar los datos mediante preguntas en lenguaje normal.
- Se integra con el correo electrónico, internet y Windows, permitiendo una comunicación local.[10]

1.3 Análisis de soluciones existentes

Varias son las herramientas que existen a nivel mundial que realizan auditorías de seguridad informática de base de datos. La correcta puesta en marcha de cada una de ellas permitirá determinar las vulnerabilidades y brechas que posean los SGBD. A continuación se exponen las principales características de estas herramientas.

1.3.1 Oracle® Audit Vault

Es una solución completa para la auditoría de base de datos y control de actividades que ofrece características de creación de informes y alertas. Automatiza el proceso de auditoría de la base de datos con nuevas características que incluyen:

- Programación de informes, notificación, autenticación que pueden ayudar a las empresas a bajar el costo que implica cumplir con las obligaciones de privacidad y protección de datos internos y externos.
- Informes de permisos y privilegios con copias actualizadas de los usuarios, privilegios y perfiles de Oracle Database, que permiten a los auditores controlar los cambios al acceso de la base de datos.

[9] **Ramos Ortega, Diego Martin.** Microsoft SQL Server - Monografías.com. monografias.com. [En línea] [Citado el: 19 de Enero de 2012.] http://www.monografias.com/trabajos73/microsoft-sql-server/microsoft-sql-server.shtml.

[10] **Raráz Tinoco, Jorge** Luis. slideshare.net. [En línea] [Citado el: 19 de Enero de 2012.] http://www.slideshare.net/jorg_leoxd/comparacion-entre-my-sql-y-sql-server.

- Informes de cumplimiento reglamentario para cumplir con las disposiciones establecidas en la ley Sarbanes-Oxley (SOX), ley de Responsabilidad y Transferibilidad de Seguros Médicos (HIPAA) y Normas de Seguridad de Datos (DSS) de la Industria de Tarjetas de Pago (PCI) en lo referente al control y auditoría de la actividad de la base de datos.
- Limpieza automática de los datos del proceso de auditoría desde bases de datos Oracle y de otros proveedores, una vez que los datos de auditorías se han consolidado en forma segura en el repositorio Oracle Audit Vault. Esto ayuda a reducir los costos operativos de auditar la base de datos.

Para ayudar a las empresas a racionalizar aún más el control de la actividad y monitoreo de la base de datos y responder a las excepciones de seguridad con rapidez, la última versión de Oracle Audit Vault también admite alertas de correo electrónico.

Los datos de auditoría se consolidan automáticamente en un repositorio centralizado y sumamente seguro que se basa en la tecnología probada de data warehouse de Oracle. Estos datos se analizan en tiempo real teniendo en cuenta las políticas definidas por la empresa. Cualquier actividad no autorizada puede detectarse inmediatamente y notificar al personal para ayudar a garantizar la seguridad de la base de datos.

Incluye numerosos informes predefinidos que se pueden personalizar fácilmente para ayudar a las empresas a ahorrar tiempo y costos relativos a la creación de informes de cumplimiento reglamentario. Las poderosas capacidades de personalización de informes incluyen la posibilidad de aplicar filtros a los datos de auditoría, destacar filas con valores de condición, así como generar cuadros y gráficos. Los informes personalizados pueden guardarse y compartirse dentro de la empresa o con auditores externos.
Forma parte de conjunto de soluciones de seguridad de base de datos, entre las que se encuentran Oracle Advanced Security, Oracle Database Vault, Oracle Label Security, Oracle Data Masking y Oracle Total Recall. Estas soluciones ayudan a las empresas a protegerse de manera transparente contra la

filtración de datos y garantizar el cumplimiento reglamentario sin que sea necesario realizar cambios a las aplicaciones existentes.[11]

Oracle® Audit Vault es una herramienta privativa que, aunque es de las mejores en el mercado, no soluciona el problema debido a que, está diseñada según las métricas establecidas por Oracle y no según las métricas usadas en el Departamento de Seguridad Informática de ETECSA. También por la diversidad de versiones de Oracle que usa ETECSA, que van desde Oracle 8i hasta la versión actual, la 11g, teniendo en cuenta que Oracle Audit Vault sólo es compatible con versiones iguales o superiores a Oracle 9i Release 2.

1.3.2 Apex SQL Audit

Herramienta completa para la fabricación de bases de datos de auto-auditoría y para la integración de auditoría e informes de auditoría en sus aplicaciones. La herramienta es totalmente personalizable permitiendo modificar fácilmente las arquitecturas de auditoría por defecto, o incluso crear otras nuevas con la funcionalidad completa de la plantilla IDE. Apex SQL Audit es una herramienta de auditoría para empresas que necesitan auditar bases de datos Microsoft SQL Server.

Es una solución de auditoría ideal si el administrador es nuevo en SQL Server y necesita una herramienta fácil de usar o un usuario avanzado que se siente cómodo creando su arquitectura de auditoría desde cero. Permite realizar un seguimiento de las inserciones, actualizaciones y borrados por el usuario. Presenta una arquitectura totalmente personalizable y extensible que permite modificar las plantillas de disparadores. Incluye un módulo de informes, así como las exportaciones a Excel, CSV.[12]

Esta herramienta tiene como inconveniente que es privativa, su licencia tiene un costo de $484.03 dólares[13] y sólo está disponible para Windows.

[11] **CimaConsulting.Com.Mx.** Oracle Audit Vault. cimaconsulting.com.mx. [En línea] [Citado el: 18 de Enero de 2012.] http://www.cimaconsulting.com.mx/mportal/index.php?option=com_content&task=view&id=323&Itemid=125.
[12] **Auditoria.Com.Mx.** AUDITORIA Y SEGURIDAD INFORMATICA. [En línea] [Citado el: 19 de Enero de 2012.] http://www.auditoria.com.mx/productos/analyze/apex/apexsql.htm.
[13] **Componet Source. Componet Source.** [En línea] [Citado el: 19 de Enero de 2012.] http://www.componentsource.com/products/apexsql-audit/index-es.html.

1.4 Soluciones usadas en Cuba

En Cuba, particularmente en ETECSA, las auditorías a los SGBD Oracle y SQL Server se realizan a partir de scripts que contienen consultas SQL, usando una guía de buenas prácticas, que contiene los parámetros a evaluar, los valores sugeridos por mejores prácticas, el riesgo, así como dónde buscar la información para evaluar dichos parámetros. Esta guía conduce todo el proceso de auditoría y es utilizada para interpretar los resultados que se obtienen al ejecutar las consultas contenidas en los scripts. Además, para plasmar los resultados de la auditoría se utiliza una matriz de resultado. El proceso en su mayoría, se realiza de forma manual, lo que da paso a posibles errores humanos y disminuir la calidad del mismo.

De manera general, las herramientas antes mencionadas no satisfacen la necesidad existente en el Departamento de Seguridad Informática de ETECSA, pues son herramientas privativas, con licencias muy caras, no están diseñadas según las métricas usadas por el departamento, no abarcan todas las versiones de Oracle y SQL Server que están en explotación por parte de ETECSA y no comprenden todos los aspectos que incluye una auditoría a estos SGBD en el Departamento de Seguridad Informática de ETECSA. Por lo que se acordó con el Departamento de Seguridad Informática de ETECSA, desarrollar complementos que permitan evaluar los resultados obtenidos de auditorías a las diferentes versiones de Oracle y SQL Server en explotación, que cumplan con las métricas utilizadas en el departamento, que engloben todos los aspectos comprendidos en una auditoría a los SGBD Oracle y SQL Server y aumenten la calidad de ejecución de la misma.

1.5 Herramientas, Tecnologías y Metodología

Para el desarrollo de estos complementos se utilizarán un conjunto de herramientas, tecnologías y metodología que permitirán que el mismo quede con la mayor calidad posible.

1.5.1 Metodología de desarrollo RUP

Debido a las dificultades que afrontan los desarrolladores para coordinar las múltiples cadenas de trabajo de un gran proyecto de software surge la necesidad de hallar una forma de trabajar coordinadamente, de tener un proceso que integre las múltiples facetas del desarrollo, un método común, un proceso que:

- Proporcione una guía para ordenar las actividades de un equipo.
- Dirija las tareas de cada desarrollador por separado y del equipo como un todo.

- Especifique los artefactos que deben desarrollarse.
- Ofrezca criterios para el control y la medición de los productos y actividades del proyecto.[14]

Rational Unified Process (RUP) es un proceso de ingeniería de software que proporciona un enfoque disciplinado para la asignación de tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es garantizar la producción de software de alta calidad que satisfaga las necesidades de sus usuarios finales dentro de un horario predecible y presupuesto.[15] Es también, un marco de proceso que puede ser adaptado y ampliado para satisfacer las necesidades de una organización.

RUP está basado en componentes interconectados a través de interfaces y utiliza UML (Unified Modeling Language) como lenguaje de modelado de procesos, es dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. Captura muchas de las mejores prácticas en el desarrollo del software moderno en una forma que es apropiada para una amplia gama de proyectos y organizaciones. Incluye artefactos como: el diagrama de casos de uso del negocio, diagrama de clases del diseño, diagrama de componentes, el código fuente, entre otros y roles.

Se seleccionó RUP como metodología de desarrollo debido a que es una metodología flexible que puede ser adaptada según las necesidades del proyecto, además, genera abundante documentación, lo cual resulta de vital importancia, teniendo en cuenta que el equipo de desarrollo es inestable por estar conformado en su mayoría por estudiantes. Esta documentación serviría de referencia para los nuevos miembros que ingresen al proyecto.

1.5.2 Lenguajes de Modelado

UML

El Lenguaje Unificado de Modelado (UML) es un lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Captura decisiones y conocimiento sobre los sistemas que se deben construir. Se usa para entender, diseñar, configurar,

[14] **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** El Proceso Unificado de Software. 1era en Español. s.l. : Pearson Educación, S.A., 2000.

[15] **Gestionrrhhusm.Blogspot.Com.** [En línea] [Citado el: 18 de Enero de 2012.] http://gestionrrhhusm.blogspot.com/2011/05/modelo-rup-rational-unified-process-o.html.

mantener, y controlar la información sobre tales sistemas. Pretende unificar la experiencia pasada sobre técnicas de modelado e incorporar las mejores prácticas actuales en un acercamiento estándar. Incluye conceptos semánticos, notación, y principios generales. Permite realizar diagramas estáticos, dinámicos, de entorno y organizativos.

Está pensado para ser utilizado en herramientas interactivas de modelado visual que tengan generadores de código así como generadores de informes. Pretende dar apoyo a la mayoría de los procesos de desarrollo orientados a objetos.[16]

Aunque es el lenguaje de modelado por naturaleza de la metodología seleccionada, se seleccionó por ser un lenguaje de modelado que puede ser usado independientemente de la metodología de desarrollo y el lenguaje de programación a usar en el desarrollo de los complementos. Además de que permite modelar el flujo de desarrollo de los complementos mediante los diagramas de casos de uso, diagramas de clases, diagramas de actividades y diagramas de componentes entre otros.

BPMN

Notación de Modelado de Procesos del Negocio (BPMN) es una notación gráfica que describe la lógica de los pasos de un proceso de negocio. Esta notación ha sido especialmente diseñada para coordinar la secuencia de los procesos y los mensajes que fluyen entre los participantes de las diferentes actividades. BPMN proporciona un lenguaje común para que las partes involucradas puedan comunicar los procesos de forma clara, completa y eficiente.[17]

Se seleccionó BPMN debido a que es un estándar internacional de modelado de procesos aceptado por la comunidad, independiente de cualquier metodología de modelado de procesos, disminuye la brecha entre los procesos del negocio y la implementación de estos y permite modelar los procesos de una manera unificada y estandarizada, permitiendo al personal de una organización, un mejor entendimiento de los mismos.

[16] **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** El Lenguaje Unificado de Modelado. Manual de Referencia. 1998. págs. 3-4.
[17] **Bizagi.**BPMN Bussines Process Modeling Notation.Bizagi. [En línea] 2011. [Citado el: 17 de Enero de 2012.] www.bizagi.com/docs/BPMNbyExampleSPA.pdf.

1.5.3 Herramienta CASE. Visual Paradigm-UML

Es una herramienta diseñada para usuarios como: Ingenieros de Software, Analistas de Sistemas, Arquitectos de Sistemas y otros que estén interesados en el diseño de software orientado a objetos. Incluye un soporte completo para BPMN.

Se caracteriza por:

- Disponibilidad en múltiples plataformas (Windows, Linux).
- Diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Varios idiomas.
- Generación de código para Java.
- Fácil de instalar y actualizar.
- Compatibilidad entre ediciones.
- Diagramas de Procesos de Negocio - Proceso, Decisión, Actor de negocio, Documento.
- Editor de Detalles de Casos de Uso - Entorno todo-en-uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso.
- Distribución automática de diagramas - Reorganización de las figuras y conectores de los diagramas UML.
- Editor de figuras.[18]

Teniendo en cuenta las características antes mencionadas se decidió utilizar Visual Paradigm debido a que es independiente de la plataforma, se integra fácilmente al Eclipse permitiendo la ingeniería inversa.

[18] **EcuRed.** Visual Paradigm. EcuRed. [En línea] [Citado el: 18 de Enero de 2012.] http://www.ecured.cu/index.php/Visual_Paradigm.

1.5.4 El Lenguaje de Programación Java

Java es un lenguaje de programación orientado a objetos que se utiliza mayormente en entornos bastante complejos y basados en red. Proporciona una colección de clases para su uso en aplicaciones de red, que permiten abrir sockets y establecer y aceptar conexiones con servidores o clientes remotos, facilitando así la creación de aplicaciones distribuidas. Es interpretado, soporta aplicaciones que pueden ser ejecutadas en los más variados entornos de red como: Unix, Windows NT, Mac y estaciones de trabajo, sobre arquitecturas distintas y con sistemas operativos diversos. Java es un lenguaje originalmente desarrollado por un grupo de ingenieros de Sun Microsystems, utilizado por Netscape posteriormente como base para JavaScript. Si bien su uso se destaca en la Web, sirve para crear todo tipo de aplicaciones.[19]

Algunas características notables:

- Robusto.
- Gestiona la memoria automáticamente.
- No permite el uso de técnicas de programación inadecuadas.
- Multithreading.
- Cliente-servidor.
- Mecanismos de seguridad incorporados.
- Herramientas de documentación incorporadas.[20]

Java resultó escogido como lenguaje de programación debido a la gran cantidad de documentación que existe en Internet sobre su uso y debido a su portabilidad, característica que garantiza que los complementos a desarrollar sean multiplataforma, lo que puede ayudar al Departamento de Seguridad Informática de ETECSA en el proceso de migración hacia el software libre que está llevando a cabo el país.

[19] **Álvarez, Gonzalo Marañón.** Que es Java. iec.csic.es. [En línea] [Citado el: 18 de Enero de 2012.] http://www.iec.csic.es/criptonomicon/java/quesjava.html.
[20] **Pergamino Virtual. Definición de Java.** Pergamino Virtual. [En línea] 2011. [Citado el: 17 de Enero de 2012.] http://www.pergaminovirtual.com.ar/definicion/Java.html.

1.5.5 El Entorno de Desarrollo Integrado Eclipse

Eclipse es un entorno de desarrollo integrado de código abierto multiplataforma y extensible que permite a los desarrolladores crear rápidamente aplicaciones web, empresariales, de escritorio y móviles. Dispone de un editor de texto con resaltado de sintaxis. La compilación es en tiempo real. Tiene asistentes para creación de proyectos, clases, enumerativos, interfaces y refactorización. A través de plugins es posible añadir control de versiones con Subversion.

Eclipse es una plataforma gratuita, de código abierto, creada inicialmente por IBM y desarrollada en la actualidad por el Proyecto Eclipse. Es multiplataforma y totalmente extensible con módulos que aumentan su funcionalidad y que permiten que el entorno de desarrollo soporte otros lenguajes además de Java. La aplicación más conocida realizada con este entorno es el IDE Java llamado Java Development Toolkit (JDT) y el compilador incluido en Eclipse, ambas se utilizaron para desarrollar el propio Eclipse.

La definición que da el proyecto Eclipse acerca de su software es: "una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular". [21]

Fue seleccionado como IDE para el desarrollo de los complementos, debido a que es un producto de código libre y gratuito sin restricciones de uso. También porque brinda facilidades para el trabajo con el framework Spring y con XML. Además porque los complementos al no presentar interfaces gráficas no se hace necesario utilizar un IDE que facilite el trabajo con ellas.

1.5.6 Marco de Trabajo Spring

Spring es un framework de aplicación desarrollado por la compañía Interface 21, para aplicaciones escritas en el lenguaje de programación Java. Es una aplicación que no requiere de muchos recursos para su ejecución, además el framework completo puede ser distribuido en una archivo .jar de alrededor de 1mb. Es una aplicación de código libre, lo cual implica que no tiene ningún costo, ni es necesario el pago de una licencia para utilizarlo. Es un framework modular que cuenta con una arquitectura dividida en 7

[21] **Rbytes Reviews.** Rbytes reviews, Descargar Eclipse Classic v3.3.1.1. Rbytes reviews. [En línea] [Citado el: 18 de Enero de 2012.] http://rbytes.org/descargar/cat/programaci%C3%B3n/java--java-script/eclipse-classic/.

capas o módulos, lo cual permite tomar y ocupar únicamente las partes que interesen para el proyecto y juntarlas con gran libertad.[22]

Fue seleccionado debido a que puede emplearse en cualquier aplicación desarrollada en Java, brinda muchas libertades a los desarrolladores y ofrece soluciones bien documentadas. Además de permitir escribir un código más limpio, manejable y fácil de probar.

1.5.7 Analizador de SQL JSqlParser

JSqlParser es un analizador de SQL que examina la consulta SQL y genera una jerarquía de clases en Java. Las operaciones sobre las clases se realizan a través del patrón Visitante. JSqlParser permite la detección de errores de sintaxis e inyecciones SQL. Es un proyecto de software libre alojado en SourceForge bajo licencia GNU Lesser General Public License (LGPL).

Fue seleccionado para la validación de las consultas debido a que es una herramienta libre y sin restricciones de uso que facilita la validación de consultas SQL.[23]

1.6 Conclusiones

Se abordaron los conceptos fundamentales a utilizar durante el desarrollo de los complementos. Se presentó un estado del arte del tema a nivel nacional e internacional así como las herramientas, tecnologías y metodología en las que se apoya la presente investigación para la solución del problema, lográndose, de esta manera, una visión más clara del tema en cuestión.

[22] **Catarina.Udlap.Mx.** Capítulo 3: Spring un framework de aplicación. catarina.udlap.mx. [En línea] [Citado el: 19 de Enero de 2012.] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/sanchez_r_ma/capitulo3.pdf.

[23] **Sourceforge.net.** JSqlParser. *JSqlParser.* [En línea] [Citado el: 18 de Enero de 2012.] http://jsqlparser.sourceforge.net/.

CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA

2.1 Introducción

En el presente capítulo se abordarán las principales características del sistema que se va a implementar, se presentará el modelo de procesos del negocio y una descripción de cada uno de los procesos, se especificarán los requerimientos funcionales y no funcionales, se definirán los actores y casos de uso del sistema y se mostrará una descripción textual de estos últimos.

2.2 Modelo de Negocio

A continuación se muestra la descripción general del proceso de negocio propuesto y las mejoras que propone el negocio actual, indicando cómo se solucionarán los problemas que originaron la problemática.

2.2.1 Modelo de Procesos del Negocio

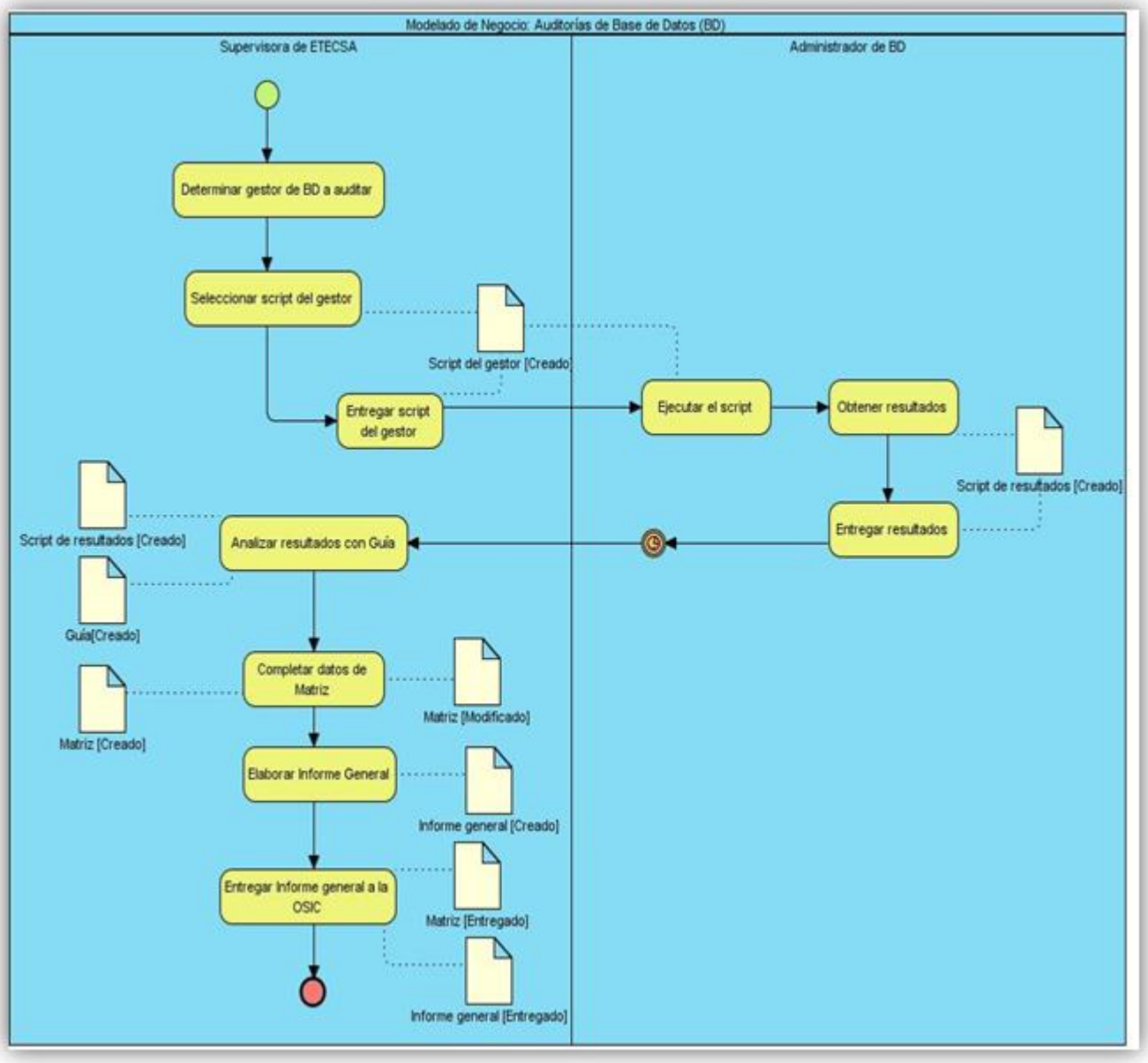

Figura 1. Modelo de Procesos del Negocio.

2.2.2 Descripción Textual de los Procesos del Negocio

Determinar el SGBD a auditar.
El negocio inicia cuando se determina por la planificación anual, realizar una supervisión a uno o varios gestores de BD instalados, por lo que se debe seleccionar el mismo y la versión correspondiente.
Responsable: Supervisora de ETECSA.
Entradas: No aplica.
Salidas: Gestor de BD seleccionado.

Seleccionar script del gestor.
Una vez seleccionado el gestor de BD se procede a seleccionar el script correspondiente para realizar la supervisión.
Responsable: Supervisora de ETECSA.
Entradas: Gestor de BD seleccionado.
Salidas: Script del gestor.

Entregar script del gestor.
El supervisor le entrega el script al administrador de la BD a supervisar.
Responsable: Supervisor de ETECSA.
Entradas: Script del gestor.
Salidas: Script del gestor.

Ejecutar el script.
El administrador ejecuta el script correspondiente a la BD a auditar.
Responsable: Administrador de BD.
Entradas: Script del gestor.
Salidas: No aplica.

Obtener resultados.
El administrador una vez que ejecuta el script, obtiene los resultados de la supervisión en un script de resultados.
Responsable: Administrador de BD.
Entradas: No aplica.

Salidas: Script de resultados.

Entregar resultados.
El administrador entrega el script de resultados para su posterior análisis.

Responsable: Administrador de BD.

Entradas: Script de resultados.

Salidas: Script de resultados.

Analizar resultados con Guía.
Se analiza el script de resultados con la Guía la cual permite interpretar los parámetros del script resultante.

Responsable: Supervisora de ETECSA.

Entradas: Script de resultados, Guía.

Salidas: No aplica.

Completar datos de Matriz.
Una vez que se analicen los datos del script resultante se procede a completar la Matriz la cual registra los resultados de la auditoría realizada al gestor.

Responsable: Supervisora de ETECSA.

Entradas: Matriz.

Salidas: Matriz.

Elaborar Informe General.

Se elabora un informe que sintetiza los resultados de la supervisión.

Responsable: Supervisora de ETECSA.

Entradas: Matriz.

Salidas: Informe general.

Entregar Informe General a la OSIC.

Se entrega el Informe general y Matriz a la OSIC para su revisión, archivo y distribución a las personas indicadas.

Responsable: Supervisora de ETECSA.

Entradas: Informe general, Matriz.

Salidas: Informe general, Matriz.

Los procesos que serán objeto de automatización son:

- Analizar resultados con la Guía.
- Completar datos de Matriz.

2.4 Propuesta de Solución

Para resolver la problemática descrita, se decide llevar a cabo el desarrollo de complementos que brinden una solución real a los problemas existentes en el Departamento de Seguridad Informática de ETECSA, especialmente en la evaluación de los resultados obtenidos de auditorías a SGBD Oracle y SQL Server.

Los complementos deben ofrecer la posibilidad de:

- Cargar sus configuraciones iniciales. Estas estarán contenidas en un fichero de configuración cuya estructura se muestra en la Figura 2. Este fichero contendrá el tipo de gestor, las versiones de ese gestor que soportan los complementos. Conjuntamente contendrá los indicadores que se miden del gestor, de los cuales se guarda el identificador, el nombre su estado (Activado/Desactivado) y una breve descripción. Asimismo las consultas asociadas a su indicador correspondiente, de las cuales se almacena el identificador, nombre, la sentencia SQL, su forma evaluativa (Intervalo/Lista/Variante), su estado (Activada/Desactivada), el impacto, las mejoras prácticas, el nivel de riesgo (Alto/Medio/Bajo), su peso (0.33/0.66/1.0), una breve descripción y las versiones del gestor para las que funciona. Dentro de las consultas además, en dependencia de su forma evaluativa se plasman sus intervalos, de los cuales se almacena el límite superior e inferior y si son cerrados o abiertos, sus variantes, su lista de valores predeterminados y sus evaluaciones.
- Cargar la información de los ficheros que contienen los resultados.
- Evaluar los resultados de auditorías a SGBD Oracle y SQL Server.
- Conformar la matriz de resultados con los valores encontrados, los valores sugeridos, los pesos correspondientes y la evaluación ofrecida por el sistema.

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<gestor Gestor="" Versiones="version1;version2;versionN">
    <indicador id="1" Nombre="" Estado="" Descripcion="">
        <consulta id="" Nombre="" SQL="" Forma_de_evaluacion=""
            Estado="" Impacto="" Mejores_practicas="" Nivel_de_riesgo=""
            Peso="" Descripcion="" Versiones_Soportadas="version1;version2">
            <Lista Nivel_Coincidencia="" Nivel_NO_Coincidencia="">
                <Valor Valor="" />
                <Valor Valor="" />
            </Lista>
        </consulta>
    </indicador>
</gestor>
```

Figura 2. Estructura del Fichero de Configuración.

La evaluación de los resultados se realizará a través de las formas evaluativas. Estas son clases encargadas de emitir una evaluación a partir de un valor o una lista de valores.

Existen tres tipos de formas evaluativas:

- Por Intervalos (FEBuscarEnIntervalos): Presentará en su estructura, tres listas de intervalos predeterminados. Una lista para los intervalos que reciben una evaluación de bien, otra para los intervalos que reciben una evaluación de regular y otra para los intervalos que reciben una evaluación de mal. Esta forma evaluativa solo acepta un valor a la vez. Si el valor pasado a evaluar está en algún intervalo dentro de alguna lista, el valor recibirá la evaluación que posee la lista correspondiente. Por ejemplo si el valor a evaluar se encuentra en un intervalo perteneciente a la lista de intervalos que reciben una evaluación de mal, el valor recibirá una evaluación de mal. Un intervalo es una clase que contiene la información de los límites superior e inferior (valores numéricos) y si éstos son cerrados o abiertos (valores booleanos).
- Por Lista (FEBuscarEnLista): Presentará en su estructura, una lista de valores predeterminados (cadenas de caracteres), además de una evaluación predeterminada para si el valor a evaluar se encuentra en la lista y otra en caso contrario. Esta forma evaluativa acepta un valor o una lista de valores.

- Por Variantes (FEBuscarEnVariantes): Su estructura y funcionamiento es similar a FeBuscarEnIntervalos pero con variantes. Las variantes son cadenas de caracteres.

El proceso de evaluación tiene como entrada, un fichero de resultados, que contendrá el gestor, la versión del gestor auditado, los indicadores que se comprobaron, los resultados obtenidos, de los cuales se almacenó el identificador de la consulta que lo haya generado y los valores que se obtuvieron.

```
<?xml version="1.0" encoding="UTF-8"?>
<gestor nombre="" version="">
    <indicador id="">
        <resultados idconsulta="">
            <value valor="" />
            <value valor="" />
        </resultados>
        <resultados>
            <value valor="" />
            <value valor="" />
        </resultados>
    </indicador>
</gestor>
```

Figura 3. Estructura del Fichero de Resultados.

El final del proceso de evaluación tiene lugar cuando, una vez evaluados los resultados con las formas evaluativas, dichas evaluaciones son almacenadas en una matriz de resultados y es enviada al SASGBD.

En la matriz se recogen además de las evaluaciones, el nombre y descripción de los parámetros, los valores encontrados, el impacto y las mejoras sugeridas.

Nombre del Parámetro y Descripción	Valor Encontrado	Impacto	Mejoras Sugeridas

Tabla 1. Formato de la Matriz de Resultados.

Debido a que toda la información que se almacena en los servidores de bases de datos de ETECSA, constituye información oficial, los complementos deberán validar las consultas SQL antes de que se le asocien para evitar acciones no deseadas.

2.5 Relación de los Requerimientos

El Glosario Estándar de Terminología de Ingeniería de Software del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), define como requerimiento a una condición o capacidad que tiene que ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente.[24] Los requisitos se clasifican en: funcionales y no funcionales.

2.5.1 Requerimientos Funcionales

Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir. De forma general estos requisitos indican lo que debe hacer el sistema. Los requerimientos funcionales que se identificaron para el desarrollo de los complementos fueron los siguientes:

- RF1 Cargar Configuración Inicial.

  1.1 Obtener las versiones del gestor.

  1.2 Obtener la información de los indicadores.

  - ✓ Descripción.
  - ✓ Estado.
  - ✓ Nombre.
  - ✓ Identificador.

  1.3 Obtener la información de las consultas.

  - ✓ Descripción.
  - ✓ Estado.
  - ✓ Forma de Evaluación.
  - ✓ Impacto.
  - ✓ Mejoras Prácticas.
  - ✓ Nivel de Riesgo.
  - ✓ Nombre.

[24] Thayer, R.H. y Dorfman, M. Software Requeriments Engineering. 2nd. s.l.: IEEE Computer Society Press, 1997.

- ✓ Peso.
- ✓ Sentencia SQL.
- ✓ Versiones Soportadas.
- ✓ Identificador.

- RF2 Cargar Información de los Resultados.

  2.1 Comprobar datos del complemento.
  - ✓ Gestor.
  - ✓ Versión.

  2.2 Organizar datos del fichero:
  - ✓ Indicadores.
  - ✓ Consultas.
  - ✓ Valores encontrados.
- RF3 Realizar Evaluación de los Resultados por Indicadores.

  2.1 Realizar evaluación de los resultados del indicador “Usuarios Genéricos y Usuarios por Defecto del Sistema”.

  2.2 Realizar evaluación de los resultados del indicador “Login de las cuentas de usuarios”.

  2.3 Realizar evaluación de los resultados del indicador “Configuración de Contraseñas de las cuentas de usuarios”.

  2.4 Realizar evaluación de los resultados del indicador “Configuración de Auditoría”.

  2.5 Realizar evaluación de los resultados del indicador “Roles y privilegios de las cuentas de Usuarios”.

  2.6 Realizar evaluación de los resultados del indicador “Privilegios otorgados a cuentas de usuarios”.

  2.7 Realizar evaluación de los resultados del indicador “Análisis del archivo init.ora”.

  2.8 Realizar evaluación de los resultados del indicador “Configuración de Cuentas de usuarios”.

  2.9 Realizar evaluación de los resultados del indicador “Configuración del servidor de bases de datos”.

  2.10 Realizar evaluación de los resultados del indicador “Accesos de usuarios y permisos sobre objetos de la base de datos”.

- RF3 Conformar Matriz de Resultado.

  3.1 Llenar la matriz de resultado con los datos encontrados y sus respectivas evaluaciones.
  - ✓ Identificador.
  - ✓ Consulta.
  - ✓ Nivel de Riesgo.
  - ✓ Valores Encontrados.
  - ✓ Mejoras Sugeridas.
- RF4 Validar la consulta insertada a partir de las siguientes especificaciones.
  - Sintaxis de Consulta.
  - Sentencia de tipo SQL.

2.5.2 Requerimientos no Funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Estas propiedades constituyen las características que hacen al producto atractivo, usable, rápido y confiable. En muchos casos dichos requisitos son fundamentales en el éxito del producto. Los requerimientos no funcionales para el desarrollo de los complementos para auditar SGBD Oracle y SQL Server son:

- **Rendimiento.**
  - ✓ Los complementos deben estar concebidos para el consumo mínimo de recursos.
  - ✓ Deben tener un rápido procesamiento de los datos.
- **Portabilidad.**
  - ✓ Los complementos deben ser multiplataforma.
- **Software.**
  - ✓ Los complementos necesitan la Máquina Virtual de Java (JVM) en su versión 6 o superior y la aplicación SASGBD.
- **Seguridad.**
  - ✓ Disponibilidad: La aplicación estará disponible en todo momento para aquellas personas con acceso a la información, y los mecanismos utilizados para lograr la seguridad no serán un obstáculo a los usuarios para obtener los datos deseados en el momento que lo requieran.

- **Hardware.**
  - ✓ Se requiere al menos 512Mb de RAM.

2.6 Modelo de Casos de Uso del Sistema

El Modelo de Casos de Uso del Sistema permite que los desarrolladores de software y los clientes lleguen a un acuerdo sobre los requisitos, es decir sobre las condiciones y posibilidades que debe cumplir el sistema. Además proporciona la entrada fundamental para el análisis, diseño y las pruebas.[25]

2.6.1 Definición de los Actores del Sistema

Actores del sistema	Justificación
SASGBD.	Este actor representa el rol del Módulo de Bases de Datos, que es el que interactúa con los complementos.

Tabla 2. Definición de los Actores del Sistema.

2.6.2 Diagrama de Casos de Usos del Sistema

Un diagrama de casos de uso del sistema (DCUS) representa gráficamente a los procesos y su interacción con los actores. Los actores del sistema representan el rol que juega una o varias personas, un equipo o un sistema automatizado. No son parte del sistema, pero si pueden intercambiar información con él; así como pueden ser recipiente pasivo de información.[26] A continuación se muestra el DCUS de los complementos.

[25] **Jacobson, Ivar, Booch, Grady and Rumbaugh, James**. El Proceso Unificado de Desarrollo de Software. 1era en Español. s.l. : Pearson Educación, S.A., 2000.
[26] **Pimentel, Annia and Hernández, Antonio.** Sistema Informático para la Gestión de Auditoría y Control (SIGAC). Módulo de Planificación. 2009. Tesis de pregrado.

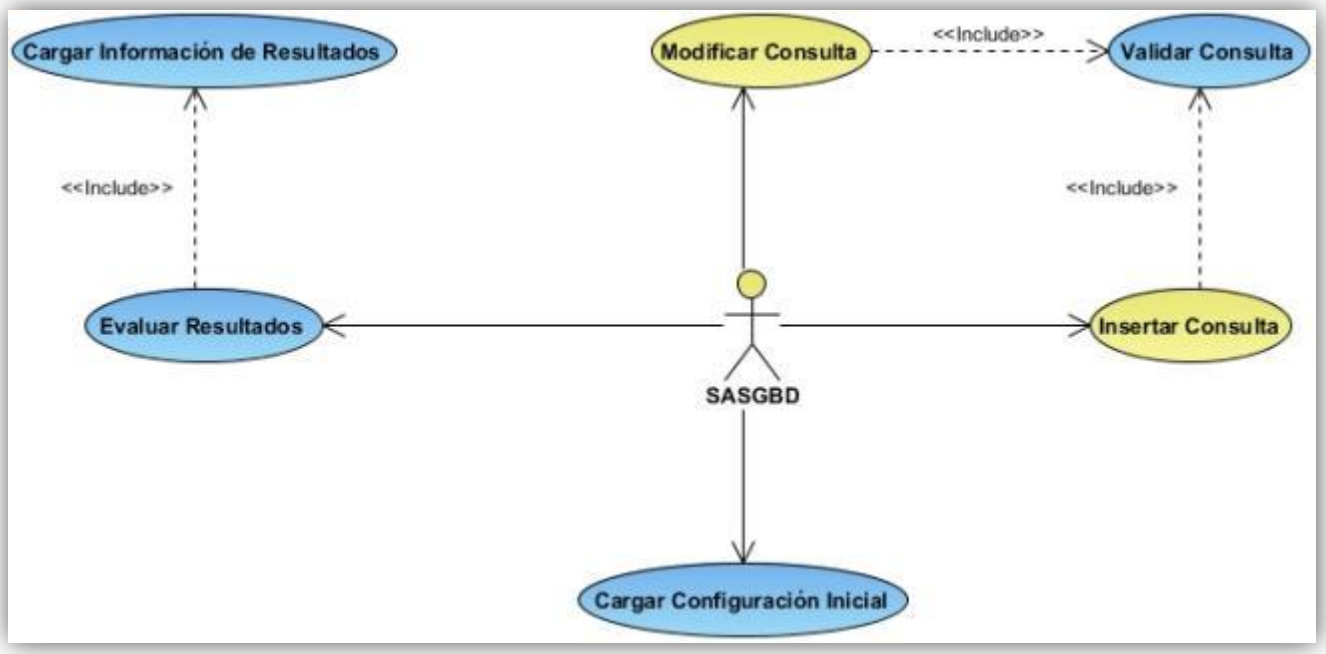

Figura 4. Diagrama de Casos de Uso del Sistema.

Nota: El caso de uso de color amarillo es un caso de uso que pertenece al SASGBD.

2.6.3 Descripción textual de los Casos de Uso

El objetivo principal de detallar los casos de uso de un sistema es describir su flujo de sucesos en detalle, incluyendo cómo inicia, termina e interactúan con los actores.[27]

Evaluar Resultados

Objetivo	Evaluar Resultados.
Actores	Módulo de Bases de Datos.
Resumen	El caso de uso se inicia cuando el Módulo de Bases de Datos solicita realizar la evaluación de los resultados. El CU termina cuando todos los

[27] **Jacobson, Ivar, Booch, Grady and Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. 1era en Español. s.l. : Pearson Educación, S.A., 2000.

	resultados fueron evaluados.
Complejidad	Media.
Prioridad	Crítico.
Precondiciones	Para realizar la evaluación debe haberse cargado con anterioridad el fichero de resultados.
Postcondiciones	Se realizó la evaluación de los resultados.
Flujo de eventos	
Flujo básico Evaluar Resultados	

	Actor	**Sistema**
1.	Solicita realizar la evaluación de los resultados obtenidos.	2. Carga la información del fichero de resultados. Ver CU Cargar Información de Resultados. 3. Crea un diccionario que contendrá los siguientes datos: • Id de la consulta • Forma evaluativa correspondiente. 4. Obtiene por cada resultado los siguientes datos: • Id de la consulta • Forma evaluativa correspondiente. 5. Evalúa cada resultado pasándole los valores encontrados a la forma evaluativa correspondiente y se obtiene la evaluación propuesta. 6. Almacena en una lista, de cada resultado, el id de la consulta

	asociada al resultado y la evaluación propuesta por el sistema. 7. Devuelve una matriz con los siguientes datos: ✓ Consultas. ✓ Evaluaciones propuestas. ✓ Valores encontrados. ✓ Mejoras sugeridas.
8. Carga la matriz. 9. Muestra la matriz de resultados.	

Relaciones	**CU Incluidos**	Ver caso de uso “Cargar Información de Resultados”.
	CU Extendidos	

Tabla 3. Descripción textual del CU "Evaluar Resultados".

Validar Consulta

Objetivo	Validar Consulta.
Actores	Módulo de Bases de Datos.
Resumen	El caso de uso se inicia cuando el Módulo de Bases de Datos solicita insertar o modificar una consulta, acciones para las cuales es necesario validar que sea una consulta válida. El CU termina cuando la consulta es validada.
Complejidad	Media.
Prioridad	Crítico.
Precondiciones	Para validar una consulta se debe haber inicializado del CU Gestionar Consultas, las secciones “Insertar Consulta” o “Modificar Consulta”.

Postcondiciones	Se validó la consulta.	
Flujo de eventos		
Flujo básico Validar Consulta		
	Actor	**Sistema**
1. Solicita validar la consulta.		2. Obtiene la cadena de la consulta. 3. Accede a la librería JSqlParser. 4. Parsea la consulta para comprobar que sea sintácticamente correcta. 5. Verifica que la consulta sea de tipo SELECT. 6. Devuelve Verdadero o un mensaje con el error encontrado.
7. Muestra un mensaje con el resultado recibido.		

Tabla 4. Descripción textual del CU "Validar Consulta".

2.6 Conclusiones

En este capítulo, a partir del análisis de los procesos del negocio comenzó a desarrollarse la propuesta de solución de los complementos. Se definieron los requisitos funcionales y no funcionales que debe tener el mismo, así como, se representaron cada una de estas funcionalidades a través de un Diagrama de Casos de Uso. Finalmente se describieron paso a paso las interacciones de los actores del sistema con cada uno de los casos de uso.

CAPÍTULO 3. DISEÑO DEL SISTEMA

3.1 Introducción

En el presente capítulo se muestran los patrones de diseño y la arquitectura del sistema. También se presenta el diagrama de clases del diseño y los diagramas de secuencia por funcionalidad. De manera general se describe cómo el sistema será realizado a partir de las funcionalidades previstas y las restricciones impuestas, por lo que se indica con precisión lo que se va a programar.

3.2 Modelo de Diseño

El modelo de diseño es un modelo de objetos que describe la realización física de los casos de usos centrándose en cómo los requisitos funcionales y no funcionales, junto con otras restricciones relacionadas con el entorno de implementación, tienen impacto en el sistema a considerar. Además, el modelo de diseño sirve de abstracción de la implementación del sistema y es, de ese modo, utilizada como una entrada fundamental de las actividades de implementación.[28]

3.2.1 Arquitectura

La arquitectura es el conjunto de decisiones significativas sobre la organización de un sistema software. Incluye la selección de elementos estructurales y las interfaces mediante las que se conectan, la organización a gran escala de los elementos estructurales y la topología de su conexión, su comportamiento en las colaboraciones entre dichos elementos, los mecanismos importantes de que se dispone en el sistema y el estilo arquitectónico que guía su organización.[29]

[28] **García Peñalvo, Francisco José, Conde González, Miguel Ángel y Bravo Martín, Sergio .** OpenCourseWare de la Universidad de Salamanca. OpenCourseWare de la Universidad de Salamanca. [En línea] 16 de Octubre de 2008. [Citado el: 20 de Febrero de 2012.] http://ocw.usal.es/ensenanzas-tecnicas/ingenieria-del-software/contenidos/Tema6-DOO-1pp.pdf.

[29] **Rumbaugh, James, Jacobson, Ivar and Booch, Grady**. El Lenguaje Unificado de Modelado. Manual de Referencia. 1998.

En otras palabras una arquitectura es esencial para el éxito o el fracaso de un proyecto, proporciona una visión global del sistema a construir, es una vista estructural de alto nivel que define el estilo arquitectónico o la combinación de ellos para la solución de un problema.

Los estilos arquitectónicos son útiles para resumir estructuras de soluciones, definir los patrones posibles de las aplicaciones y evaluar arquitecturas alternativas con ventajas y desventajas conocidas ante diferentes conjuntos de requerimientos no funcionales. Para el desarrollo de los complementos se seleccionó el estilo arquitectónico Llamada y Retorno que permite construir una estructura de programa relativamente fácil de modificar y ajustar a escala.

Llamada y Retorno agrupa varios patrones arquitectónicos como el patrón Arquitectura en Capas, este permite la reutilización de las capas, facilita la estandarización y la modularización del software y elimina las dependencias entre las capas, lo que quiere decir que los cambios aplicados sobre una capa, no afectan las demás. Razones por las que se utilizará en el desarrollo de los complementos.

Por lo expresado anteriormente, los complementos para Oracle y SQL Server, se integrarán al Módulo de Bases Datos, el cual presenta una arquitectura compuesta por las siguientes capas:

- GUI contendrá las interfaces gráficas mediante las cuales los usuarios harán uso de la aplicación.
- Entorno es la encargada de establecer la comunicación entre las capas GUI y Negocio.
- Negocio contendrá toda la lógica de negocio.
- AD manejará todo lo relacionado con el acceso a los datos.
- Comunicador encargada de establecer la comunicación con los complementos.
- Plugins es una capa vertical que es donde se encuentran cada uno de los complementos, que serán ejecutados por la aplicación.

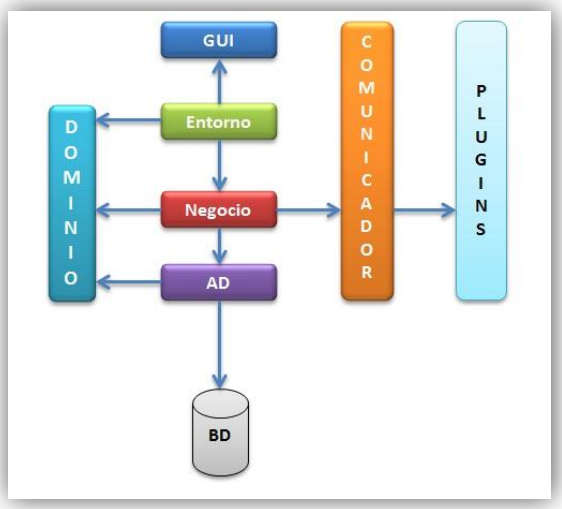

Figura 5. Estructura General del Módulo de Bases de Datos.

Dichos plugins serán capaces de analizar los resultados obtenidos. Los mismos en su estructura, cuentan con una clase principal, que es la encargada de heredar e implementar las funciones de la interfaz de comportamiento definida en la capa Comunicador. Esta es la que conoce y controla toda la lógica del negocio de los complementos contenida en la capa Negocio y esta última se relaciona con la capa Recursos, que es la que contiene las clases auxiliares, las clases utilizadas del framework Spring y las de la librería JSqlParser que se utilizan en el presente trabajo.

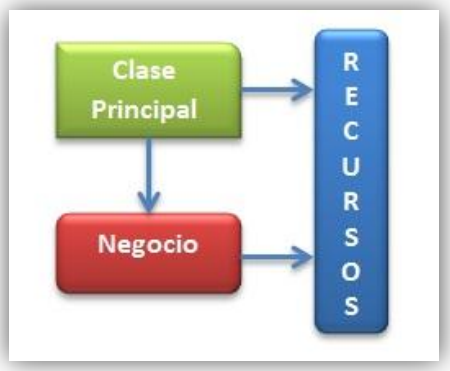

Figura 6. Estructura General de los Complementos.

3.2.2 Patrones de Diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema de diseño general en un contexto particular. Identifica: clases, instancias, roles, colaboraciones y la distribución de responsabilidades. Seguidamente se describen los patrones de diseño a usar en la implementación de los complementos.

- **Patrones de Asignación de Responsabilidades (GRASP)**
  - ✓ **Alta cohesión:** Es el responsable de asignar una responsabilidad de forma tal que la cohesión siga siendo alta, este patrón caracteriza a las clases que están estrechamente relacionadas y consiste en colaborar con otros objetos para compartir el esfuerzo si la tarea a realizar es grande.
  - ✓ **Experto:** Permite a los objetos valerse de su propia información para hacer lo que se les pide, favorece la existencia de mínimas relaciones entre las clases, lo que permite contar con un sistema sólido y fácil de mantener. Este patrón se utilizó en las clases principales de ambos complementos. Estas clases son las que conocen toda la lógica del negocio de su complemento.
  - ✓ **Bajo acoplamiento:** Garantiza que exista una alta reutilización entre las funcionalidades de las clases y una escasa dependencia, lo que contribuye al mantenimiento de las mismas. El uso de los patrones Experto y Creador favorecen al bajo acoplamiento entre las clases del sistema. Este patrón se tiene en cuenta por la importancia que tiene realizar un diseño de clases independientes que puedan soportar los cambios de una manera fácil y permitan la reutilización.
- **Patrones Gang of Four (GOF)**
  - ✓ **Fábrica Abstracta:** Proporciona una interfaz para crear familias de objetos relacionados o dependientes sin especificar su clase concreta. El mismo permite configurar en tiempo de ejecución un sistema con una familia u otra de objetos. Además garantiza que un conjunto de clases se usen a la vez.
  - ✓ **Solitario:** Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.

- ✓ **Visitante:** Representa una operación que está pensada para ser aplicada sobre los elementos de una estructura de objetos, permitiendo así definir y añadir un nuevo comportamiento sin necesidad de cambiar las clases de los elementos de la estructura de objetos. Este patrón se utilizó en la clase Parser para la validación de las consultas.

El uso del marco de trabajo Spring, a través de la Inversión del Control o Inyección de Dependencias como también se conoce, garantiza la utilización de los patrones GRASP Alta Cohesión y Bajo Acoplamiento y de los GOF Fábrica Abstracta y Solitario usando la clase ClassPathXmlApplicationContext que implementa la interfaz ApplicationContext y que a su vez usa la clase BeanFactory para la creación de objetos.

3.3 Diagramas de Clases del Diseño (DCD)

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación. Normalmente contienen la siguiente información:

- Clases, asociaciones y atributos.
- Interfaces, con sus operaciones y constantes.
- Métodos.
- Información sobre los tipos de atributos.
- Navegabilidad.
- Dependencias.

A diferencia del modelo conceptual, un diagrama de este tipo contiene las definiciones de las entidades del software en vez de conceptos del mundo real.[30] La siguiente figura muestra el diagrama de paquetes de los complementos.

[30] **Instituto de Ingeniería Eléctrica (IIE).** Sitio Web Instituto de Ingeniería Eléctrica de la Universidad de la República. [En línea] [Citado el: 3 de Marzo de 2012.] http://iie.fing.edu.uy.

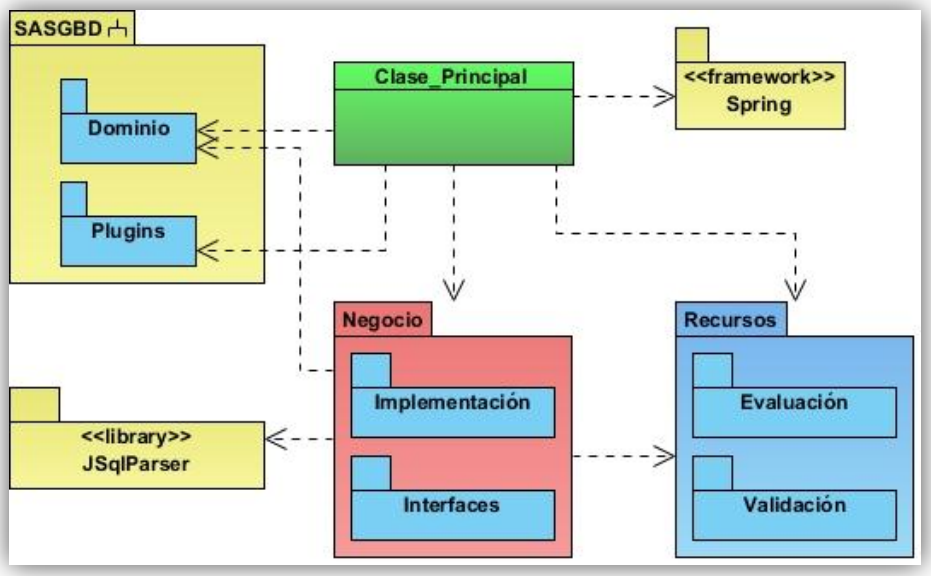

Figura 7. Diagrama de Paquetes de los Complementos.

La figura 5 muestra las relaciones existentes entre los diferentes paquetes que conforman los complementos. La Clase Principal es quien conoce toda la lógica del negocio y está relacionada con el subsistema externo SASGBD, del cual se utilizan las clases del Dominio y la interfaz que han de implementar los complementos. También se relaciona con el framework Spring, para garantizar la inversión de control. Además se vincula con el paquete Negocio el cual contiene las principales funcionalidades de los complementos. Este último utiliza la librería JSqlParser con el cuál valida las consultas que se quieren insertar en la BD y al igual que la Clase Principal accede al paquete Recursos en el cual se encuentran las clases auxiliares que utilizan los complementos.

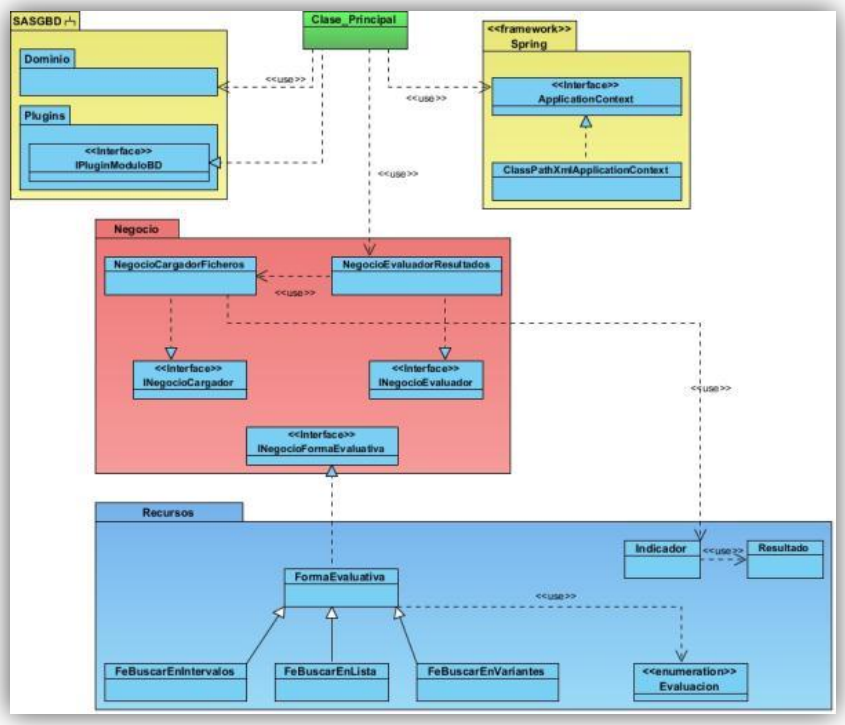

Figura 8. Diagrama de Clases del Diseño del Caso de Uso "Evaluar Resultados".

3.4 Diagramas de Interacción

Los diagramas de interacción representan una vista dinámica del sistema y constituyen la secuencia de acciones que ocurren cuando el actor comienza el caso de uso, así como los mensajes que se envían entre cada una de las clases. Los mismos se pueden clasificar en: diagramas de colaboración y diagramas de secuencia. Se utilizaron los diagramas de secuencia debido a que ilustran la interacción entre objetos y el orden secuencial en el que ocurren dichas interacciones, es decir cómo se comunican los objetos entre sí.

A continuación se muestran los diagramas de secuencia de los complementos.

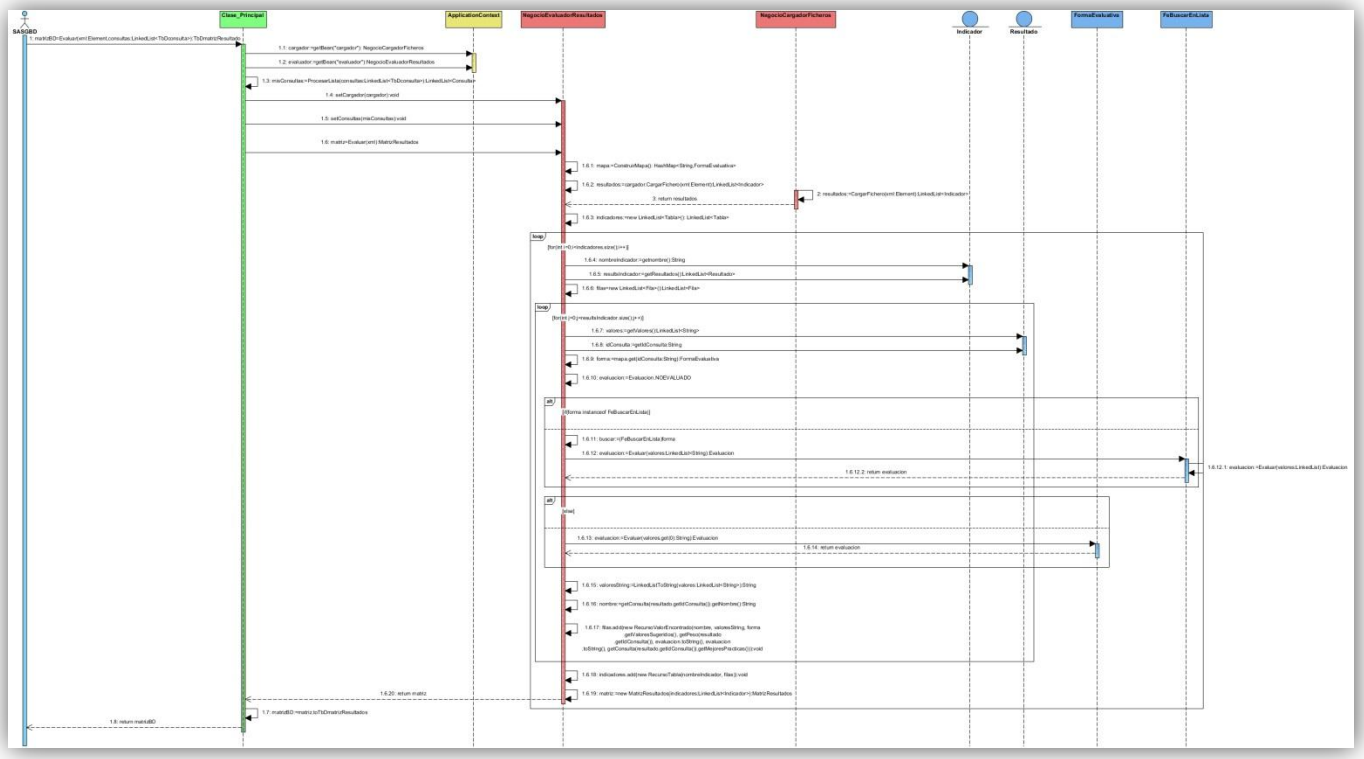

Figura 9. Diagrama de Secuencia del Caso de Uso "Evaluar Resultados".

3.5 Conclusiones

En este capítulo se ha presentado el diseño de la aplicación, que es la base en la implementación del sistema. Se abordaron los patrones de diseño, estilo y patrón arquitectónicos a utilizar en la implementación. Se mostraron además, los diagramas de clases del diseño y los diagramas de interacción por casa caso de uso, quienes constituyen una guía que puede ser fácilmente comprendida por los desarrolladores con el objetivo de implementar la aplicación que se ha diseñado.

CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBA

4.1 Introducción

En el presente capítulo se muestra el diagrama de componentes generado durante la etapa de implementación, así como la estrategia trazada para realizar los casos de prueba y los resultados alcanzados durante la etapa de prueba del sistema.

4.2 Implementación

En la implementación se comienza con el resultado del diseño y se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ficheros de código binario, ejecutables y similares. La mayor parte de la arquitectura del sistema es capturada durante el diseño, siendo el propósito principal de la implementación el desarrollar la arquitectura y el sistema como un todo.[31]

4.2.1 Diagramas de Componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y ejecutable. Los componentes representan todos los tipos de elementos software que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, etc.[32]

[31] **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** El Proceso Unificado de Desarrollo de Software. 1era en Español. s.l. : Pearson Educación, S.A., 2000.

[32] **Dsi.Uclm.Es.** dsi.uclm.es. [En línea] [Citado el: 19 de Mayo de 2012.] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.

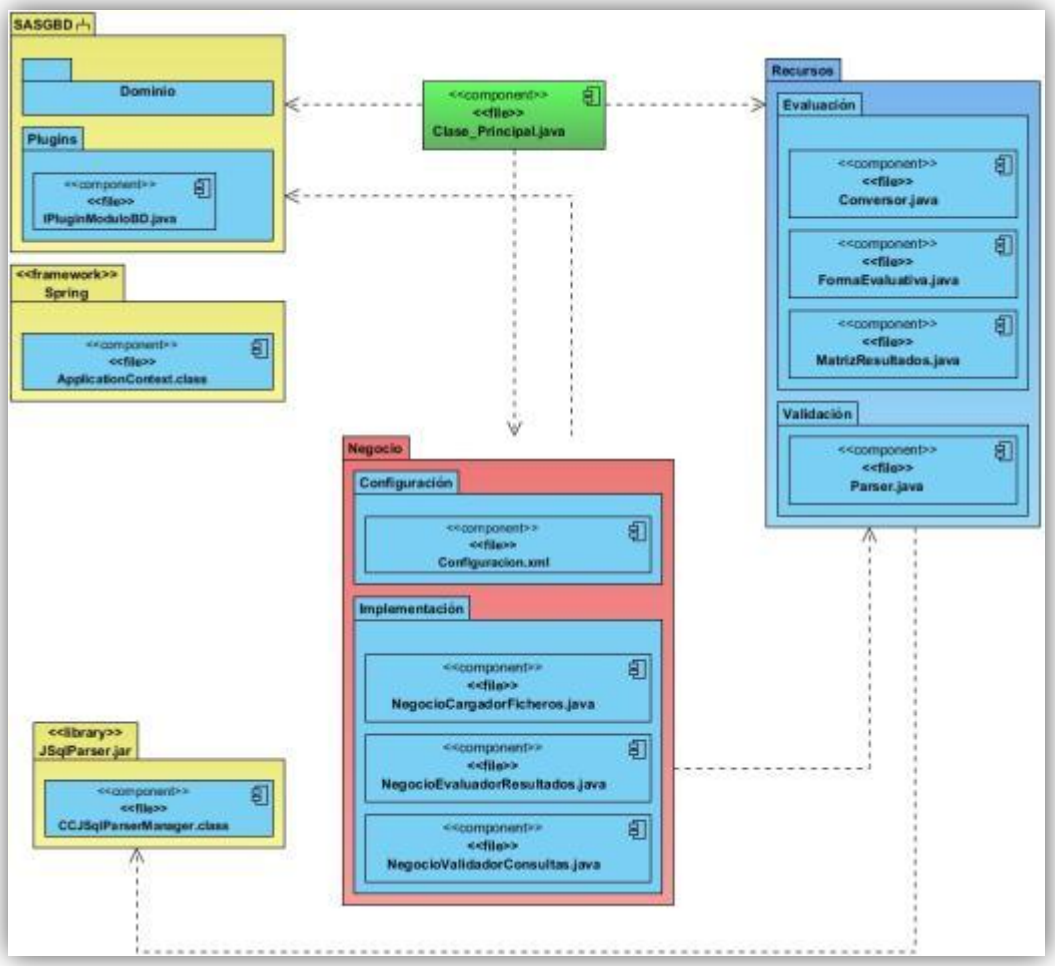

Figura 10. Diagrama de Componentes General de los Complementos.

A continuación se muestra la descripción de cada uno de los componentes que se muestran en la figura.

- **Clase_Principal.java:** Es quien conoce toda la lógica del negocio.
- **Negocio:** Contiene toda la lógica del negocio.
  - ✓ **Configuración:** Contiene las configuraciones.
    - o **Configuración.xml:** Contiene la configuración inicial de los complementos, toda la información que necesitan los complementos para su correcto funcionamiento. Este componente almacena datos como: las versiones soportadas del gestor que

soportan los complementos, los indicadores, las consultas, sus formas evaluativas con sus intervalos, valores predeterminados o variantes.

- **SpringXMLConfig.xml:** Contiene la configuración de los beans de Spring.

✓ **Implementación:** Agrupa las principales funcionalidades de los complementos.

- **NegocioCargadorFicheros.java:** Es el encargado de cargar la información de los resultados y la configuración inicial de los complementos.
- **NegocioEvaluadorResultados.java:** Se encarga de realizar la evaluación de los resultados.
- **NegocioValidadorConsultas.java:** Su responsabilidad es la de validar la sintaxis de las consultas. Se apoya en el componente JSqlParser.

- **Recursos:** Comprende los componentes auxiliares.
  - ✓ **Evaluación:** Comprende los componentes auxiliares en el proceso de evaluación.
    - **Conversor.java:** Agrupa utilidades de conversión de objetos.
    - **FormaEvaluativa.java:** Es el único capacitado para emitir una evaluación tomando como entrada un valor o una lista de valores.
    - **MatrizResultados.java:** Almacena los resultados con sus respectivas evaluaciones.
  - ✓ **Validación:** Comprende los componentes auxiliares en el proceso de validación.
    - **Parser.java:** Es el encargado de validar la consulta.
- **SASGBD:** Subsistema externo.
  - ✓ **Dominio:** Contiene las clases que mapean la base de datos de SASGBD.
  - ✓ **Plugins:** Contiene la interfaz que han de implementar los complementos.
    - **IPluginModuloBD.java:** Interfaz que deben implementar los complementos.
- **Spring:** Marco de trabajo que garantiza la inversión del control a través de los siguientes componentes.
  - ✓ **ApplicationContext.java**
- **JSqlParser:** Librería que ofrece utilidades en la validación de consultas.

4.3 Pruebas

El flujo de pruebas le presta servicios a los demás flujos. Su principal objetivo es evaluar o valorar la calidad del producto a través de:

- Buscar y documentar errores.
- Validar el cumplimiento de requerimientos.
- Dar una indicación de calidad.[33]

En las pruebas un sistema o componente es ejecutado bajo unas condiciones o requerimientos especificados, los resultados son observados y registrados. La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación. [34]

4.3.1 Estrategia de Pruebas

Con el objetivo de garantizar el correcto funcionamiento de los complementos, se decidió aplicarles un conjunto de pruebas basadas en una estrategia compuesta por niveles, tipos de pruebas, métodos y técnicas de pruebas correspondientes en aras de perfeccionar las soluciones desarrolladas.

Dentro de los niveles generales de pruebas se encuentran:

- Pruebas Internas.
- Pruebas Cruzadas.
- Pruebas de Liberación.
- Pruebas de Aceptación.
- Pruebas Piloto.

La estrategia definida se centra en el nivel de Pruebas Internas donde se encuentran los niveles:

- Prueba de Unidad.
- Prueba Independiente.
- Prueba de Desarrollador.
- Prueba de Integración.
- Pruebas de Sistema.

[33] **Entorno Virtual de Aprendizaje.** Entorno Virtual de Aprendizaje. Entorno Virtual de Aprendizaje. [En línea] [Citado el: 20 de Mayo de 2012.] http://eva.uci.cu/file.php/158/Documentos/Recursos_didacticos/Presentaciones_digitales_UD_2/pruebas_de_unidad.pdf.

[34] **Pressman, Roger S.** Ingeniería del Software: un enfoque práctico. La Habana: Félix Varela, 2005.

Cada nivel se relaciona con los tipos de pruebas, estos a su vez se aplican a través de métodos para los cuales existen técnicas.

Niveles de Pruebas	Tipos de Pruebas	Métodos	Técnicas
Unidad.	Funcionalidad Función.	Caja Blanca.	Camino Básico.
Integración.	Funcionalidad Función.	Caja Negra.	Partición de Equivalencia.

Tabla 5. Estrategia de Pruebas.

4.3.2 Niveles de Pruebas

- **Pruebas de Unidad:** Constituyen la primera fase de las pruebas que se le aplican a cada módulo de un software de manera independiente. Su objetivo es verificar que el módulo, entendido como una unidad funcional de un programa independiente, está correctamente codificado.[35]
- **Pruebas de Integración:** Se utilizan para asegurar que los componentes en el modelo de implementación operen correctamente cuando son combinados para ejecutar un caso de uso. Se prueba un paquete o un conjunto de paquetes del modelo de implementación.[36]

4.3.3 Tipos de Pruebas

Funcionalidad Función: Pruebas que fijan su atención en la validación de las funciones, métodos, servicios, caso de uso.

4.3.4 Métodos y Técnicas de Pruebas

Caja Blanca

Verifican que se ejecuten todos los caminos independientes, los bucles y las demás estructuras internas y se prueban todas las decisiones lógicas en sus vertientes verdadera y falsa.

[35] **Entorno Virtual de Aprendizaje.** Entorno Virtual de Aprendizaje. Entorno Virtual de Aprendizaje. [En línea] [Citado el: 20 de Mayo de 2012.] http://eva.uci.cu/file.php/158/Documentos/Recursos_didacticos/Presentaciones_digitales_UD_2/pruebas_de_unidad.pdf.
[36] **Pressman, Roger S. Ingeniería del Software: un enfoque práctico**. La Habana : Félix Varela, 2005.

Camino Básico

Esta técnica consiste en derivar casos de prueba a partir de un conjunto dado de caminos independientes. Camino independiente es aquel que introduce por lo menos una sentencia de procesamiento (o valor de condición) que no estaba considerada. Para obtener el conjunto de caminos independientes se construirá el Grafo de Flujo asociado y se calculará su Complejidad Ciclomática.[37]

Esta complejidad se denota como V (G) y se calcula de las siguientes formas:

- V (G)=A-N+2, siendo A la cantidad de aristas y N la cantidad de nodos del grafo.
- V (G)=P, siendo P la cantidad de nodos predicados, que son los nodos de los que salen más de una arista.
- V (G)=R, siendo R la cantidad de regiones que contiene el grafo.

La complejidad calculada es correcta, si al calcularse por las tres vías anteriormente expuestas, el valor resultante es el mismo.

Caja Negra

Son las pruebas que se llevan a cabo sobre la interfaz del software. O sea, los casos de prueba pretenden demostrar que:

- Las funciones del software son operativas.
- La entrada se acepta de forma adecuada.
- Se produce un resultado correcto.
- La integridad de la información externa se mantiene.

Partición de Equivalencia

Esta técnica divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.[38]

[37] **Entorno Virtual de Aprendizaje.** Entorno Virtual de Aprendizaje. Entorno Virtual de Aprendizaje. [En línea] [Citado el: 20 de Mayo de 2012.] http://eva.uci.cu/file.php/158/Documentos/Recursos_didacticos/Presentaciones_digitales_UD_2/pruebas_de_unidad.pdf.

[38] **Pressman, Roger S.** Ingeniería del Software: un enfoque práctico. La Habana : Félix Varela, 2005.

4.3.5 Pruebas de Caja Blanca

A continuación se presenta una muestra de los principales resultados obtenidos al realizar las pruebas de caja blanca.

CU Evaluar Resultados

```
public MatrizResultados Evaluar (Element xml) throws Exception {
    LinkedList<Indicador> resultados = cargador.CargarFichero (xml); (1)
    this.resultadosPorIndicadores = resultados; (1)
    HashMap<String, FormaEvaluativa> mapa = ConstruirMapa (); (1)
    LinkedList<Tabla> indicadores = new LinkedList<Tabla> (); (1)
    for (int i = 0 (1); i < resultados.size () (1); i++ (7)) {
       Indicador indicador = resultados.get (i); (2)
       String nombreIndicador = indicador.getNombre (); (2)
       LinkedList<Resultado> resultIndicador=indicador.getResultados (); (2)
       LinkedList<Fila> filas = new LinkedList<Fila> (); (2)
        for (int j = 0 (2); j < resultIndicador.size () (2); j++ (6)) {
          Resultado resultado = resultIndicador.get (j); (3)
          LinkedList<String> valores = resultado.getValores (); (3)
          FormaEvaluativa forma=mapa.get (resultado.getIdConsulta ()); (3)
          Evaluacion evaluacion = Evaluacion.NOEVALUADO; (3)
          if (forma instanceof FeBuscarEnLista) (3) {
                FeBuscarEnLista buscar = (FeBuscarEnLista) forma; (4)
                evaluacion = buscar.Evaluar (valores); (4)
          } else {
                evaluacion = forma.Evaluar (valores.get (0)); (5)
          }
          String valoresString = LinkedListToString (valores); (6)
         filas.add(new Fila(resultado.getIdConsulta(), valoresString,
forma.getValoresSugeridos(),getPeso(resultado.getIdConsulta()),evaluacion.toS
tring(),evaluacion.toString(),"MEJORA")); (6)
       }
       indicadores.add (new Tabla (nombreIndicador, filas)); (7)
    }
       return new MatrizResultados (indicadores); (8)
}
```

Figura 11. Fragmento de código del CU "Evaluar Resultados".

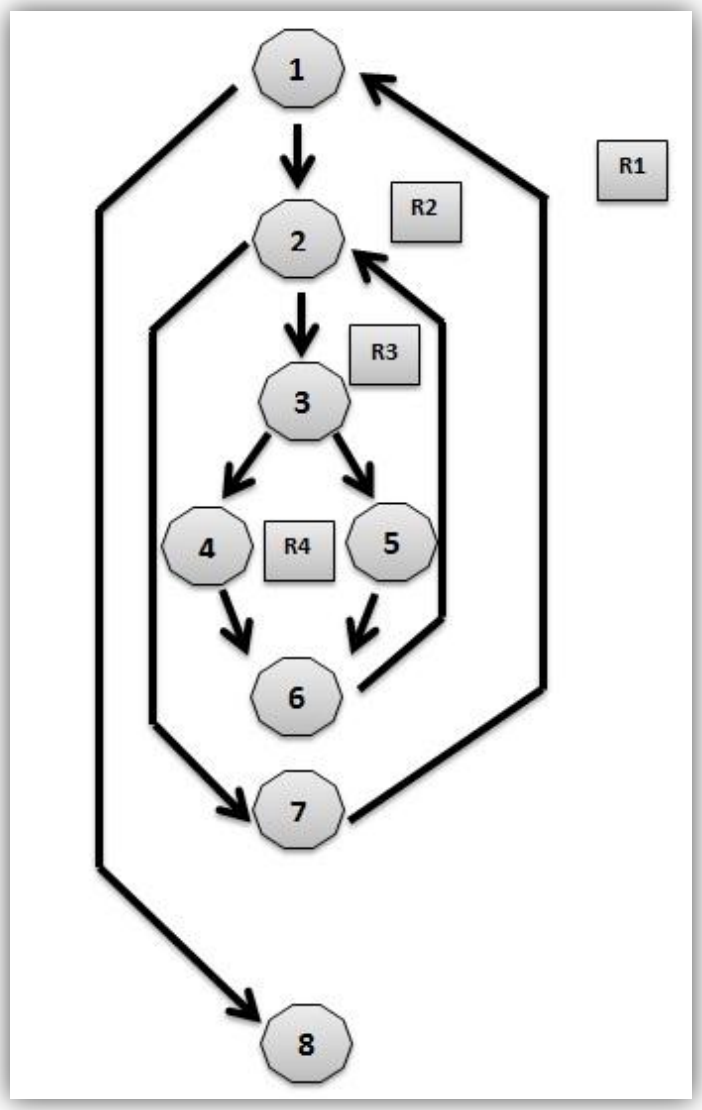

Figura 12. Grafo de Flujo del Caso de Uso "Evaluar Resultados".

V (G) = A - N + 2= 10 – 8 + 2 = 4.

V (G) = R = 4.

V (G) = P = 4.

Caminos: 1-8; 1-2-7-1-8; 1-2-3-4-6-2-7-1-8; 1-2-4-6-2-7-1-8;

Camino 1-8

Caso de prueba: Evaluar Resultados. Lista de indicadores vacía.

Entrada: Element xml.

Resultado: Se retorna una matriz vacía.

Camino 1-2-7-1-8

Caso de prueba: Evaluar Resultados. Lista de resultados correspondientes a un indicador vacía.

Entrada: Element xml.

Resultado: Se retorna una matriz con los nombres de los indicadores pero sin resultados.

Camino 1-2-3-4-6-2-7-1-8

Caso de prueba: Validar Consulta

Entrada: Element xml.

Resultado: Se retorna una matriz con resultados, algunos de ellos evaluados por formas evaluativas FeBuscarEnLista.

Camino 1-2-4-6-2-7-1-8

Caso de prueba: Validar Consulta

Entrada: Element xml.

Resultado: Se retorna una matriz con resultados, algunos de ellos evaluados por formas evaluativas FeBuscarEnIntervalos y/o FeBuscarEnVariantes.

CU Validar Consulta

```
public boolean ValidarConsulta(String consulta) throws Exception {
        boolean sintaxisOk = validador.Validar(consulta);(1)
        if (sintaxisOk) (1) {
                return true;(2)
        }
        return false;(3)
} (4)
```

Figura 13. Fragmento de código del CU "Validar Consulta".

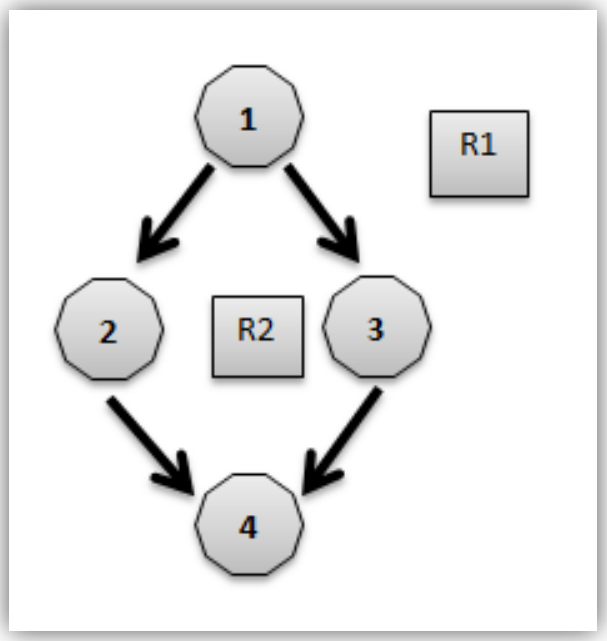

Figura 14. Grafo de Flujo del Caso de Uso "Validar Consulta".

V (G) = A - N + 2= 4 – 4+ 2 = 2.

V (G) = R = 2.

V (G) = P = 2.

Caminos: 1-2-4; 1-3-4

Camino 1-2-4

Caso de prueba: Consulta Válida.

Entrada: Para un valor de consulta = “select col1 from table1 where col1 > 1”.

Resultado: Retorna true.

Camino 1-3-4

Caso de prueba: Consulta no válida.

Entrada: Para un valor de consulta = “select * from tablex where;drop table t1;”.

Resultado: Retorna false.

4.3.6 Pruebas de Caja Negra

Debido a que los complementos no tienen interfaz gráfica, las pruebas de caja negra se realizarán usando el Módulo de Bases de Datos. A continuación se muestran los diferentes casos de pruebas de los diferentes casos de uso.

CU Evaluar Resultados

Descripción general				
Se carga un fichero con resultados para que sean evaluados los mismos por el sistema.				
Condiciones de ejecución				
Debe existir un fichero de resultados.				
SC Evaluar Resultados				
Escenario	**Descripción**	**Fichero**	**Respuesta del sistema**	**Flujo central**

EC 1.1 **Evaluar Resultados**	Se carga el fichero de una ubicación introducida y el sistema evalúa los resultados que tiene el fichero.	V.	El sistema carga el fichero y evalúa cada uno de los resultados que contiene devolviendo el siguiente mensaje: "Operación Satisfactoria".	1- Clic en el menú Evaluar Resultados. 2- El sistema muestra una interfaz solicitando los siguientes datos: ✓ Dirección del fichero. 3- Se introduce la dirección del fichero y selecciona el botón "Aceptar". 4- El sistema evalúa los resultados contenidos en el fichero. 5- El sistema muestra el mensaje: "Operación Satisfactoria".
EC 1.2 **Dirección incorrecta.**	Se introduce una dirección incorrecta.	I.	El sistema muestra el mensaje: "El archivo no es válido".	1- Clic en el menú cargar fichero. 2- El sistema muestra una interfaz solicitando los siguientes datos: ✓ Dirección del fichero. 3- Se introduce la dirección del fichero y selecciona el botón "Aceptar". 4- Muestra el mensaje: "El

				archivo no es válido".

Tabla 6. Casos de Prueba para el CU "Evaluar Resultados".

Descripción de las variables.				
No	**Nombre de campo**	**Clasificación**	**Valor Nulo**	**Descripción**
1	Fichero	Campo de Texto	No	Cadena de texto que especifica la Dirección del fichero resultado.

Tabla 7. Descripción de las Variables de los Casos de Prueba "Evaluar Resultados".

CU Validar Consultas

Descripción general
Valida que las consultas que introduce el auditor en la BD no presenten errores en la sintaxis y que sean solamente consultas del tipo SELECT.
Condiciones de ejecución
Debe existir una BD para introducir las consultas.
SC Validar Consulta

Escenario	Descripción	Nueva Consulta	Respuesta del sistema	Flujo central
EC *1.1* Validar Consulta	Verifica que la consulta tenga una sintaxis correcta y sea del tipo SELECT.	V. "select col1 from table1 where col1 > 1"	Se introduce la consulta en la BD y se muestra el mensaje: "Operación Satisfactoria".	1- Clic en el menú validar consulta. 2- El sistema muestra una interfaz solicitando los siguientes datos: ✓ Nueva Consulta. 3- Introduce la consulta y selecciona el botón "Aceptar". 4- Introduce la consulta en la BD.
EC 1.2 Consulta No Permitida	El tipo de consulta que se desea introducir no está permitida.	I. "drop table t1"	Muestra el mensaje: "Las consultas deben ser solamente del tipo SELECT".	1- Clic en el menú validar consulta. 2- El sistema muestra una interfaz solicitando los siguientes datos: ✓ Nueva Consulta. 3- Introduce la consulta y selecciona el botón "Aceptar". 4- Muestra un mensaje que diga" Las consultas deben ser solamente del tipo SELECT".
EC 1.3 Sintaxis Incorrecta.	El tipo de consulta que se desea introducir no presenta una sintaxis correcta.	I. ""select col1;col2 from table1 where col1 > 1""	Muestra un mensaje que diga "Sintaxis Incorrecta".	1- Clic en el menú validar consulta. 2- El sistema muestra una interfaz solicitando los siguientes datos: ✓ Nueva Consulta. 3- Introduce la consulta y selecciona el botón "Aceptar". 4- Muestra el mensaje:" Sintaxis

				Incorrecta".

Tabla 8. Casos de Prueba para el CU "Validar Consulta".

Descripción de las variables.				
No	**Nombre de campo**	**Clasificación**	**Valor Nulo**	**Descripción**
1	Nueva Consulta.	Campo de texto.	No.	Cadena de texto que debe cumplir con la sintaxis de la consulta y contener la instrucción SELECT.

Tabla 9. Descripción de las Variables del Caso de Prueba Validar Consulta.

4.3.7 Resultados de las Pruebas

Elemento	No	Ubicación de la No Conformidad	Etapa de Detección	Significativa	Estado NC
Carga solo los datos del último resultado.	1	Cargar información de resultados.	1ra Iteración.	NO.	-7/5/2012 pendiente. -12/5/2012 resuelta.
No carga las consultas.	2	Cargar Configuración Inicial.	1ra Iteración.	NO.	-7/5/2012 pendiente. -8/5/2012 resuelta.
Carga solo la forma evaluativa Buscar en Lista.	3	Cargar Configuración Inicial.	1ra Iteración.	NO.	-7/5/2012 pendiente. -10/5/2012 resuelta.

No convierte las formas evaluativas de las consultas.	4	Evaluar Resultados.	1ra Iteración.	SI.	-7/5/2012 pendiente. -11/5/2012 resuelta.
No evalúa los resultados.	5	Evaluar Resultados.	1ra Iteración.	SI.	-7/5/2012 pendiente. -13/5/2012 resuelta.
Construye el mapa de <id, Forma Evaluativa> incorrectamente.	6	Evaluar Resultados.	1ra Iteración.	NO.	-7/5/2012 pendiente. -11/5/2012 resuelta.
No valida que las consultas sean solo del tipo SELECT.	7	Validar Consulta.	1ra Iteración.	NO.	-7/5/2012 pendiente -12/5/2012 resuelta.
No carga todos los valores del método que obtiene el fichero de resultados.	8	Evaluar Resultados.	1ra Iteración.	NO.	-7/5/2012 pendiente. -18/5/2012 resuelta.
Carga un solo id. de las consultas.	9	Cargar información de resultados.	1ra Iteración.	NO.	-7/5/2012 pendiente. -17/5/2012 resuelta.
No valida que la consulta contenga dentro de la sentencia	10	Validar Consulta.	1ra Iteración.	SI.	-7/5/2012 pendiente. -17/5/2012 resuelta.

funciones: UPDATE, INSERT, DELETE.					
Evalúa un solo resultado.	11	Evaluar Resultados.	1ra Iteración.	SI.	-7/5/2012 pendiente. -16/5/2012 resuelta.
No devuelve la evaluación esperada.	12	Evaluar Resultados.	2da Iteración.	NO	-14/5/2012 pendiente. -20/5/2012 resuelta.

Tabla 10. No Conformidades.

Se realizaron 2 iteraciones de pruebas luego de implementar las funcionalidades en las cuales se detectaron un total de 12 no conformidades en los diferentes CU. De ellas, 4 fueron significativas. La siguiente figura muestra las no conformidades detectadas en cada iteración. Todas fueron resueltas al término de la segunda iteración por lo que se puede concluir que las pruebas fueron exitosas, el sistema está apto para su utilización y cumple con todas las funcionalidades para las que fue concebido.

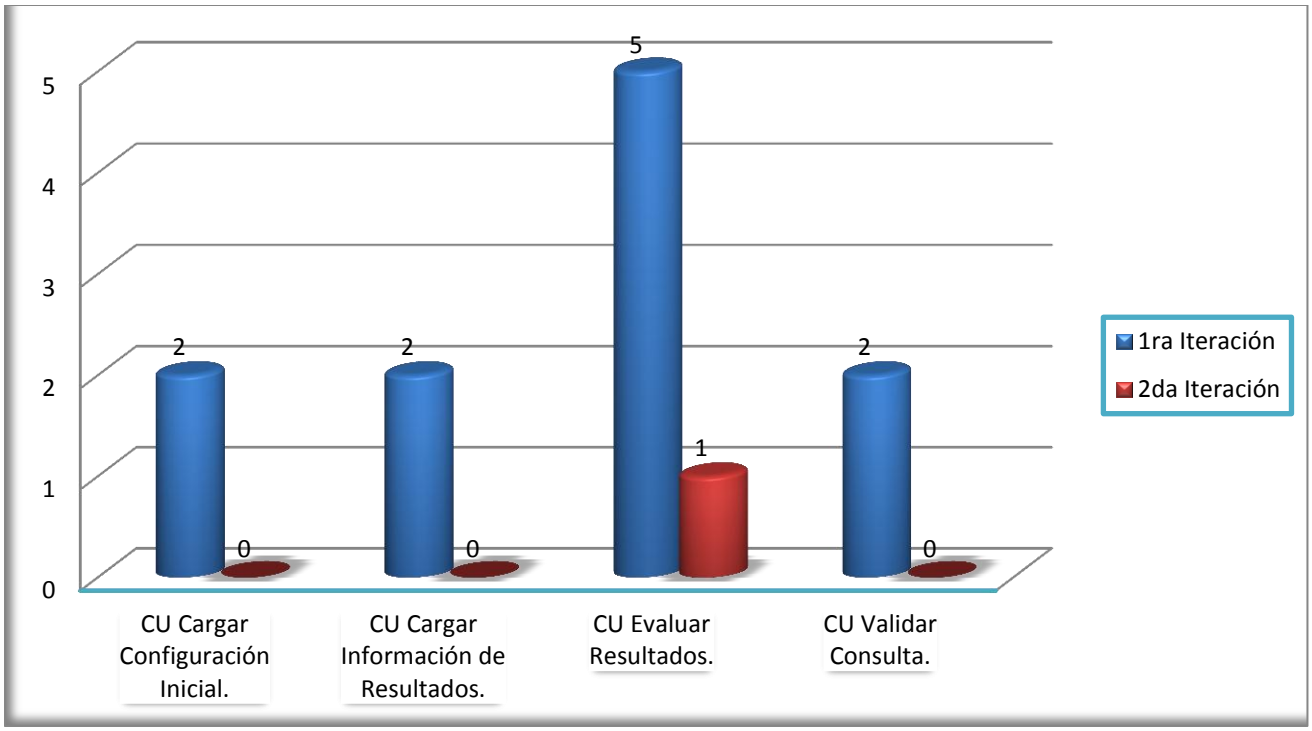

Figura 15. No Conformidades por Iteraciones.

4.4 Conclusiones

En el presente capítulo se abordaron los flujos de trabajo Implementación y Prueba, donde se confeccionó el diagrama de componentes general. Se utilizó una estrategia de prueba que se basó en los niveles de Integración y Unidad, utilizando los tipos de prueba Funcionalidad Función. Se utilizaron los métodos de prueba Caja Blanca y Caja Negra, recurriendo a las técnicas de prueba Camino Básico y Partición de Equivalencia.

CONCLUSIONES GENERALES

Se concluye este trabajo de diploma dando cumplimiento al objetivo propuesto para la realización del mismo. Mediante el presente documento, se ilustra al lector, acerca del proceso de desarrollo de los complementos para realizar la evaluación de los resultados de auditorías a SGBD Oracle y SQL Server, que forman parte del SASGBD.

Además esta investigación arrojó los siguientes resultados:

- Se formalizó un estudio sobre cómo el Departamento de Seguridad Informática de ETECSA realiza la evaluación de los resultados de auditorías a los SGBD Oracle y SQL Server que permitió comprender los procesos del negocio envueltos en el campo de acción y las posibles actividades a automatizar.
- Se realizó una caracterización de los sistemas informáticos existente en el mundo que realizan auditorías de seguridad informática a SGBD Oracle y SQL Server que posibilitó conocer otras soluciones similares existentes en el mercado para validar la necesidad de la propuesta de solución.
- Fueron modelados dichos complementos utilizando la notación BPMN, en el caso del negocio y el lenguaje de modelado UML, en el caso de los demás flujos de trabajos que propone la metodología RUP para el desarrollo de sistemas informáticos.
- Se modeló el diseño de los complementos que aseguró las bases para la implementación.
- Se implementaron los complementos para realizar la evaluación de los resultados de auditorías a los SGBD Oracle y SQL Server.
- Se realizaron pruebas de caja blanca y de caja negra a ambos complementos que contribuyeron a verificar el correcto funcionamiento de los mismos.
- Los complementos que culminan proveen de un mecanismo al SASGBD para la evaluación de los resultados de auditorías a SGBD.
- Los complementos al ser multiplataforma y estar desarrollados sobre tecnologías de código abierto contribuyen al proceso de migración al software libre en el que el país está envuelto.

RECOMENDACIONES

La experiencia acumulada durante la realización de esta investigación, permite plasmar las siguientes recomendaciones:

- Utilizar esta investigación como referencia para investigaciones futuras en el área de la evaluación de auditorías a SGBD, especialmente Oracle y SQL Server.
- Brindar soporte para versiones futuras de Oracle y SQL Server.
- Incluir técnicas de inteligencia artificial en el proceso de evaluación.
- Profundizar en el estudio de las premisas o requisitos de seguridad que plantean los SGBD Oracle y SQL Server, para una mejor precisión del producto.
- Incluir mecanismos para conocer si los SGBD mencionados, son vulnerables a inyecciones SQL.
- Continuar agregando nuevas funcionalidades de acuerdo con las necesidades que surjan.
- Realizar pruebas de liberación y aceptación con el cliente.

BIBLIOGRAFÍA

1. **Velasco, Roberto Hernando.** El SGBDR Oracle. *rhernando.net.* [En línea] [Citado el: 08 de 12 de 2011.] http://www.rhernando.net/modules/tutorials/doc/bd/oracle.html.
2. **Wikipedia.** Oracle 11g. *Wikipedia.* [En línea] Fundación Wikipedia, Inc. [Citado el: 08 de 12 de 2011.] http://es.wikipedia.org/wiki/Oracle_11g.
3. **ETECSA.** Misión y Visión. *Portal Web de la Empresa de Telecomunicaciones de Cuba.* [En línea] ETECSA. [Citado el: 1 de Diciembre de 2011.] http://www.etecsa.cu/index.php?sel=nuestraempresa&content=id_mision_vision.
4. **Saberia.com.** Qué es un plugin. *Saberia.com.* [En línea] 2010. [Citado el: 18 de Enero de 2012.] http://www.saberia.com/2010/01/que-es-un-plugin/.
5. **ISO.** ISO 9000:2000. *CNQA.* [En línea] http://www.cnqa.org/upimg/200921671914457.pdf.
6. **Oracle Corporation.** Oracle Audit Vault. *Oracle.* [Online] [Cited: Enero 19, 2012.] http://www.oracle.com/technetwork/database/audit-vault/overview/index.html.
7. **SoftTree Technologies Inc.** DB Tools for Oracle Version 6.1. *softtreetech.com.* [En línea] [Citado el: 18 de Enero de 2012.] http://www.softtreetech.com/dbtools/.
8. **Secure Bytes.** Auditor de seguridad de Oracle(SOA). [Online] 2011. [Cited: Octubre 05, 2011.] http://secure-bytes.com.
9. **Bizagi.** BPMN Bussines Process Modeling Notation. *Bizagi.* [En línea] 2011. [Citado el: 17 de Enero de 2012.] www.bizagi.com/docs/BPMNbyExampleSPA.pdf.
10. **Oracle Corporation.** Bienvenido a NetBeans y www.netbeans.org. *NetBeans.* [En línea] 2012. [Citado el: 17 de Enero de 2012.] http://netbeans.org/index_es.html.
11. **Pergamino Virtual.** Definición de Java. *Pergamino Virtual.* [En línea] 2011. [Citado el: 17 de Enero de 2012.] http://www.pergaminovirtual.com.ar/definicion/Java.html.
12. **Cima Consulting.** Oracle Audit Vault. *Cima Consulting.* [En línea] 2010. [Citado el: 18 de Enero de 2012.] http://www.cimaconsulting.com.mx/mportal/index.php?option=com_content&task=view&id=323&Itemid=125.
13. **Gómez, Juan Pablo.** RUP. *Scribd.* [En línea] 16 de Septiembre de 2007. [Citado el: 18 de Enero de 2012.] http://es.scribd.com/doc/297224/RUP.
14. **EcuRed.** Visual Paradigm. *EcuRed.* [En línea] [Citado el: 18 de Enero de 2012.] http://www.ecured.cu/index.php/Visual_Paradigm.
15. **Martínez, Rafael.** Sobre PostgreSQL. *PostgresSQL-es.* [En línea] 2 de Octubre de 2010. [Citado el: 17 de Enero de 2012.] http://www.postgresql.org.es/sobre_postgresql.
16. **Picando Codigo.** Introducción a Spring Framework Java. *Picando Codigo.* [En línea] [Citado el: 20 de Enero de 2012.] http://picandocodigo.net/2010/introduccion-a-spring-framework-java/.
17. **Oracle Corporation.** *Oracle.com.* [En línea] [Citado el: 22 de Enero de 2012.] http://www.oracle.com/lad/corporate/press/press-release-laddec14-09-333682-esa.html.
18. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* 1998. págs. 3-4.
19. **Hoy Digital.** Tecnología ¿Qué es auditoría informática? *Hoy Digital.* [En línea] 5 de Enero de 2009. [Citado el: 18 de Enero de 2012.] http://www.hoy.com.do/investigacionTecnologiaQue-es-auditoria-informatica..

20. **Raráz Tinoco, Jorge Luis.** *slideshare.net.* [En línea] [Citado el: 19 de Enero de 2012.] http://www.slideshare.net/jorg_leoxd/comparacion-entre-my-sql-y-sql-server.
21. **Ramos Ortega, Diego Martin.** Microsoft SQL Server - Monografías.com. *monografias.com.* [En línea] [Citado el: 19 de Enero de 2012.] http://www.monografias.com/trabajos73/microsoft-sql-server/microsoft-sql-server.shtml.
22. **Alvarez, Gonzalo Marañón.** Que es Java. *iec.csic.es.* [En línea] [Citado el: 18 de Enero de 2012.] http://www.iec.csic.es/criptonomicon/java/quesjava.html.
23. **FileCluster.** NetBeans IDE 7.1. *filecluster.es.* [En línea] [Citado el: 19 de Enero de 2012.] http://www.filecluster.es/programas/NetBeans-IDE-144777.html.
24. **Thayer, R.H. y Dorfman, M.** *Software Requeriments Engineering.* 2nd. s.l. : IEEE Computer Society Press, 1997.
25. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* 1era en Español. s.l. : Pearson Educación, S.A., 2000.
26. **Pimentel, Annia y Hernández, Antonio.** *Sistema Informático para la Gestión de Auditoría y Control (SIGAC). Módulo de Planificación.* 2009. Tesis de pregrado.
27. **Rbytes reviews.** Rbytes reviews, Descargar Eclipse Classic v3.3.1.1. *Rbytes reviews.* [En línea] [Citado el: 18 de Enero de 2012.] http://rbytes.org/descargar/cat/programaci%C3%B3n/java--java-script/eclipse-classic/.
28. **Entorno Virtual de Aprendizaje.** Entorno Virtual de Aprendizaje. *Entorno Virtual de Aprendizaje.* [En línea] [Citado el: 20 de Mayo de 2012.] http://eva.uci.cu/file.php/158/Documentos/Recursos_didacticos/Presentaciones_digitales_UD_2/pruebas_de_unidad.pdf.
29. **Deldago Picazo, Mario.** E-Archivo. Repositorio Institucional de la Universidad Carlos III. [En línea] [Citado el: 18 de Enero de 2012.] http://e-archivo.uc3m.es/bitstream/10016/6247/1/PFC_MDP_v1.0.pdf.
30. **ApexSQL.** ApexSQL. *ApexSQL.* [En línea] [Citado el: 19 de Enero de 2012.] http://www.apexsql.com/sql_tools_audit.aspx.
31. **Componet Source.** Componet Source. [En línea] [Citado el: 19 de Enero de 2012.] http://www.componentsource.com/products/apexsql-audit/index-es.html.
32. **García Peñalvo, Francisco José, Conde González, Miguel Ángel y Bravo Martín, Sergio .** OpenCourseWare de la Universidad de Salamanca. *OpenCourseWare de la Universidad de Salamanca.* [En línea] 16 de Octubre de 2008. [Citado el: 20 de Febrero de 2012.] http://ocw.usal.es/ensenanzas-tecnicas/ingenieria-del-software/contenidos/Tema6-DOO-1pp.pdf.
33. **SpringSource.org.** SpringSource.org. *SpringSource.org.* [En línea] [Citado el: 20 de Febrero de 2012.] http://static.springsource.org/spring/docs/1.2.x/reference/beans.html#beans-factory-modes.
34. **Pressman, Roger S.** *Ingeniería del Software: un enfoque práctico.* La Habana : Félix Varela, 2005.
35. **Instituto de Ingeniería Eléctrica (IIE).** Sitio Web Instituto de Ingeniería Eléctrica de la Universidad de la República. [En línea] [Citado el: 3 de Marzo de 2012.] http://iie.fing.edu.uy.
36. **FreeDownLoadManager.** Auditor de Seguridad del SQL (Secure SQL Auditor). *freedownloadmanager.org.* [En línea] [Citado el: 18 de Enero de 2012.] http://www.freedownloadmanager.org/es/downloads/Auditor_de_Seguridad_del_SQL_80080_p/.
37. **FreeDownloadManager.** Auditor SQL de Apex (Apex SQL Audit). *freedownloadmanager.org.* [En línea] [Citado el: 18 de Enero de 2012.] http://www.freedownloadmanager.org/es/downloads/%C3%81pice_Auditor%C3%ADa_de_SQL_2909_p/.
38. **Auditoria.Com.Mx.** AUDITORIA Y SEGURIDAD INFORMATICA. [En línea] [Citado el: 19 de Enero de 2012.] http://www.auditoria.com.mx/productos/analyze/apex/apexsql.htm.

39. **Catarina.Udlap.Mx.** Capítulo 3: Spring un framework de aplicación. *catarina.udlap.mx.* [En línea] [Citado el: 19 de Enero de 2012.] http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/sanchez_r_ma/capitulo3.pdf.
40. **DefinicionABC.Com.** *definicionabc.com.* [En línea] [Citado el: 31 de Enero de 2012.] http://www.definicionabc.com/general/script.php.
41. **Dsi.Uclm.Es.** *dsi.uclm.es.* [En línea] [Citado el: 19 de Mayo de 2012.] http://www.dsi.uclm.es/asignaturas/42530/pdf/M2tema12.pdf.
42. **Gestionrrhhusm.Blogspot.Com.** *gestionrrhhusm.blogspot.com.* [En línea] [Citado el: 18 de Enero de 2012.] http://gestionrrhhusm.blogspot.com/2011/05/modelo-rup-rational-unified-process-o.html.
43. **Mato García, Rosa María.** *Diseño De Bases de Datos.* 2nd. 2005. pág. 2.
44. **Adescargar.Es.** Secure Ora Auditor. *adescargar.es.* [En línea] [Citado el: 20 de Enero de 2012.] http://www.adescargar.es/es/software/123633/secure+ora+auditor+2.0.1267.0081.
45. **Cnet.Com.** DB Tools for Oracle. *cnet.com.* [En línea] [Citado el: 20 de Enero de 2012.] http://download.cnet.com/DB-Tools-for-Oracle/3000-10254_4-10034993.html.
46. **Cqure.Net.** cquere.net Oracle Auditing Tools. *cqure.net.* [En línea] [Citado el: 18 de Enero de 2012.] http://www.cqure.net/wp/test/.
47. **Definicion.De.** Definición de Auditoría. *Definicion.de.* [En línea] 2010. [Citado el: 18 de Enero de 2012.] http://definicion.de/auditoria/.
48. **Gudu Software.** General SQL Parser. *General SQL Parser.* [En línea] 2012. [Citado el: 20 de Enero de 2012.] http://www.sqlparser.com.
49. **Sourceforge.net.** JSqlParser. *JSqlParser.* [En línea] [Citado el: 18 de Enero de 2012.] http://jsqlparser.sourceforge.net/.