

Universidad de las Ciencias Informáticas

FACULTAD 6



Título: Herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.

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

Autores:

Yasmari Soto Rivero

Keimer Montes Oliver

Tutores:

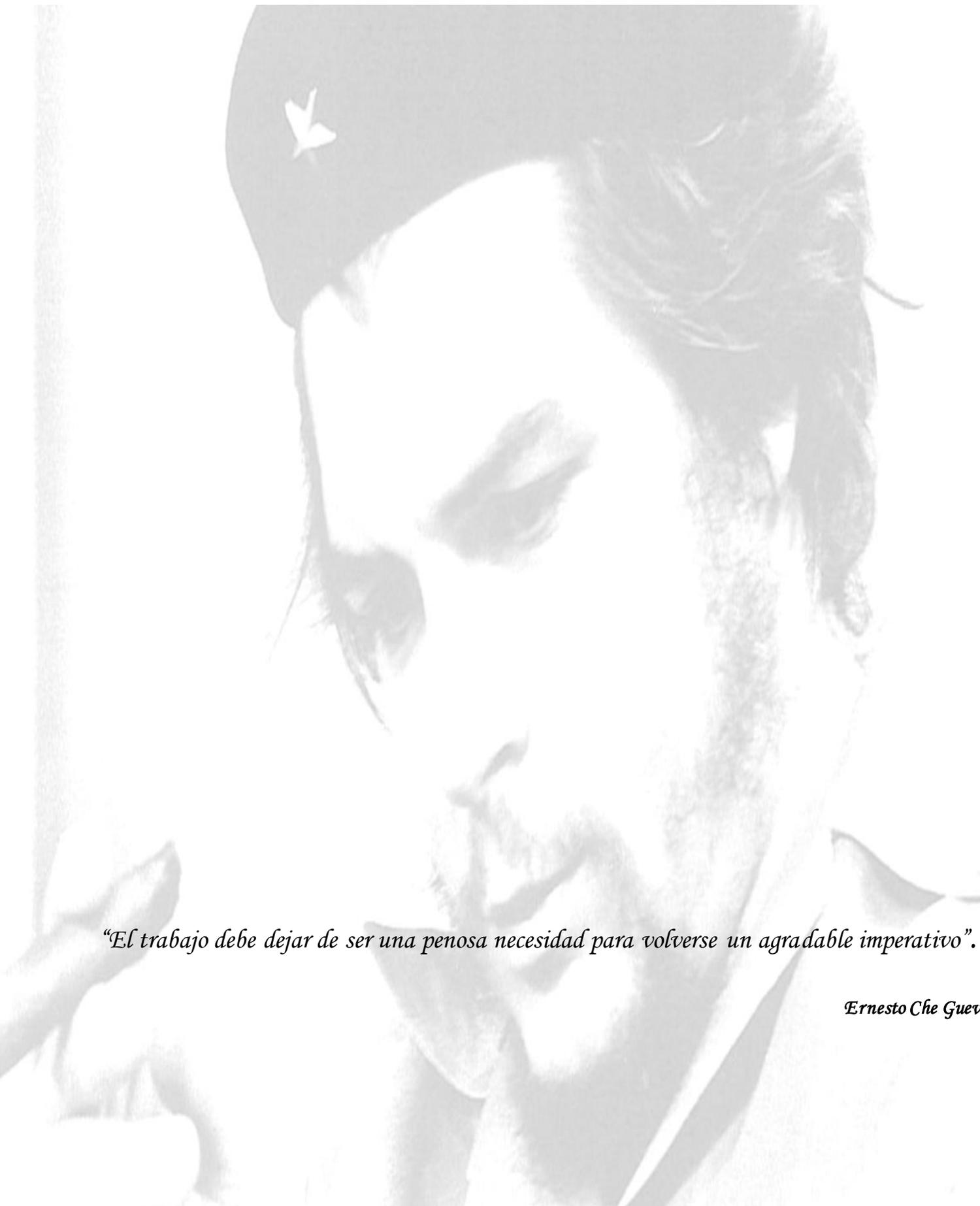
Ing. Yuneimy Tellez Pérez

Ing. Carlos González Iglesias

Ing. Lully Díaz Montero

La Habana, 2013

“Año 55 de la Revolución”



“El trabajo debe dejar de ser una penosa necesidad para volverse un agradable imperativo”.

Ernesto Che Guevara

DECLARACIÓN DE AUTORÍA

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firma la presente a los ____ días del mes de junio del año 2013.

Yasmari Soto Rivero

Keimer Montes Oliver

Firma del Autor

Firma del Autor

Yuneimy Téllez Pérez

Carlos González Iglesias

Lully Díaz Montero

Firma del Tutor

Firma del Tutor

Firma del Tutor

DATOS DE CONTACTO

Autores:

Yasmari Soto Rivero

Universidad de las Ciencias Informáticas

e-mail: ysoto@estudiantes.uci.cu

Keimer Montes Oliver

Universidad de las Ciencias Informáticas

e-mail: kmontes@estudiantes.uci.cu

Tutores:

Yuneimy Téllez Pérez

Ingeniero de las Ciencias Informáticas

e-mail: ytellez@uci.cu

Lully Díaz Montero

Ingeniero de las Ciencias Informáticas

e-mail: ldmontero@uci.cu

Carlos González Iglesias

Ingeniero de las Ciencias Informáticas.

e-mail: cgonzalez@uci.cu

Agradecimientos de Yasmari:

Quiero agradecer a todas aquellas personas que me han apoyado, querido y creído en mí y que han contribuido de una forma positiva en mi vida.

Hay una persona que me enseñó a tener el carácter, la disciplina, el profesionalismo que hoy tengo. Me enseñó que las oportunidades pasan una sola vez en la vida y hay que saberlas aprovechar. Te agradezco a ti mi hermana, por haberme educado así, porque hoy estoy orgullosa de ser quien soy, por ser mi ejemplo a seguir durante todos estos años, por ser mi luz y mi guía en todo momento. Siempre te decía cuando era niña que eras una hermana mala, por llevarme tan recio y me decías que algún día te lo iba agradecer, ese día es hoy, no existen palabras en este mundo para describir lo que siento por ti, te quiero mi hermana linda.

Estas líneas que voy a escribir son para agradecer a una hermosa mujer, al ángel dulce, tierno y maravilloso que hizo posible que yo estuviera aquí hoy: A mi linda madre, la que ha sido madre y padre al mismo tiempo, que ha dedicado su vida entera a darme amor, cariño y ser la amiga que necesité. A la que siempre estaré eternamente agradecida y que me alegro que existan momentos como estos para agradecerle una vez más por hacer de mi vida una gran persona. Hoy quiero darte las gracias por todo lo que me has brindado y darle gracias a Dios por darme una madre como tú.

En esta vida, gracias a dios nada sucede como deseamos, como suponemos, ni como tenemos previsto. Quiero agradecer a una persona que ha sido como un padre para mí, a mi cuñado Edgar y agradecerle por ser capaz de hacer feliz a una de las personas más importantes en mi vida.

Doy las gracias a toda mi familia en especial a mi padrastro por quererme como una hija, a mis lindos sobrinitos, a mis abuelos, a mis tías, a mis primas Mara y Mamita, a mi tío Juancito y a mi primo Dunny

que me han apoyado y ayudado siempre. A Lién y a Papito que me han acogido en su casa como si fuera su propia hija y me han dado todo su cariño y amor.

Quiero agradecer a una persona muy especial en mi vida, que ha sido mi compañera, amiga y hermana en esta Universidad: A ti Lieny, por quererme aunque sea insoportable, por estar siempre presente cuando lo he necesitado, por compartir mis rizas y mis lágrimas. Quiero que sepas que te quiero mucho y que nunca te voy a olvidar.

Cuando menos lo esperamos la vida nos coloca delante de un desafío que pone a prueba nuestro coraje y nuestra voluntad de cambio. Quiero agradecer a una persona que me hace sonreír solo con una mirada, gracias Jorge Alejandro por todo el tiempo que compartimos juntos.

Le agradezco a todas mis amistades porque de una manera u otra estuvieron siempre presente: a mis amistades de la primaria, secundaria y pre: Axiela, Mirelis, Lilian, Zuleidis, Aliuska, Danieska, Ramón, Ramoncetín y Pintico, a Leo y Yasel por ser los primeros amigos que me apoyaron en todo cuando entre a esta universidad, a Are por ser la primera personita con la que compartí una verdadera amistad, a Karel por haber compartido siempre todos sus conocimientos, a Alex por ser un buen compañero, a Martica por preocuparse siempre por mí, a Tomy por la amistad que hemos compartido estos últimos meses y por aconsejarme positivamente, al piquete más explosivo de la UCI: Poti, Popi, Marquito, Raúl, los quiero mucho, a los chicos del 95 que son tremendos, en entre ellos : Emir, Bárbaro, Reinier, Adrián, Carlos, Ariel y en especial a Danny que le tengo tremendo cariño ,te quiero mucho y darte las gracias porque gracias a ti conocí a Jorgito.

A Keimer, quien ha sido la persona que siempre ha estado presente en mis buenos y malos momentos, por soportarme, quererme y siempre creer en mí.

Doy las gracias a mis tutores por su grandísimo esfuerzo: Gracias Yuneimy por apoyarme todo este tiempo y porque sin ti esto no hubiese sido posible. Luilly te agradezco la forma en que contribuiste con este trabajo, eres una persona que da el paso al frente cuando es necesario.

También le agradezco a aquellos profesores que marcaron mi vida en cada año de mi carrera: Manolo, Reinaldo Luna, Anthony, Yindra, Yosuan, Reinier, Yadira, Yonelbys, Ana Mary y por último a mi profesor preferido, al cual admiro mucho Lacoste.

Agradecimientos de Keimer:

Siempre soñé con este momento y si lo estoy viviendo hoy es gracias a una persona muy especial, que es lo más importante para mí y lo que más quiero en esta vida, mi madre Gloria. Gracias por permitirme ser quien soy, por guiarme siempre y estar presente cuando te necesité. Gracias por nunca permitirme que me rindiera, por alentarme, por estar orgullosa de mí, por ver siempre lo mejor de mí y por ser madre y padre a la vez. Gracias por aconsejarme y convertirme en la persona que soy. Sin ti no lo había logrado, tú fuiste y serás la razón por la que me levanto todos los días buscando la forma de ser una mejor persona y de que te sientas orgullosa de mí.

Durante toda mi vida, mi familia siempre me ha ayudado y apoyado en mis decisiones aunque no sean las más correctas. Cuando entre en la UCI se sacrificaron mucho para que yo no tuviera que pasar trabajo, gracias a todos, pero en especial le agradezco a mis abuelos, a mi padrastro que lo considero como mi padre y a mis hermanos Isel Alejandro, Marco y Luis Daniel, gracias a todos.

Cuando entre a la UCI, entre con un grupo de amigos que vienen conmigo desde el pre y siempre serán mis hermanos; Yisel, Yamile, Luis Carlos, Luis Daniel, Salgado, Gerardo, Yannier, Omar, Agustín, siempre los voy a tener presente, gracias a todos por permitirme ser su amigo.

El hecho que no entraras a la UCI no significa que no sea mi hermano, Ugarte a ti te tengo que agradecer aún más, porque te comportaste mejor que un hermano cuando lo necesitaba y me aconsejaste, me siento orgulloso de que me consideres tu amigo.

Tengo que agradecerle mucho a una persona que me hizo ver la UCI de otra forma, que solo no era juego, me cuidó y me puso por encima de sus intereses personales, nunca te voy a olvidar porque fuiste una persona muy especial, gracias por ayudarme y compartir conmigo todos esos momentos tan especiales que tuvimos juntos. Gracias Marcia.

Desde primer año he tenido muchos amigos que me han ayudado durante mi carrera pero no puedo dejar de mencionar a Maikel, Tony, Elizabeth, Mirita, Anabel, Antonio Miguel, Isenith, Julio, Javier, Yohan, Darien, Odalis, José Carlos Carmen y a Mirita, mi negra, que siempre me ayudó y me aconsejó, gracias a todos por compartir los momentos malos y buenos de la carrera, las malas noches cuando nos poníamos a estudiar, las borracheras de los cumpleaños y los ranchones.

He tenido muchos profesores bueno que me han ayudado y me educaron para que fuera una mejor persona, gracias Reinier, Liudmila, Yanelis y Yurima.

Durante la realización de mi tesis me ayudaron muchas personas pero no puedo dejar de mencionar a Pablo, que lo molesté tantas veces (en la interfaz de la aplicación) y amanecimos aprendiendo Spring, a Andry, Yosbel, Edel, Ernesto, René, Justin y muy especial a la profesora Yuneimy que me ayudó mucho en la tesis. Gracias a todos.

Y no por ser la última persona que agradezco no significa que no fueras y seas importante para mí, tú supiste soportarme durante más de dos años y me defendías como una madre defiende a su hijo. Me enseñaste muchas cosas durante todo este tiempo y siempre estabas a mi lado cuando te necesitaba. Nos fajamos mucho, pero mejor compañera de tesis no quería. Gracias por TODO Yasmari.

Dedicatoria de Yasmari:

A mi madre, por estar siempre presente cuando la necesito y a quien le debo todo lo que he logrado en esta vida.

A mi hermana que me apoyó con su espíritu alentador, contribuyendo incondicionalmente a lograr mis metas y objetivos propuestos y que al brindarme su ejemplo me enseñó a ser perseverante y darme la fuerza necesaria para siempre seguir adelante.

Dedicatoria de Keimer:

A toda mi familia, en especial a mi madre y a mis abuelos que siempre me han apoyado.

A mis abuelos Otilio y Ernestina que aunque no estén conmigo hoy sé que se sentirían orgullosos de mí.

RESUMEN

La presente investigación surge por la necesidad que existe en el departamento de Almacenes de Datos del Centro de Tecnologías de Gestión de Datos (DATEC) de automatizar los diferentes mecanismos de autenticación del Pentaho BI Server. Con el propósito de solucionar la dificultad mencionada anteriormente, se decidió desarrollar una herramienta para la configuración de la seguridad del Pentaho BI Server. Con este fin se realiza un estudio bibliográfico relacionado con los mecanismos de autenticación en la herramienta Pentaho BI Server y se definieron las tecnologías y programas para desarrollar el sistema. El proceso estuvo guiado por la metodología de desarrollo de software OpenUP y se utilizó como lenguaje de programación Java. Como resultado de la investigación se obtuvo una herramienta que permite mejorar la seguridad del Pentaho BI Server al realizar la configuración de los diferentes mecanismos de autenticación y del CAS de manera automática, de brindar la posibilidad de validar si un usuario pertenece al dominio, de garantizar que exista un control eficiente de las contraseñas y de ser capaz de restringir una dirección IP específica o un rango de direcciones IP. La herramienta se integra al Pentaho BI Server y puede ser utilizada por otras instituciones.

Palabras claves: Autenticación, Inteligencia de Negocios, Pentaho BI Server.

CONTENIDO

INTRODUCCIÓN..... 15

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS.....21

 1.1 Inteligencia de Negocios21

 1.1.1 Herramientas de Inteligencia de Negocios21

 1.2 Pentaho BI Server 24

 1.3 Mecanismos de autenticación..... 28

 1.4 Metodologías de desarrollo.....30

 1.5 Herramientas de modelado..... 31

 1.6 Sistema Gestor de Bases de Datos (SGBD).....32

 1.7 Lenguajes de programación34

 1.8 Framework de desarrollo Spring MVC 35

 1.9 Entorno de Desarrollo Integrado (IDE) 36

 Conclusiones del capítulo 37

CAPÍTULO 2: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA. 38

Introducción..... 38

 2.1 Propuesta de solución.....38

 2.2 Requisitos..... 39

 2.2.1 Requisitos funcionales 39

 2.2.2 Requisitos no funcionales 41

 2.3 Diagrama de casos de uso del sistema..... 44

 2.4 Diagrama de clases del diseño.....50

 2.5 Patrones 53

2.5.1	Patrones arquitectónicos	53
2.5.2	Patrones de diseño	54
2.6	Diagrama de secuencia	57
2.7	Diagrama de despliegue	59
	Conclusiones del capítulo	60
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA.....		61
Introducción.....		61
3.1	Modelo de implementación	61
3.2	Diagrama de componentes.....	61
3.3	Estándares de codificación.....	62
3.4	Mapa de navegación	66
3.5	Pruebas de software	66
3.5.1	Niveles de prueba	67
3.5.2	Método de pruebas de caja blanca	67
3.5.3	Método de pruebas de caja negra	68
3.5.4	Diseño de casos de prueba	68
3.5.5	Proceso de prueba de la herramienta.....	69
3.5.6	Aplicación de pruebas de caja blanca.....	69
3.5.7	Casos de pruebas de caja negra.....	70
3.5.8	Resultado de las pruebas	72
	Conclusiones del capítulo	74
CONCLUSIONES GENERALES.....		75
RECOMENDACIONES.....		76
REFERENCIAS BIBLIOGRÁFICAS		77

BIBLIOGRAFÍA	79
ANEXOS	¡Error! Marcador no definido.

ÍNDICE DE TABLAS

Tabla 1: Funcionalidades de las versiones de la consola de administración del Pentaho BI Server. 27

Tabla 2: Descripción del actor del sistema.....45

Tabla 3: Configurar gestores de base de datos.50

Tabla 4: Diagrama de clases del diseño para el caso de uso “Configurar gestores de base de datos”. 52

Tabla 5: Descripciones de las clases Postgres, Oracle y MySQL.....53

Tabla 6: Variables del caso de uso “Configurar gestores de base de datos”.71

Tabla 7: Caso de prueba para el escenario 1 “Configuración exitosa a Postgres”72

Tabla 8: Caso de prueba para el escenario 2 “Configuración con presencia de errores a Postgres”.72

ÍNDICE DE FIGURAS

Figura 1: Propuesta de la solución.	38
Figura 2: Diagrama de casos de uso del sistema.....	44
Figura 3: Diagrama de clases del diseño para el caso de uso “Configurar gestores de base de datos”. ..	51
Figura 4: Arquitectura del patrón arquitectónico Modelo-Vista-Controlador.	54
Figura 5: Patrón GRASP Experto.	55
Figura 6: Patrón GRASP Experto.	56
Figura 7: Patrón GRASP Alta Cohesión.....	56
Figura 8: Patrón GRASP Controlador.	57
Figura 9: Diagrama de secuencia de la sección ConfigurarPostgres del caso de uso “Configurar gestores de base de datos”.	58
Figura 10: Diagrama de secuencia de la sección ConfigurarPostgres del caso de uso “Configurar gestores de base de datos”.	59
Figura 11: Diagrama de secuencia de la sección ConfigurarMySQL del caso de uso “Configurar gestores de base de datos”.	59
Figura 12: Diagrama de despliegue de la herramienta.	60
Figura 13: Diagrama de componentes para el caso de uso “Configurar gestores de base datos“. ..	62
Figura 14: Mapa de navegación de la herramienta.	66
Figura 15: Proceso de prueba para la herramienta.	69
Figura 16: Enumeración de código para el método setAtributo.	70
Figura 17: Grafo de flujo para el método setAtributo.	70
Figura 18: Cantidad de no conformidades.	73
Figura 19: Clasificación de las no conformidades.	73

INTRODUCCIÓN

En la actualidad, la información juega un papel importante en el desarrollo económico y es un factor clave para la búsqueda de nuevas alternativas que ayuden al proceso de toma de decisiones, la administración y control estadístico de cualquier empresa moderna.

En esta nueva era tecnológica, la acumulación de la información es común en la mayoría de las empresas. Con el objetivo de tomar decisiones factibles, se hace necesario realizar análisis exhaustivos, consultas de la información y obtención de los datos en el menor tiempo posible. Una forma de solucionar este problema es por medio del uso de la tecnología de *Business Intelligence*, por sus siglas en inglés (BI) o *Inteligencia de Negocios*, esta representa una alternativa tecnológica, que incluye arquitecturas para almacenar los datos, metodologías de desarrollo, técnicas para analizar la información, herramientas, entre otros; con la finalidad de apoyar el proceso de toma de decisiones centrándose en los datos valiosos de las empresas (2).

Las aplicaciones de BI son herramientas que proporcionan a los usuarios un mayor entendimiento que les permite identificar las oportunidades y los problemas de los negocios. Los usuarios son capaces de gestionar una gran cantidad de información, analizar sus relaciones y entender las tendencias que últimamente están apoyando las decisiones de los negocios. También son capaces de prevenir una potencial pérdida de conocimiento dentro de la empresa, que resulta de una acumulación masiva de información que no es fácil de leer o de usar. Existen diferentes herramientas de BI entre las que se encuentran Microsoft Business Intelligence, Microstrategy, Oracle, Qlikview y la Suite del Pentaho (2).

La Universidad de las Ciencias Informáticas (UCI), forma parte de las nuevas alternativas que se han llevado a cabo en Cuba, para lograr un avance vertiginoso en las Tecnologías de Información y las Comunicaciones (TIC). La misma cuenta con varios centros de desarrollo, uno de ellos es el Centro de Tecnologías de Gestión de Datos (DATEC), en el cual se desarrollan soluciones integrales de software con componentes de análisis de información utilizando herramientas de BI.

Una de las herramientas de BI que utiliza DATEC es el Pentaho BI Server en su versión comunitaria. Las funcionalidades que brinda esta versión son limitadas, pues no permite simplificar muchas tareas comunes como: las configuraciones de los diferentes mecanismos de autenticación se realizan de forma manual, por lo que se tienen que editar directamente los ficheros dentro de la aplicación. Una vez integrado el

Pentaho BI Server con un servidor de Protocolo Ligerito de Acceso a Directorios¹, por sus siglas en inglés (LDAP), los usuarios deben ser agregados de forma manual al esquema de seguridad. No se realiza la configuración del Servicio Central de Autenticación², por sus siglas en inglés (CAS) de manera automática. En la sección de seguridad no se realiza una validación eficiente de las contraseñas. No se realizan restricciones de una dirección IP específica o un rango de direcciones IP.

Todas estas limitaciones de la versión comunitaria trajeron consigo que el departamento se limitara a firmar contratos con algunos clientes, porque estos pedían solicitudes específicas de seguridad, que no podían ser atendidas por parte del grupo de desarrollo.

Por lo expuesto anteriormente el **problema de la investigación** queda definido en la siguiente interrogante: ¿Cómo automatizar la configuración de los diferentes mecanismos de autenticación del Pentaho BI Server en su versión comunitaria?

La presente investigación tiene como **objeto de estudio**: los mecanismos de autenticación de las herramientas de Inteligencia de Negocios, enmarcado en el **campo de acción**: los mecanismos de autenticación de la herramienta Pentaho BI Server.

Para darle solución al problema de la investigación se define como **objetivo general**: desarrollar una herramienta para la configuración de la seguridad del Pentaho BI Server en su versión comunitaria.

A partir del objetivo general se desglosaron los siguientes **objetivos específicos**:

- Elaborar el marco teórico para el desarrollo de la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.
- Realizar el análisis y diseño de la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.
- Implementar la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.

¹ Es un servicio de autenticación de usuario.

² Sistema de autenticación de aplicaciones.

- Realizar pruebas de software a la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.

Para dar cumplimiento a los objetivos planteados se trazaron las siguientes **tareas de la investigación**:

- Realización de un estudio bibliográfico relacionado con los mecanismos de autenticación en la herramienta Pentaho BI Server, permitiendo conocer el comportamiento de las tendencias actuales.
- Selección de las herramientas y tecnologías a utilizar en la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.
- Levantamiento de los requisitos, para conocer las necesidades del negocio.
- Definición de la arquitectura, para establecer el marco de trabajo de la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.
- Diseño de la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.
- Implementación de la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.
- Realización de las pruebas de software para validar las funcionalidades de la herramienta de configuración de la seguridad del Pentaho BI Server en su versión comunitaria.

Estructuración del trabajo de diploma

Para respaldar el cumplimiento de todos los elementos planteados anteriormente, el presente trabajo de diploma se ha estructurado en introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía y anexos. Para una mejor comprensión se presenta una breve descripción de cada capítulo:

En el **primer capítulo “Fundamentación teórica”**, se presentan los principales conceptos investigativos asociados al dominio del problema. También se seleccionan la metodología, herramientas y tecnologías a utilizar en el desarrollo de la solución.

En el **segundo capítulo “Descripción y análisis de la solución propuesta”**, se realiza todo lo vinculado con el flujo de trabajo análisis y diseño, así como el planteamiento de la solución que se quiere desarrollar. Se realiza la captura de los requisitos funcionales y no funcionales que el sistema debe cumplir, para lograr una mejor comprensión de la funcionalidad e integridad del sistema. También se realizan los diagramas de clases y los diagramas de interacción para los casos de uso arquitectónicamente significativos, el diagrama de despliegue para dar una visión de cómo debe quedar el despliegue de la solución, se explica la arquitectura utilizada y los principales patrones de diseño utilizados.

En el **tercer capítulo “Implementación y prueba”**, se realizan las actividades de implementación y prueba de la solución propuesta. Se realiza el diagrama de componentes para los casos de uso arquitectónicamente significativos. Se aplican niveles, métodos, técnicas y los casos de prueba relacionados con la implementación; para lograr el desarrollo de un software con la calidad requerida.

CAPÍTULO 1: FUNDAMENTOS TEÓRICOS

Introducción

En el presente capítulo se realiza una breve caracterización de las herramientas de BI; Microstrategy, Oracle y la Suite del Pentaho, haciendo énfasis en sus características, ventajas y desventajas. También se realiza un estudio bibliográfico de los principales mecanismos de autenticación en la herramienta Pentaho BI Server, permitiendo conocer el comportamiento de las tendencias actuales. Se explica brevemente la metodología seleccionada, así como el lenguaje de programación, las herramientas y tecnologías a utilizar en el desarrollo de la solución.

1.1 Inteligencia de Negocios

El término Inteligencia de Negocios se refiere al uso de datos en una empresa para facilitar la toma de decisiones. Abarca la comprensión del funcionamiento actual de la empresa, como la anticipación de acontecimientos futuros, con el objetivo de ofrecer conocimientos para respaldar las decisiones empresariales. Como lo define *Gartner Group*:³ "transformación de datos de una compañía en conocimiento para obtener una ventaja competitiva" (2).

Las aplicaciones de BI son herramientas de soporte de decisiones que permiten en tiempo real, acceso interactivo, análisis y manipulación de información crítica para la empresa. Estas aplicaciones proporcionan a los usuarios un mayor entendimiento que les permite identificar las oportunidades y los problemas de los negocios. Los usuarios son capaces gestionar una gran cantidad de información, analizar sus relaciones y entender las tendencias que últimamente están apoyando las decisiones de los negocios. También son capaces de prevenir una potencial pérdida de conocimiento dentro de la empresa, que resulta de una acumulación masiva de información que no es fácil de leer o de usar (2).

1.1.1 Herramientas de Inteligencia de Negocios

Las herramientas de BI se basan en la utilización de un sistema de información de inteligencia, que se forma con distintos datos extraídos del negocio de la empresa que se está analizando. Este conjunto de herramientas tienen en común las siguientes características (1):

³ Es una empresa consultora y de investigación de las tecnologías de la información.

- Accesibilidad a la información.
- Apoyo en la toma de decisiones.
- Orientación al usuario final.

A continuación se brinda una breve caracterización de algunas de las herramientas de BI, entre ellas se encuentra la Suite del Pentaho, que es la herramienta que utiliza el departamento Almacenes de Datos para desarrollar sus soluciones.

MicroStrategy

MicroStrategy es un proveedor de software OLAP⁴, que se caracteriza por brindarles las soluciones necesarias a los clientes. Les brinda un marco de ayuda que les permite acceder a un mayor conocimiento de la información que es manejada por la empresa (3).

Las principales ventajas que posee son:

- El desarrollo de aplicaciones Business Intelligence con MicroStrategy es relativamente sencillo, con una interfaz fácil de utilizar.
- Permite a los usuarios diseñar nuevos informes mediante una interfaz amigable.

Las principales desventajas que posee son:

- La implantación de la herramienta es costosa, tanto económicamente como en tiempo.
- Brinda precios moderados para las grandes empresas, pero se les hace imposible pagarlo a las empresas pequeñas.

Oracle

⁴ Es el procesamiento analítico en línea de la información.

Oracle es básicamente una herramienta cliente/servidor para la gestión de base de datos. Es un producto vendido a nivel mundial, la gran potencia que tiene y su elevado precio hace que solo se utilice en empresas muy grandes y multinacionales, por norma general (4).

Las principales ventajas que posee son:

- Gran velocidad de acceso a la información.
- Integra módulos de seguridad de acceso a la información, adaptables a cada perfil de usuario e incluso a cada usuario, con una interfaz muy intuitiva para su mantenimiento.
- Las aplicaciones desarrolladas son muy estables, incluso cuando los orígenes de datos tienen problemas de actualización de la información o han sido reestructurados.

Las principales desventajas que posee son:

- Orientada únicamente a grandes empresas debido a su precio.
- El tiempo de desarrollo puede tardar años para proyectos simples.
- En el aspecto técnico, las aplicaciones BI desarrolladas con esta herramienta son muy estáticas y su evolución al análisis de nuevas áreas de negocio complejo. Es difícil incluir nuevos orígenes de datos, lo que repercute en su reusabilidad futura.

Suite de Pentaho

La Suite de Pentaho es una plataforma de BI “orientada a la solución” y “centrada en procesos” que incluye todos los principales componentes requeridos para implementar soluciones basadas en procesos. La plataforma es capaz de ejecutar las reglas de negocio necesarias, expresadas en forma de procesos, actividades, de presentar y entregar la información adecuada en el momento adecuado. La plataforma fue desarrollada bajo el lenguaje de programación Java y tiene un ambiente de implementación también basado en este lenguaje, haciendo así que Pentaho sea una solución muy flexible al cubrir una alta gama de necesidades empresariales (1).

Las principales ventajas que posee son (1):

- Facilidad de uso.
- Arquitectura flexible y de calidad.
- Madurez de la tecnología.
- Integración rápida con la infraestructura existente.
- Bajo costo.
- Alto desempeño, estabilidad y escalabilidad.
- Facilita el camino para conseguir una completa solución de Inteligencia de negocios.
- Uso de tecnologías estándar: Java, XML, Java Script.

La Suite de Pentaho está integrada por varias herramientas que se utilizan para realizar las transformaciones de datos, además están destinadas a facilitar la toma de decisiones de las empresas. Entre las que se encuentran el Pentaho Reporting, Pentaho Analysis Services, Pentaho Dashboard, Pentaho Data Integration, Pentaho Data Mining y por último el Pentaho BI Server. A continuación se hace una breve descripción sobre las principales características y funcionalidades que presenta el Pentaho BI Server.

1.2 Pentaho BI Server

Pentaho BI Server ofrece dos versiones de su solución, entre las que se encuentran una versión comunitaria gratuita, orientada principalmente al mundo académico y una versión privativa, orientada a la implementación profesional. Esta herramienta funciona como un sistema de gestión basado en la web, está diseñada para integrarse fácilmente con cualquier proceso de negocio y proporciona una interfaz para la visualización de la información analizada, dígame en forma de gráfica o agrupada por un determinado atributo que presenta el negocio (3).

Dentro del Pentaho BI Server (ce) se encuentra la consola de administración que reduce los costos de operación al simplificar el despliegue, mejorar la fiabilidad, la facilidad de uso, el rendimiento y la escalabilidad de la solución (3).

Consola de administración en su versión privativa

La consola de administración del Pentaho BI Server en su versión privativa, es la encargada de gestionar todos los servicios de la herramienta relacionados con su seguridad, entre los que se encuentran: gestionar las diferentes formas de autenticación, la verificación de las conexiones y controlar la actividad remota (5).

Consola de administración en su versión comunitaria

Las funcionalidades de la consola de administración del Pentaho BI Server en su versión comunitaria son limitadas, pues no permite simplificar muchas tareas administrativas comunes como (5):

- Las configuraciones de los diferentes mecanismos de autenticación se realizan de forma manual, por lo que se tienen que editar directamente los ficheros dentro de la aplicación.
- Una vez integrado el Pentaho BI Server con un servidor LDAP, los usuarios deben ser agregados de forma manual al esquema de seguridad.
- No se realiza la configuración del CAS de manera automática.
- La gestión de permisos de usuarios, no se realiza de una forma óptima: la contraseña de los usuarios, no cuenta con una restricción mínima o máxima.
- No se realizan restricciones de una dirección IP específica o un rango de direcciones IP.

A continuación se muestra la tabla 1, donde se ofrece una breve explicación de cada una de las funcionalidades de las versiones de la consola de administración del Pentaho BI Server (3). Se marca con una X si la versión contiene esta funcionalidad.

Funcionalidades de la consola de administración	Función	Versión privativa	Versión comunitaria

Configuraciones de seguridad	Posee un directorio centralizado que permite la integración con directorios ya existentes.	X	
Diagnósticos de las aplicaciones	Son capaces de diagnosticar el estado de los servicios, posibilitando a los usuarios administradores de herramientas validar las configuraciones de sistemas, incluyendo el acceso a repositorios, configuraciones de sistema, e-mail y aplicaciones web.	X	
Utilidades de repositorios	Permite simplificar backups, migraciones y se utiliza para restaurar repositorios relevantes de información.	X	
Manejo del ciclo de vida de la aplicación	Permite a los administradores hacer cambios en un sistema de prueba o desarrollo, validar esos cambios, y luego implementar los cambios a los sistemas BI en producción. Además, permite que versiones anteriores puedan ser restauradas y/o auditadas en caso de error.	X	

<p>Reportes de auditoría</p>	<p>Los reportes de auditorías proveen un informe de las actividades de los usuarios en tiempo real, el rendimiento del sistema y las aplicaciones más demandadas. Además ayudan a los administradores a entender y optimizar sus aplicaciones de BI.</p>	<p>X</p>	
<p>Usuarios y roles</p>	<p>Permite definir los permisos que tendrá cada uno de los usuarios y los roles en el sistema BI. Las configuraciones de cuentas de los usuarios pueden ser rescatadas a partir de sistema o repositorios que ya estén presentes en la empresa, por lo que brinda que cada usuario tenga acceso a la información que corresponda de acuerdo a las políticas de seguridad de la información vigente.</p>	<p>X</p>	<p>X</p>

Tabla 1: Funcionalidades de las versiones de la consola de administración del Pentaho BI Server.

El Pentaho BI Server contiene diferentes mecanismos de autenticación que garantizan la seguridad de la herramienta, entre los cuales se encuentran los basados en un LDAP, las bases de datos y mediante la utilización del CAS. A continuación se brinda una breve explicación de cada uno de estos.

1.3 Mecanismos de autenticación

La autenticación es la verificación de la validez del usuario y su contraseña para iniciar una sesión (14). Existen cuatro tipos de técnicas que permiten realizar la autenticación de la identidad del usuario, las cuales pueden ser utilizadas individualmente o combinadas (autenticación de varios factores).

Servicio Central de Autenticación (CAS)

El Servicio Central de Autenticación (CAS), es un sistema de autenticación creado originalmente por la Universidad de Yale, ubicada en la ciudad de New Haven en el estado de Connecticut, con el objetivo de proporcionar un medio confiable a las aplicaciones para la autenticación de usuarios. Es un servicio que funciona por el protocolo HTTP, con el propósito de permitir al usuario tener acceso a múltiples aplicaciones, proporcionando un nombre de usuario y contraseña una sola vez (Single Sign On, por sus siglas en inglés (SSO)). El CAS está implementado como varios Servlets de Java, funciona a través del servidor HTTPS y se accede a través de tres direcciones URL: la URL de login, la URL de validación y la URL de logout. Además, utiliza los tickets⁵ como medio de autenticación, los cuales no pueden ser fácilmente falsificados ya que únicamente el servidor que los genera puede reconocerlos como válidos.

El proceso SSO, se realiza cuando el usuario se autentica al servidor CAS y solo necesita hacerlo una sola vez por sesión del navegador. Trabaja sólo con aplicaciones y recursos accedidos vía web, los accesos son interceptados con la ayuda de un servidor proxy o de un componente instalado en el servidor web destino. Brinda la posibilidad a las aplicaciones de autenticación por proxy a terceras capas de abastecedores de servicios que eligen aceptar sus credenciales de proxy (5) (6).

Tratamiento de la seguridad:

- Las contraseñas se envían siempre de los navegadores para el servidor de autenticación, a través de un túnel cifrado.
- Las autenticaciones son transparentes para los usuarios, siempre que se acepte una cookie⁶.

⁵ Ticket es un número de caracteres único e irrepitible, generado por el servidor CAS.

⁶ Es la información que guarda el servidor sobre la navegación del usuario.

Protocolo Ligero de Acceso a Directorios (LDAP)

Protocolo Ligero de Acceso a Directorios (LDAP), es un protocolo a nivel de aplicación basado en el estándar X.500, que permite el acceso a un servicio de directorio ordenado y distribuido para buscar diversa información en un entorno de red. LDAP también es considerado una base de datos a la que pueden realizarse consultas. Habitualmente almacena la información de autenticación (usuario y contraseña) y es posible almacenar otras informaciones como la ubicación de diversos recursos de la red, permisos, certificados, entre otros datos (7).

Características de un directorio LDAP:

- Es muy rápido en la lectura de registros.
- Permite replicar servidores de forma muy sencilla y económica.
- Usa un sistema jerárquico de almacenamiento de información.
- Permite múltiples directorios independientes.
- La mayoría de las aplicaciones disponen de soporte para LDAP.
- La mayoría de servidores LDAP son fáciles de instalar, mantener y optimizar.

Dadas las características de un LDAP sus usos más comunes son:

- Directorios de información.
- Sistemas de autenticación/autorización centralizada.
- Sistemas de Correo electrónico.
- Servidores de certificados públicos y llaves de seguridad.
- Autenticación única o SSO para la personalización de aplicaciones.
- Perfiles de usuarios centralizados.

- Libretas de direcciones compartidas.

1.4 Metodologías de desarrollo

Las metodologías tienen un papel fundamental en el desarrollo de un software, ya que el éxito y la calidad del producto dependen en gran parte de la metodología escogida, sea tradicional o ágil. Esta debe ser capaz de guiar y organizar las actividades que conlleven a cumplir las metas trazadas y que satisfaga más allá de las necesidades definidas al inicio del proyecto (8).

OpenUP

Constituye un marco de trabajo para procesos de desarrollo de software. Se construyó sobre una donación realizada por IBM del Basic Unified Process. Fue entregada a Eclipse a fines de 2005 y renombrado como OpenUP en 2006. OpenUP es un proceso de desarrollo unificado que está basado en Rational Unified Process por sus siglas en inglés (RUP). OpenUP mantiene las mismas características de la metodología RUP: desarrollo iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura. Sólo lo fundamental está incluido, sin dejar de ser completo y extensible (menos de 20 artefactos) (8).

OpenUP propone 5 disciplinas de trabajo:

- Requisitos: Se realizan entrevistas con el cliente para comprender el problema a resolver y se definen los requerimientos.
- Arquitectura: Se realiza el diseño de los requisitos que serán después implementados.
- Desarrollo: Se realiza la implementación del sistema basándose en el diseño realizado.
- Prueba: Busca los defectos a lo largo del ciclo de vida.
- Gestión del Proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.

OpenUP define como principales roles: analista, arquitecto, desarrollador, líder de proyecto, stakeholders y probador. El ciclo de vida de un proyecto según la metodología se divide en 4 fases fundamentales: concepción, elaboración, construcción y transición.

Justificación de la metodología propuesta

De acuerdo a la decisión del grupo de desarrollo del proyecto Desarrollo Tecnológico del departamento Almacenes de Datos del centro DATEC, se utilizará la metodología ágil OpenUP debido a que es considerada una metodología ligera de desarrollo de software, cuyo enfoque está centrado al cliente. Es gratis, ágil, modificable, se integra a otras metodologías ágiles y brinda una referencia clara y simplificada para la inducción de nuevo personal. Constituye un marco de trabajo genérico que puede ser adaptado a una gran diversidad de sistemas de software independientemente del tamaño del proyecto, el tipo de organización y los diferentes niveles de aptitud. Es una de las metodologías que se aplica en proyectos pequeños y de bajos recursos como el que se desea desarrollar, permitiendo disminuir las probabilidades del fracaso e incrementar las probabilidades del éxito. Además permite detectar errores tempranos a través de su ciclo de desarrollo iterativo.

1.5 Herramientas de modelado

Las herramientas CASE son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software, reduciendo el costo de las mismas en términos de tiempo y recurso. Las herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software, en tareas como el proceso de realizar un diseño del proyecto, cálculo de costos, compilación automática, documentación o detección de errores, entre otras (10).

Visual Paradigm

Visual Paradigm es una herramienta que sirve para realizar modelado de software siguiendo el estándar UML. Posee características gráficas muy cómodas, que facilitan la realización de diferentes diagramas de modelado. Además, puede ser utilizada para la modelación de procesos de desarrollo de software que sigan la filosofía de software libre, una de las razones que justifican su selección. Entre sus características principales se encuentran (13):

- Es profesional.
- Contiene facilidades para redactar especificaciones de casos de uso del sistema.
- Permite la sincronización entre diagramas de entidad-relación y diagramas de clases.

- Permite la generación de código / ingeniería inversa.
- Permite la generación de documentos.
- Permite la interoperabilidad con otras aplicaciones.
- Permite la integración con distintos Entornos de Desarrollo Integrados (IDEs), entre ellos NetBeans.

En la presente investigación se decidió utilizar la herramienta CASE Visual Paradigm en su versión 8.0 para el modelado del sistema a desarrollar, permitiendo modelar todos los diagramas necesarios para representar gráficamente el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue.

1.6 Sistema Gestor de Bases de Datos (SGBD)

Un Sistema Gestor de Base de Datos es un sistema de software que permite la definición de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación (9).

PostgreSQL

Es un sistema de gestión de base de datos relacional orientado a objetos, libre y es utilizado por todos aquellos programadores que realizan aplicaciones cliente servidor, complejas o críticas. Es capaz de ajustarse al número de microprocesadores y a la cantidad de memoria que posee el sistema de forma óptima, haciéndole posible soportar una mayor cantidad de peticiones simultáneas de manera correcta. Como es un sistema de gestión de base de datos relacional orientada a objetos, tiene numerosas ventajas entre las cuales están (11):

- Permite un modelado más natural de la realidad.
- Facilita la reutilización de componentes de software.
- Ofrece mecanismos de abstracción para mantener controlable la construcción de sistemas complejos.

Oracle

Oracle es uno de los servidores de base de datos más conocido y ha alcanzado un buen nivel de madurez y de profesionalidad. Se destaca por su soporte de transacciones, estabilidad y escalabilidad. Los tipos objeto de Oracle son tipos de datos definidos por el usuario, que permiten modelar entidades complejas del mundo real, en una estructura que trata cada entidad como una unidad atómica simple en la base de datos (15).

Características:

- Es intuitiva y cómoda de utilizar.
- Ayuda a analizar datos y efectuar recomendaciones concernientes a mejorar el rendimiento y la eficiencia en el manejo de aquellos datos que se encuentran almacenados.
- Apoya en el diseño y optimización de modelos de datos.
- Apoya en la definición de estándares de diseño y nomenclatura de objetos.

MySQL

MySQL es un sistema de gestión de base de datos relacional y multiusuario con más de seis millones de instalaciones. Es idóneo para la creación de bases de datos con acceso desde páginas web dinámicas, así como para la creación de cualquier otra solución que implique el almacenamiento de datos, posibilitando realizar múltiples y rápidas consultas. Está desarrollado en C y C++, facilitando su integración en otras aplicaciones desarrolladas en esos lenguajes. Se considera uno de los sistemas gestores de bases de datos más utilizado en la actualidad, empleado por grandes corporaciones como Yahoo! Finance, Google, Motorola, entre otras (9).

Características:

- Es rápido, sólido y flexible.
- Posibilita realizar múltiples y rápidas consultas.

- Es un sistema cliente/servidor, por lo que permite trabajar como servidor multiusuario y de subprocesamiento múltiple.

Se seleccionó para trabajar en el desarrollo de la solución los SGBD PostgreSQL, Oracle y MySQL porque son los que emplea el Pentaho BI Server en su configuración; permitiendo gestionar la información que utiliza cuando se conecta con algunos de estos gestores.

1.7 Lenguajes de programación

Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Una característica relevante de los lenguajes de programación, es precisamente que más de un programador pueda usar un conjunto común de instrucciones que sean comprendidas entre ellos, para así realizar la construcción de un programa de forma colaborativa (16).

Java web

Java es un lenguaje de programación orientado a objetos, desarrollado por Sun Microsystems a principios de los años 90. El lenguaje en sí mismo toma mucha de su sintaxis de C y C++, pero tiene un modelo de objetos más simple y elimina herramientas de bajo nivel, que suelen inducir a muchos errores, como la manipulación directa de punteros o memoria. Con respecto a la memoria, su gestión no es un problema, ya que esta es gestionada por el propio lenguaje y no por el programador.

Algunas características del lenguaje se describen a continuación (16):

- *Orientado a Objeto*: Java implementa la tecnología básica de C++ con algunas mejoras y elimina algunas cosas para mantener el objetivo de la simplicidad del lenguaje. Soporta las tres características propias del paradigma de la orientación a objetos: encapsulación, herencia y polimorfismo.
- *Distribuido*: Java se ha construido con extensas capacidades de interconexión TCP/IP. La característica de ser distribuido es debido a que proporciona las librerías y herramientas para que los programas puedan ejecutarse en varias máquinas.

- *Robusto:* Java realiza verificaciones en busca de problemas tanto en tiempo de compilación como en tiempo de ejecución. La comprobación de tipos en Java ayuda a detectar errores lo antes posible en el ciclo de desarrollo.
- *Interpretado:* Se traduce el código fuente a un código intermedio denominado bytecode, que es interpretado por la máquina virtual de Java, lo cual permite que se pueda ejecutar en cualquier sistema operativo.
- *Seguro:* No se permite el acceso ilegal a memoria ya que no se trabaja con punteros. El código Java pasa muchas pruebas antes de ejecutarse en una máquina. El código se pasa a través de un verificador de bytecode que comprueba el formato de los fragmentos de código y aplica un probador de teoremas para detectar fragmentos de código ilegal, que falsean punteros, violan derechos de acceso sobre objetos o intentan cambiar el tipo o clase de un objeto.
- *Dinámico:* No conecta todos los módulos que comprende una aplicación hasta el tiempo de ejecución, ya que las librerías nuevas o actualizadas no paralizarán las aplicaciones actuales siempre que mantengan el API anterior.

Justificación del lenguaje de programación propuesto

Se seleccionó para trabajar en el desarrollo de la solución el lenguaje Java, porque es multiplataforma, garantiza la seguridad de la herramienta a desarrollar al no trabajar con punteros, proporciona librerías y herramientas para acceder a las bases de datos y es en el que está implementado el Pentaho BI Server.

1.8 Framework de desarrollo Spring MVC

Uno de los marcos de trabajo más populares en la comunidad de Java es Spring Framework, este incluye varias librerías que ayudan a solventar problemas comunes en el desarrollo de aplicaciones. Spring soporta la Programación Orientada a Aspecto, implementa el patrón MVC, separando la interfaz de usuario, los controladores y la capa de acceso a datos y utiliza el Objeto de Acceso a Datos (DAO, siglas en inglés de Data Access Object), que es un patrón de diseño para gestionar las bases de datos (17). También permite el trabajo con herramientas ORM como Hibernate y con motores de plantillas para interfaces de usuarios como Tiles y Velocity.

1.9 Entorno de Desarrollo Integrado (IDE)

Un Entorno de Desarrollo Integrado o IDE (acrónimo en inglés de Integrated Development Environment), es un programa informático compuesto por un conjunto de herramientas de programación. Consiste básicamente en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Provee un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, Python, Java, C#, Delphi, Visual Basic, entre otros. Sin un IDE es imposible editar el código expresado a través del lenguaje de programación, por lo que su uso se hace obligatorio (18).

Eclipse

Eclipse es un entorno de desarrollo integral de código abierto, multiplataforma, creado originalmente por IBM. Adicionalmente permite el uso de otros lenguajes de programación como son JAVA, C/C++ y Python, también permite trabajar con lenguajes para procesado de texto como Látex, aplicaciones en red como Telnet y Sistemas de Gestión de Base de Datos. Emplea módulos para proporcionar toda su funcionalidad al frente de la plataforma de cliente enriquecido, este mecanismo de módulos es una plataforma ligera para componentes de software. Dispone de un Editor de texto con resaltado de sintaxis, con pruebas unitarias con JUnit, control de versiones con CVS, integración con Ant, asistentes (wizards) para creación de proyectos, clases, tests y refactorización. Este IDE fue liberado originalmente bajo la licencia Common Public License, pero después se liberó bajo la licencia Eclipse Public License. La Free Software Foundation ha dicho que ambas licencias son licencias de software libre, pero son incompatibles con Licencia Pública General de GNU (GNU GPL) (18).

Ventajas del uso del Eclipse:

- El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos para proporcionar toda su funcionalidad al frente de la plataforma del cliente, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.
- Eclipse provee al programador los frameworks para el desarrollo de aplicaciones gráficas, definición y manipulación de modelos de software, aplicaciones web, etc.

- El SDK de Eclipse incluye las herramientas de desarrollo de Java, ofreciendo un IDE con un compilador de Java interno y un modelo completo de los archivos fuente de Java. Esto permite técnicas avanzadas de refactorización y análisis de código.

Justificación del Entorno de Desarrollo Integrado (IDE) propuesto

Se seleccionó para trabajar en el desarrollo de la solución el entorno de desarrollo Eclipse en su versión 3.0, porque soporta Java como lenguaje multiplataforma, además de introducir aplicaciones accesorias que pueden resultar útiles durante el proceso de desarrollo, como herramientas UML, editores visuales de interfaces, entre otros. Además brinda un conjunto de librerías para el framework de desarrollo que se utilizará Spring MVC y es el entorno de desarrollo que el equipo de desarrollo tiene más dominio.

Conclusiones del capítulo

En el capítulo se realizó un estudio bibliográfico de los principales mecanismos de autenticación en la herramienta Pentaho BI Server, permitiendo conocer el comportamiento de las tendencias actuales. También se realizó un análisis de los elementos fundamentales de las herramientas y tecnologías seleccionadas a utilizar en el desarrollo de la solución, arrojando como resultado las características claves que servirán de base para el diseño de la investigación. De acuerdo a la decisión del grupo de desarrollo del proyecto Desarrollo Tecnológico del departamento Almacenes de Datos del centro DATEC, se seleccionó la metodología OpenUP como guía para la documentación junto a la herramienta Visual Paradigm. Como lenguaje de programación se escogió Java, como IDE de desarrollo Eclipse 3.0 con el framework Spring MVC, que implementa el estilo arquitectónico Modelo-Vista-Controlador. Se seleccionó para trabajar en el desarrollo de la solución los SGBD PostgreSQL, Oracle y MySQL porque son los que emplea el Pentaho BI Server en su configuración.

CAPÍTULO 2: DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA

Introducción

En el presente capítulo se describirá la propuesta de solución para el problema científico de la investigación, mediante conceptos y pautas. A partir de comprender el negocio, se obtienen los requisitos funcionales y no funcionales. La traducción de los mismos a una especificación que describe cómo implementar la herramienta, permite realizar el diseño de la aplicación generando los artefactos necesarios para cada fase. Se realizan los diagramas de clases y los de interacción para los casos de uso arquitectónicamente significativos, terminando la etapa de diseño de la herramienta con la distribución física de los nodos en la red, que se propone mediante el modelo de despliegue de la solución.

2.1 Propuesta de solución

La siguiente propuesta de solución se elaboró con el objetivo de brindar un mejor entendimiento a los usuarios interesados en el resultado de la herramienta.

En la figura 1 se muestra el Pentaho BI Server con las conexiones y funcionalidades que va a ser capaz de brindar, después de incorporarle la herramienta que sea implementar.

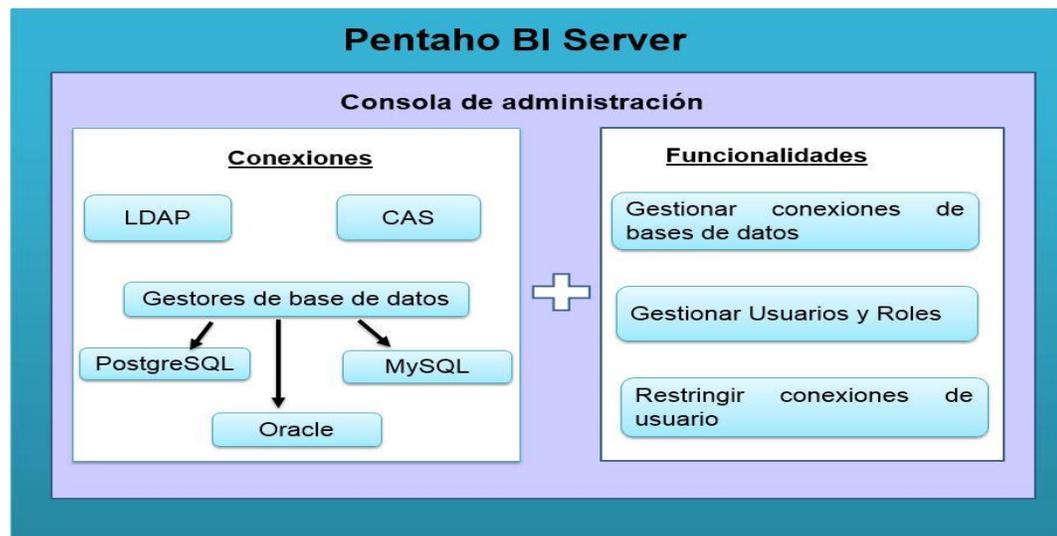


Figura 1: Propuesta de la solución.

2.2 Requisitos

Un requisito es una condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo, define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen (4).

2.2.1 Requisitos funcionales

Los requisitos funcionales son capacidades o condiciones que un sistema determinado debe cumplir (4). Expresan una especificación detallada de las responsabilidades del sistema en cuestión y permiten determinar, de una manera clara, lo que debe hacer el sistema. Para la herramienta se determinaron 10 requisitos funcionales los cuales se definen a continuación:

RF1 Autenticar usuario.

RF2 Configurar gestores de base de datos.

RF2.1 Configurar para base de datos en PostgreSQL.

RF2.2 Configurar para base de datos en MySQL.

RF2.3 Configurar para base de datos en Oracle.

RF3 Integrar mecanismos de acceso con LDAP.

RF3.1 Validar la conexión a la base de datos.

RF4 Restringir conexiones de usuario.

RF4.1 Restringir conexión por una dirección IP específica.

RF4.2 Restringir conexión por un rango de direcciones IP determinado.

RF5 Gestionar usuario.

RF5.1 Listar usuarios.

RF5.2 Adicionar usuario.

RF5.3 Modificar los datos de un usuario.

RF5.4 Buscar los datos de un usuario.

RF5.5 Eliminar usuario.

RF6 Gestionar rol.

RF6.1 Listar roles.

RF6.2 Adicionar rol.

RF 6.3 Modificar los datos de un rol.

RF 6.4 Buscar los datos de un rol.

RF 6.5 Eliminar rol.

RF7 Administrar rol de un usuario.

RF7.1 Adicionar rol a un usuario.

RF 7.2 Eliminar rol a un usuario.

RF 8 Administrar usuarios de un rol.

RF 8.1 Adicionar un usuario a un rol.

RF 8.2 Eliminar un usuario a un rol.

RF9 Configurar la conexión con el CAS.

RF10 Gestionar conexiones de base de datos.

RF10.1 Listar base de datos.

RF10.2 Adicionar base de datos.

RF10.3 Modificar los datos de una base de datos.

RF 10.4 Buscar los datos de una base de datos.

RF 10.5 Eliminar base de datos.

2.2.2 Requisitos no funcionales

Los requisitos no funcionales son propiedades o cualidades que el producto debe tener (4). Estas propiedades se describen como las características que hacen al producto atractivo, usable, rápido, confiable o eficiente con el hardware y software, con vista a las funcionalidades que el sistema debe brindar. Se identificaron a raíz de estos elementos los siguientes requisitos no funcionales:

Portabilidad

RNF.1 Potabilidad

El sistema será multiplataforma, razón por la cual podrá ser utilizado en cualquier sistema operativo y arquitectura de hardware.

Fiabilidad

RNF.2 Disponibilidad

El sistema debe poder permanecer constantemente funcionando excepto cuando sea necesario reiniciarlo o detenerlo por tareas de mantenimiento o cambio de configuración.

Eficiencia

RNF.3 Requisito de Eficiencia

La eficiencia del sistema depende en gran medida de la velocidad de conexión entre la base de datos y la herramienta. El sistema debe mantener tiempos de respuestas en un marco razonable de tiempo (600 m/seg a 5 seg).

Restricciones de diseño

RNF.4 Restricciones del diseño e implementación

El sistema deberá ser implementado en el lenguaje de programación Java, como framework de desarrollo se usará Spring MVC. Como herramienta de desarrollo se usará Eclipse en su versión 3.0 y como sistemas gestores de base de datos se utilizarán PostgreSQL 9.1, MySQL 4.5.0.9 y Oracle 10g.

Soporte

RNF.5 Instalación y capacitación

Se garantizará la instalación del sistema y se impartirá capacitaciones a los usuarios. Además, se contará con un manual de usuario y un manual de instalación que guiarán paso a paso las acciones de los usuarios. También, se le dará seguimiento al funcionamiento del sistema durante un período de tiempo de un mes.

Seguridad

RNF.6 Seguridad de la herramienta

El sistema debe ser capaz de garantizar los siguientes aspectos:

- Solo deben autenticarse los usuarios que tengan un nivel elevado de fortaleza en las contraseñas atendiendo a su longitud y combinaciones de caracteres especiales, mayúsculas y minúsculas.
- La protección contra acciones no autorizadas que puedan afectar la integridad de los datos, ejemplos:
 - Acceso al sistema de un usuario no autorizado.
 - Realizar cambios en la interfaz desde la aplicación.
 - Consultar los archivos de configuración del sistema desde la aplicación.
- Las contraseñas que se almacenen en la base de datos, se deben enviar encriptadas a su destino utilizando el algoritmo MD5.

RNF.7 Software

El servidor donde se instalará la herramienta debe cumplir con los siguientes requisitos de software:

- Sistema Operativo: Multiplataforma.

Los servidores donde se instalarán las bases de datos deben cumplir con los siguientes requisitos:

- Sistema Operativo: Multiplataforma.
- Se utilizarán los gestores de base de datos PostgreSQL 9.1, MySQL 4.5.0.9 y Oracle 10g.
- Los sistemas gestores de base de datos PostgreSQL, MySQL y Oracle deben estar correctamente configurado usando el driver jdbc.
- Herramientas de administración PgAdmin III 1.1.4, EMS SQL Manager 4.5.0.9 o superior, OracleXE 10g.

RNF.8 Hardware

El servidor donde se instalará la herramienta debe cumplir con los siguientes requisitos:

- Procesador Intel Pentium 4 a 1.6 GHz o superior.
- Al menos 512 GB de RAM.
- Al menos 150 MB de espacio en disco duro.

Los servidores donde se instalarán los SGBD PostgreSQL 9.1 y MySQL 4.5.0.9 deben cumplir con los siguientes requisitos:

- Procesador Intel Pentium 4 a 1.6 GHz o superior.
- Al menos 1 GB de memoria RAM.
- Al menos 10 GB de espacio en disco duro.

El servidor donde se instalará el SGBD Oracle debe cumplir con los siguientes requisitos:

- Procesador Intel Pentium 4 a 1.6 GHz o superior.
- Al menos 2 GB de RAM.
- Al menos 5 GB de espacio en disco duro.

La máquina utilizada para acceder a la herramienta debe cumplir con los siguientes requisitos:

- Procesador Intel Pentium 4 1.6 GHz o superior.
- 512 MB de RAM o superior.

2.3 Diagrama de casos de uso del sistema

El diagrama de casos de uso del sistema representa gráficamente a los procesos y su interacción con los actores (4). A partir de los requisitos funcionales identificados anteriormente se modeló el siguiente diagrama de casos de uso:

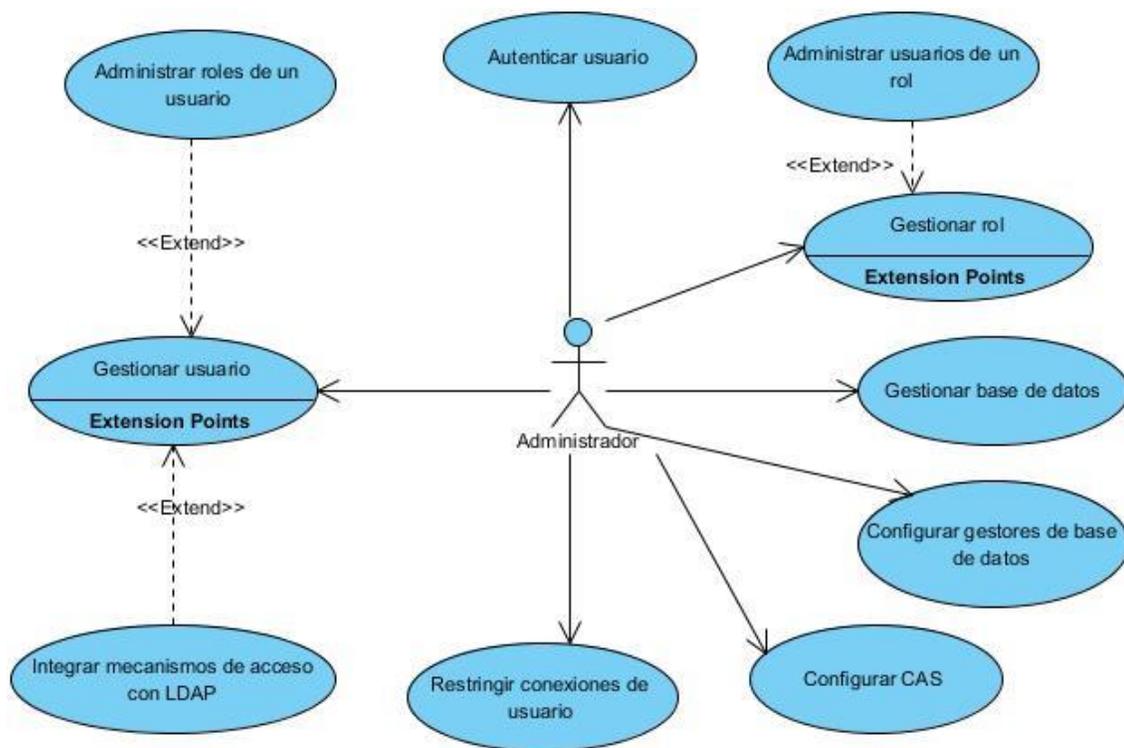


Figura 2: Diagrama de casos de uso del sistema.

Descripción del actor del sistema

Actor	Objetivo
Administrador	Usuario responsable de administrar los temas de seguridad del

	sistema gestionando la información de los usuarios que interactuarán con el sistema. Además tiene la posibilidad de restringir conexiones IP, de integrar los mecanismos de acceso con LDAP y CAS, así como configurar los gestores de base de datos.
--	---

Tabla 2: Descripción del actor del sistema.

Descripción de un caso de uso significativo

Caso de uso	Configurar gestores de base de datos
Actores	Administrador
Resumen	<p>El caso de uso se inicia cuando el Administrador decide realizar una de las siguientes acciones:</p> <ul style="list-style-type: none">– Configurar para base de datos en PostgreSQL: el administrador debe configurar la conexión del sistema al gestor de base de datos PostgreSQL, el sistema muestra un formulario para introducir los datos necesarios para la realización de la conexión, terminándose así el caso de uso.– Configurar para base de datos en MySQL: el administrador debe configurar la conexión del sistema al gestor de base de datos en MySQL, el sistema muestra un formulario para introducir los datos necesarios para la realización de la conexión, terminándose así el caso de uso.– Configurar para base de datos en Oracle: el

Capítulo 2: Descripción y Análisis de la solución propuesta

	administrador debe configurar la conexión del sistema al gestor de base de datos en Oracle, el sistema muestra un formulario para introducir los datos necesarios para la realización de la conexión, terminándose así el caso de uso.	
Complejidad	Alta	
Prioridad	Crítico	
Precondiciones	El usuario debe estar autenticado como administrador en el sistema.	
Postcondiciones	Se establecerá la conexión con la base de datos seleccionada.	
Flujo normal de eventos		
	Actor	Sistema
	<p>1. El administrador selecciona a través de un menú contextual la opción que desee realizar:</p> <ul style="list-style-type: none">– Configurar para base de datos en PostgreSQL.– Configurar para base de datos en MySQL.– Configurar para base de datos en Oracle.	<p>2. El sistema en dependencia de la opción seleccionada por el usuario, muestra la interfaz correspondiente:</p> <ul style="list-style-type: none">– Configurar para base de datos en PostgreSQL, ir a la sección “Configurar para base de datos en PostgreSQL”.– Configurar para base de datos en MySQL, ir a la sección “Configurar para base de datos en MySQL”.

		<ul style="list-style-type: none"> – Configurar para base de datos en Oracle, ir a la sección “Configurar para base de datos en Oracle”.
Sección “Configurar para base de datos en PostgreSQL”		
	Acción del actor	Respuesta del sistema
		1. El sistema muestra un formulario para especificar URL, nombre del usuario, contraseña y driver.
	2. El administrador introduce los datos de conexión y selecciona el botón aceptar.	<p>3. El sistema verifica que los campos de la URL, del nombre del usuario, la contraseña, el driver y el nombre de la conexión estén llenos.</p> <p>4. El sistema valida los datos y si estos son correctos, establece una nueva conexión, terminándose así el caso de uso.</p>
Flujos alternos		
3.1. Existen campos vacíos		
	Acción del actor	Respuesta del sistema
		Se emite un mensaje para que llene los campos obligatorios y regresa a la acción

		2 de esta sección.
4.1 Los datos son incorrectos		
	Acción del actor	Respuesta del sistema
		Se emite un mensaje para que revise los datos introducidos y regresa a la acción 2 de esta sección.

Sección "Configurar para base de datos en MySQL"		
	Acción del actor	Respuesta del sistema
		1. El sistema muestra un formulario para especificar URL, nombre del usuario, contraseña y driver.
	2. El administrador introduce los datos de conexión y selecciona el botón aceptar.	3. El sistema verifica que los campos de la URL, del nombre del usuario, la contraseña, el driver y el nombre de la conexión estén llenos. 4. El sistema valida los datos y si estos son correctos, establece una nueva conexión, terminándose así el caso de uso.
Flujos alternos		
3.1. Existen campos vacíos		

Capítulo 2: Descripción y Análisis de la solución propuesta

	Acción del actor	Respuesta del sistema
		Se emite un mensaje para que llene los campos obligatorios y regresa a la acción 2 de esta sección.
4.1. Los datos son incorrectos		
	Acción del actor	Respuesta del sistema
		Se emite un mensaje para que revise los datos introducidos y regresa a la acción 2 de esta sección.
Sección “Configurar para base de datos en Oracle”		
	Acción del actor	Respuesta del sistema
1.		1. El sistema muestra un formulario para especificar URL, el nombre del usuario, la contraseña y el driver.
1	2. El administrador introduce los datos de conexión y selecciona el botón aceptar.	3. El sistema verifica que los campos de la URL, del nombre del usuario, la contraseña, el driver y el nombre de la conexión estén llenos. 4. El sistema valida los datos y si estos son correctos, establece una nueva conexión, terminándose así el caso de

		uso.
Flujos alternos		
3.1. Existen campos vacíos		
	Acción del actor	Respuesta del sistema
		Se emite un mensaje para que llene los campos obligatorios y regresa a la acción 2 de esta sección.
4.1. Los datos son incorrectos		
	Acción del actor	Respuesta del sistema
		Se emite un mensaje para que revise los datos introducidos y regresa a la acción 2 de esta sección.

Tabla 3: Configurar gestores de base de datos.

Las demás descripciones de los casos de uso arquitectónicamente significativos se encuentran en los Anexos.

2.4 Diagrama de clases del diseño

El diagrama de clases del diseño muestra los atributos y métodos de cada clase y sus relaciones, también representa la parte estática del sistema a conformar (4). A partir de la descripción detallada de los casos de uso del sistema, se modeló el siguiente diagrama de clases del diseño:

Capítulo 2: Descripción y Análisis de la solución propuesta

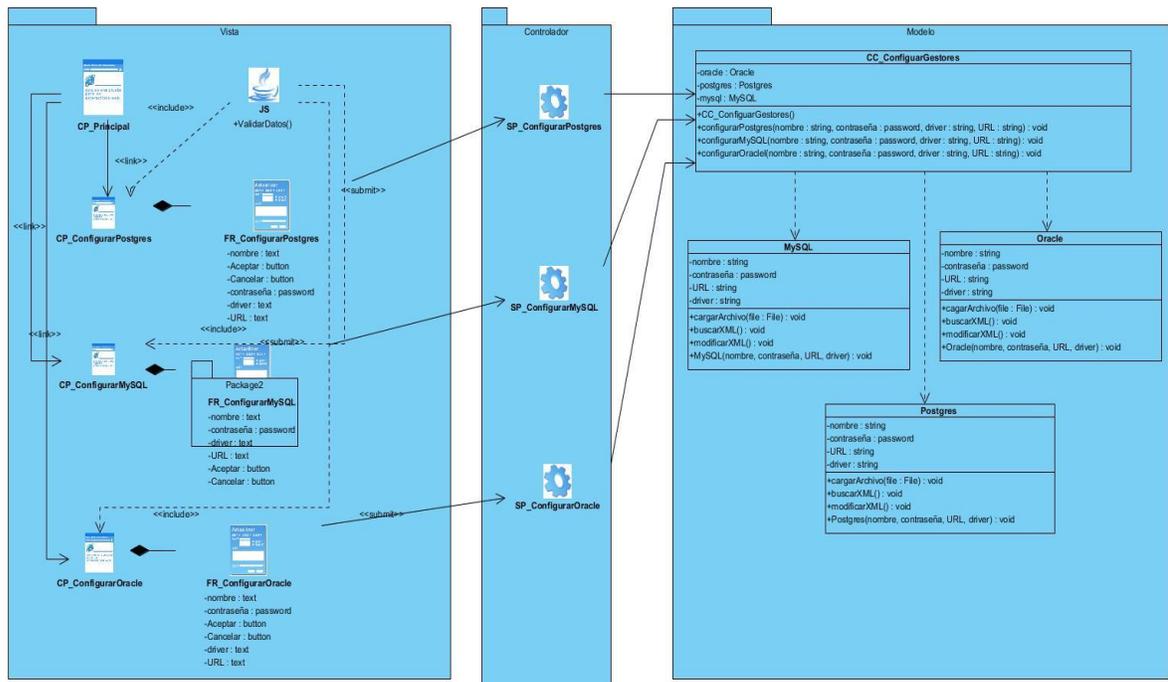


Figura 3: Diagrama de clases del diseño para el caso de uso “Configurar gestores de base de datos”. Los demás diagramas de clases del diseño arquitectónicamente significativos se encuentran en los Anexos.

Descripciones de las principales clases.

Nombre de la clase	Tipo de clase	Descripción
CC_ConfigurarGestores	Controladora	Es la clase encargada de controlar toda la información que recibe de las clases SP_ConfigurarPostgres, SP_ConfigurarOracle, SP_ConfigurarMySQL.
Nombre de métodos		Descripción

+ ConfigurarPostgres()	Permite configurar el Pentaho BI Server para que se conecte a una base de datos en PostgreSQL.
+ ConfigurarOracle()	Permite configurar el Pentaho BI Server para que se conecte a una base de datos en Oracle.
+ ConfigurarMySQL()	Permite configurar el Pentaho BI Server para que se conecte a una base de datos en MySQL.

Tabla 4: Diagrama de clases del diseño para el caso de uso “Configurar gestores de base de datos”.

Nombre de la clase	Tipo de clase	Descripción
Postgres	Entidad	Es la clase que permite la conexión del Pentaho BI Server con una base de datos en PostgreSQL.
Oracle	Entidad	Es la clase que permite la conexión del Pentaho BI Server con una base de datos en Oracle.
MySQL	Entidad	Es la clase que permite la conexión del Pentaho BI Server con una base de datos en MySQL.
Nombre de métodos		Descripción

+ cargarArchivo()	Permite cargar los archivos necesarios para la configuración del Pentaho BI Server.
+ buscarXML()	Permite la búsqueda de los archivos que se guardan en el sistema.
+ modificarXML()	Permite modificar los archivos encontrados por el método +buscarXML .

Tabla 5: Descripciones de las clases Postgres, Oracle y MySQL.

2.5 Patrones

Para hacer un diseño eficiente se tomaron en cuenta un conjunto de patrones que sirven como guía para resolver problemas comunes que surgen con frecuencia en la programación.

2.5.1 Patrones arquitectónicos

Entre patrones arquitectónicos más aplicados para el diseño web se encuentra el Modelo-Vista-Controlador (MVC) el cual es utilizado en la presente solución. El MVC es un patrón de arquitectura de software que permite separar los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos (4). A continuación se muestra el diseño de la arquitectura del patrón arquitectónico MVC.

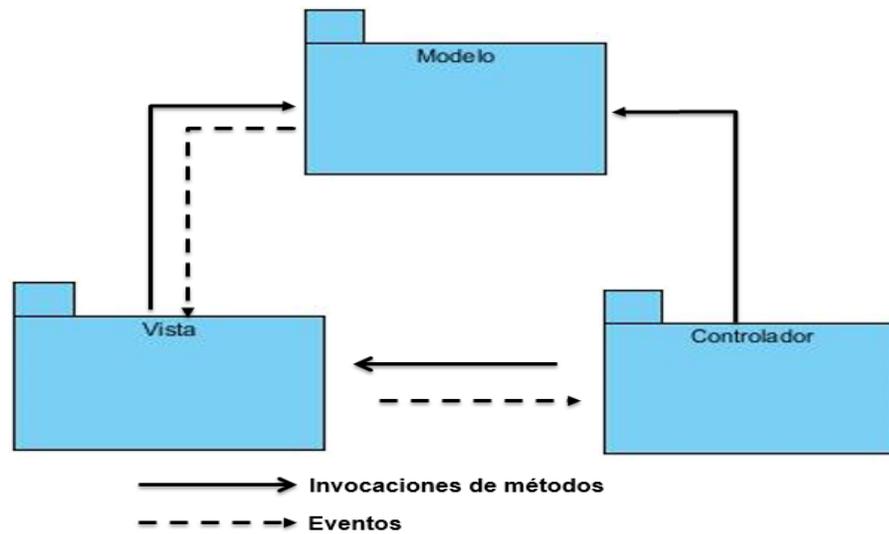


Figura 4: Arquitectura del patrón arquitectónico Modelo-Vista-Controlador.

El patrón arquitectónico Modelo-Vista-Controlador (MVC) funciona de la siguiente manera (4):

1. El controlador recibe una petición.
2. El controlador analiza la petición y usa un modelo para obtener datos de algún tipo.
3. El modelo puede leer datos y/o escribirlos en una base de datos.
4. El controlador usa una vista para poder presentar los resultados al usuario.
5. La vista usa el modelo, para visualizar los datos de la petición realizada.
6. La vista está renderizada con los datos del modelo y el controlador envía esta respuesta al usuario.
7. El usuario obtiene la respuesta.

2.5.2 Patrones de diseño

Los patrones de diseño empleados en la solución pertenecen al conjunto de patrones GRASP, su utilización ayudó a refinar el diseño y a asignar las responsabilidades de las distintas clases, haciéndolas más sencillas, reutilizables y encapsuladas. A continuación se explican los patrones utilizados (4):

Capítulo 2: Descripción y Análisis de la solución propuesta

Experto: El patrón experto en información es el principio básico de asignación de responsabilidades. Indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo.

En el diseño de la solución la clase `CC_ConfigurarGestores` al contener toda la información de la clase `Postgres`, `MySQL` y `Oracle` es la que tiene la responsabilidad de invocar los métodos de estas clases, así como la creación de objetos. Esta es precisamente la solución que propone el patrón Experto, pues cada clase cuenta con un conjunto de funcionalidades relacionadas directamente con la entidad que representa, como se muestra en la figura 5.

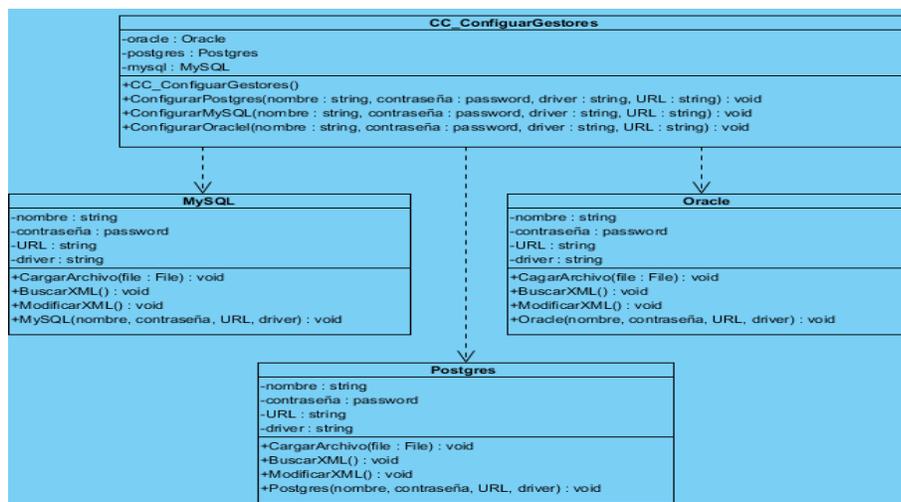


Figura 5: Patrón GRASP Experto.

Creador: El patrón creador es el principio básico de la responsabilidad de la creación de una nueva instancia de una clase. Posibilita un diseño bien asignado, que puede soportar un bajo acoplamiento, el encapsulamiento y la reutilización.

En el diseño de la solución la clase `CC_ConfigurarGestores` se crean las instancias que van a comunicarse con las clases `Postgres`, `MySQL` y `Oracle`, evidenciando de este modo que la clase `CC_ConfigurarGestores` representa el patrón "creador" de dichas entidades, como se muestra en la figura 6.

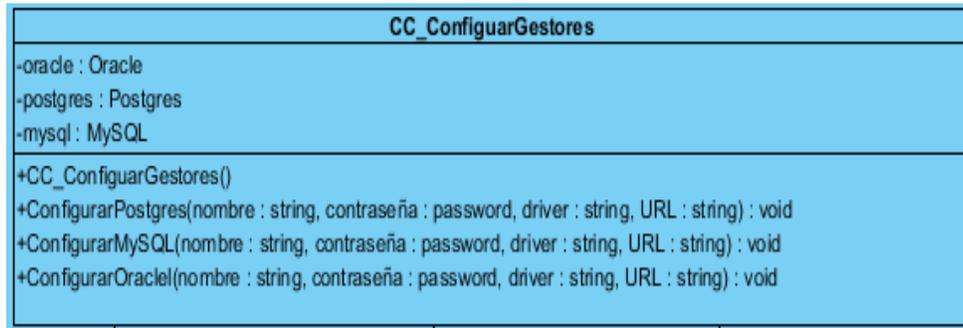


Figura 6: Patrón GRASP Experto.

Alta Cohesión: El patrón Alta Cohesión es el principio básico de mantener la complejidad dentro de los límites manejables. Una clase con alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas, que no realicen un trabajo enorme. Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo. Por lo que indica asignar la responsabilidad de modo que la cohesión siga siendo alta.

En el diseño de la solución se encuentran varias clases Server Pages, donde cada una de ellas es responsable de controlar la configuración de toda la información que gestione una determinada clase, evidenciando de este modo que las clases Server Pages representan el patrón "alta cohesión", como se muestra en la figura 7.

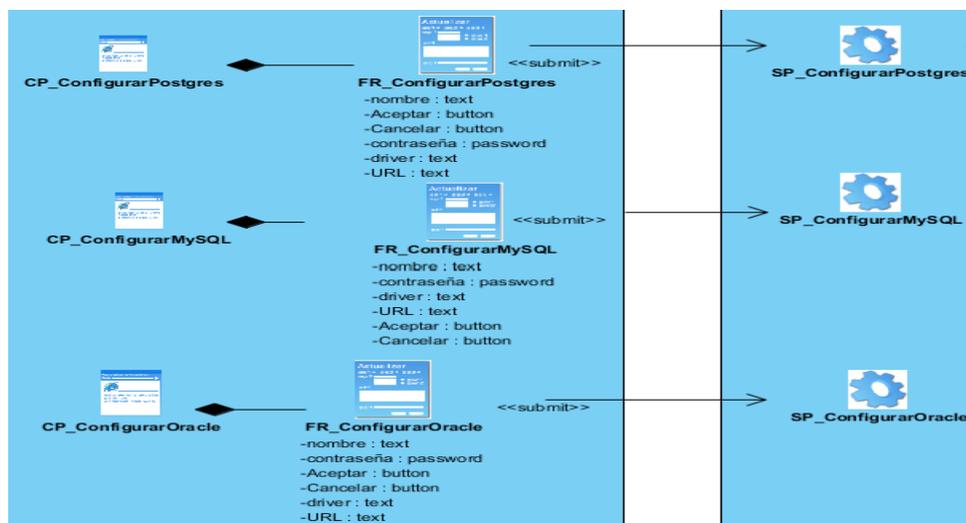


Figura 7: Patrón GRASP Alta Cohesión.

Controlador. El patrón Controlador es el principio básico de la responsabilidad de atender un evento del sistema. Un evento del sistema es un evento de alto nivel generado por un actor externo, es un evento de entrada externa.

Un controlador es un objeto de interfaz no destinado al usuario, que se encarga de manejar un evento del sistema y define además el método de su operación.

En el diseño de la solución se encuentra la clase `CC_ConfigurarGestores`, que es la encargada de controlar toda la información recibida de las clases `SP_ConfigurarPostgres`, `SP_ConfigurarOracle`, `SP_ConfigurarMySQL`, evidenciando de este modo que la clase controladora `CC_ConfigurarGestores` representa el patrón "controlador" de dichas entidades, como se muestra en la figura 8.

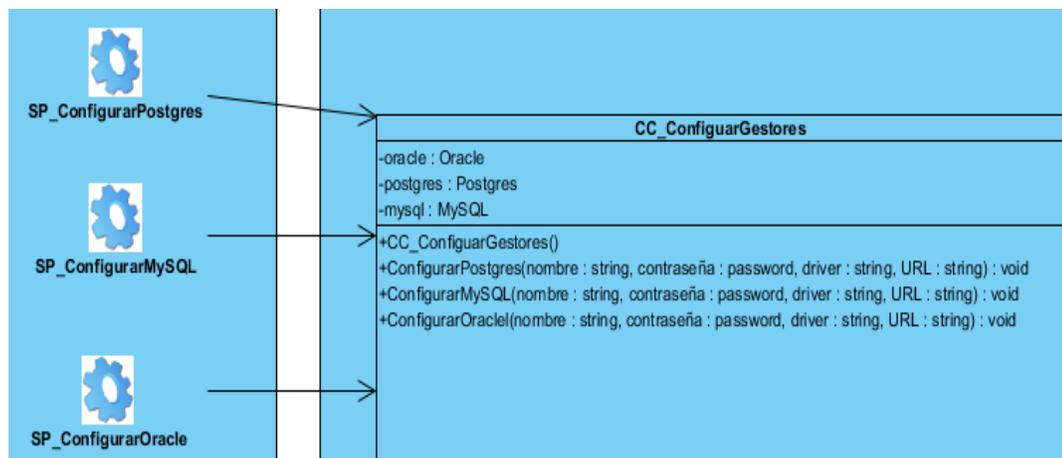


Figura 8: Patrón GRASP Controlador.

2.6 Diagrama de secuencia

El diagrama de secuencia muestra las interacciones entre un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Los diagramas de secuencia se utilizan para modelar los aspectos dinámicos de un sistema. La mayoría de las veces, esto implica modelar instancias concretas o prototípicas de clases, interfaces, componentes y nodos; todo en el contexto de un escenario que ilustre un comportamiento (4). A continuación se muestran los diagramas de secuencias de los escenarios del caso de uso Configurar gestores de base de datos.

Sección ConfigurarPostgres:

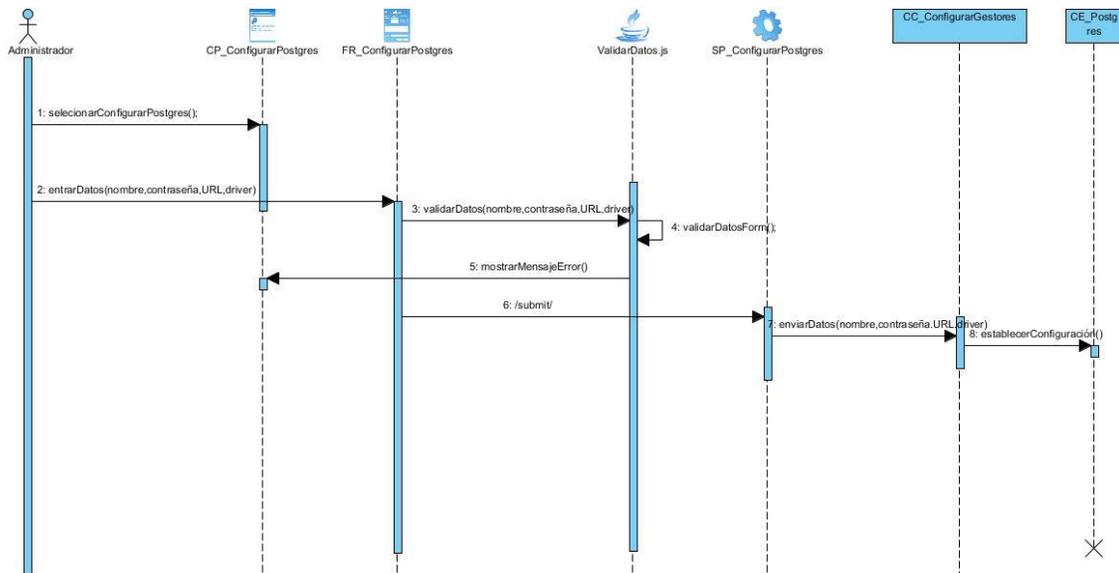


Figura 9: Diagrama de secuencia de la sección ConfigurarPostgres del caso de uso “Configurar gestores de base de datos”.

Sección ConfigurarOracle:

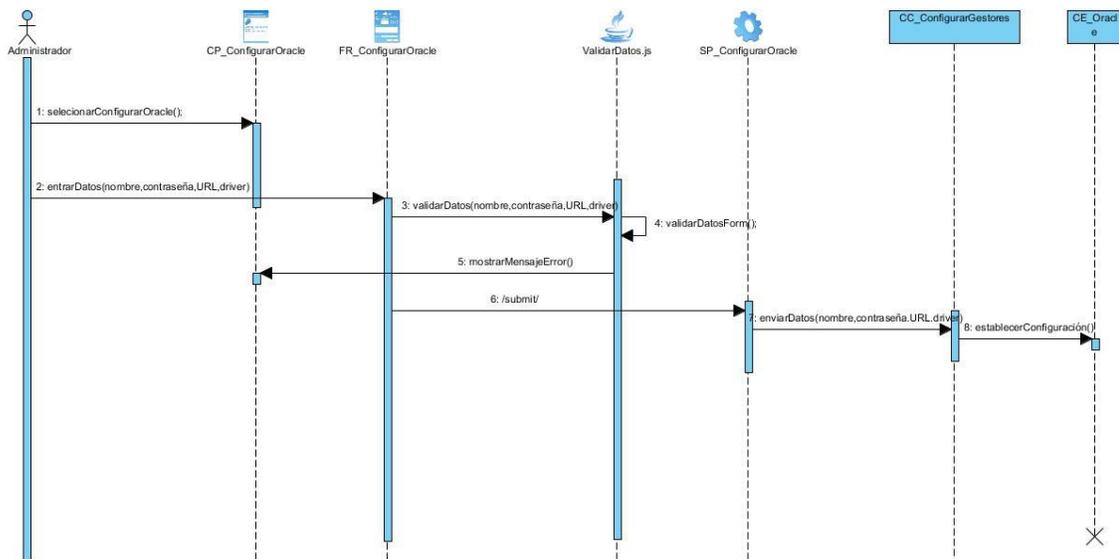


Figura 10: Diagrama de secuencia de la sección ConfigurarPostgres del caso de uso “Configurar gestores de base de datos”.

Sección ConfigurarMySQL:

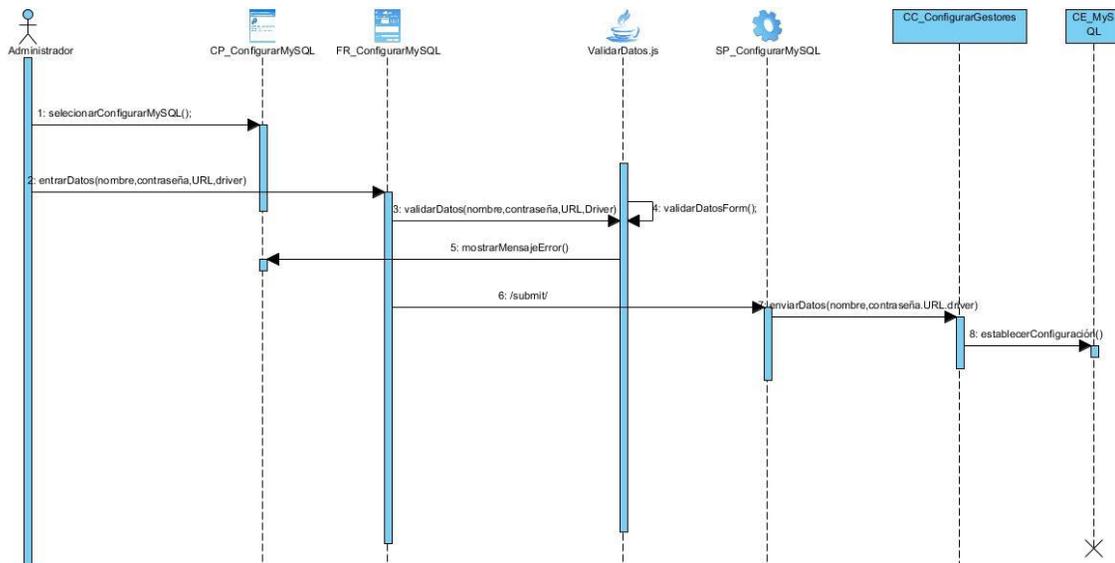


Figura 11: Diagrama de secuencia de la sección ConfigurarMySQL del caso de uso “Configurar gestores de base de datos”.

2.7 Diagrama de despliegue

El diagrama de despliegue describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos del cómputo. Detalla las capacidades de red, las especificaciones del servidor, los requisitos de hardware y otra información relacionada al despliegue del sistema propuesto (4). Además, tiene en cuenta los siguientes requerimientos: disponibilidad del sistema, rendimiento y escalabilidad.

Un nodo es un objeto físico que representa un recurso informático, este recurso generalmente dispone de datos persistentes y capacidad de proceso. Las conexiones entre nodos muestran las líneas de comunicación con las que el sistema tendrá que interactuar (4).

A continuación se muestra el diagrama de despliegue de la herramienta. El cual está compuesto por una PC_cliente para acceder a la aplicación, que se comunica con el servidor de aplicación (Pentaho BI Server) por el protocolo HTTPS, este a la vez se comunica con los tres servidores de base de datos (Postgres, Oracle y MySQL) por el protocolo JDBC .

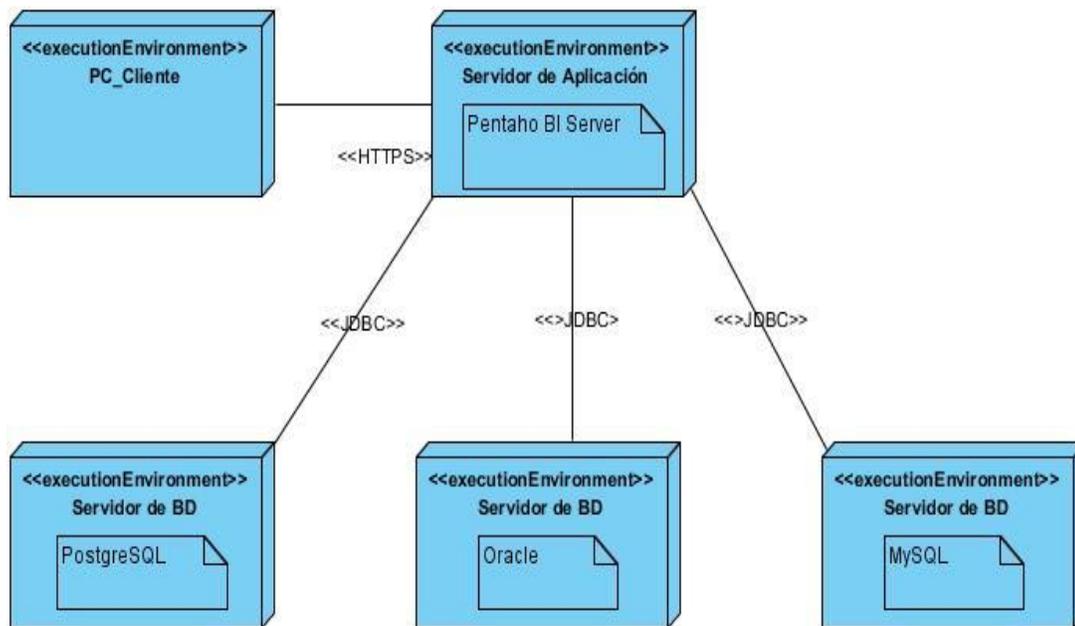


Figura 12: Diagrama de despliegue de la herramienta.

Conclusiones del capítulo

En el presente capítulo se realizó el levantamiento de los requisitos funcionales y no funcionales, los cuales se identificaron a partir de las necesidades reales de los usuarios, de las demandas de los clientes y experiencias adquiridas en el proceso de desarrollo de software. Posibilitando de esta forma lograr un entendimiento entre el cliente y el equipo de desarrollo. A partir de los requisitos se conformó el diagrama de casos de uso del sistema y posteriormente el modelo de diseño. Como parte de la realización de los casos de uso se modelaron los diagramas de secuencia, los cuales muestran una sucesión lógica de acciones para dar respuesta a las peticiones del cliente. La forma en que los componentes se desplegarán a lo largo de la infraestructura del sistema quedó descrita mediante el diagrama de despliegue, dando paso a las actividades de implementación del sistema.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

Introducción

En el presente capítulo se presentará el modelo de implementación correspondiente a la herramienta, desglosándolo en los diagramas de componentes de los casos de uso arquitectónicamente significativos. De la misma forma, se definen las pruebas que se van a incluir en el proceso de prueba para validar la herramienta de configuración de la seguridad del Pentaho BI Server, compuesto por los niveles, los métodos, las técnicas y los casos de pruebas que se aplicarán.

3.1 Modelo de implementación

El modelo de implementación describe cómo los elementos del modelo de diseño se implementan en términos de componentes, ficheros de código fuente, ejecutables, entre otros. Describe también la organización de los componentes de acuerdo a los mecanismos de estructuración y modularización disponibles en el entorno de implementación y lenguajes de implementación empleados, también cómo dependen los componentes unos de otros (19).

3.2 Diagrama de componentes

Un componente es una parte física de un sistema, también se puede decir que es la materialización de una o más clases, porque una abstracción con atributos y métodos pueden ser implementados en los componentes. Un diagrama de componentes permite visualizar con más facilidad la estructura general del sistema y el comportamiento del servicio que estos componentes proporcionan y utilizan a través de las interfaces (19). A continuación se muestra el diagrama de componentes del caso de uso significativo “Configurar gestores de base datos”.

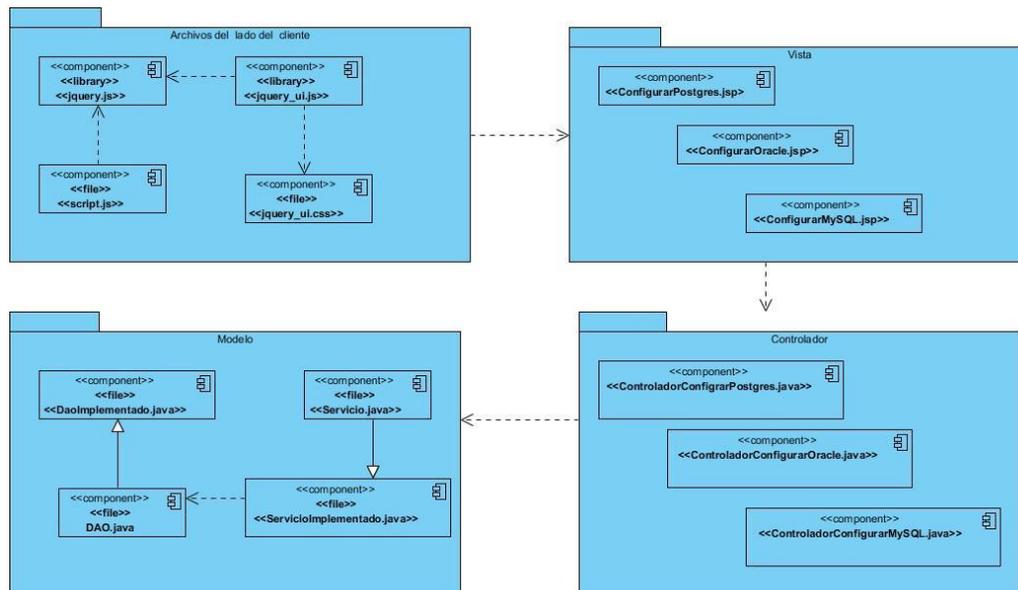


Figura 13: Diagrama de componentes para el caso de uso “Configurar gestores de base de datos”.

3.3 Estándares de codificación

Los estándares de codificación, también llamados estilos de programación o convenciones de código, no son más que convenios para escribir código fuente en ciertos lenguajes de programación. Estos estándares elevan el mantenimiento del código, sirven como punto de referencia para los programadores, mantienen un estilo de programación y ayudan a mejorar el proceso de codificación, haciéndolo, entre otras cosas, mucho más eficiente (20).

Las convenciones de código son importantes para los programadores por un gran número de razones (23):

- El 80% del coste del código de un programa va a su mantenimiento.
- Casi ningún software lo mantiene toda su vida el autor original.
- Las convenciones de código mejoran la lectura del software, permitiendo entender código nuevo mucho más rápidamente y más a fondo.

A continuación se presentarán algunas de las convenciones que se utilizarán en la implementación (23):

Indentación

Se emplearán cuatro espacios como unidad de indentación.

Ejemplo:

```
private String setAtributo(String line, String atr,String value) {
****int pos=line.indexOf(atr);
    if(pos!=-1){
****String aux1=line.substring(0,pos);
        String aux2=line.substring(pos+atr.length(),line.length());
        return aux1+value+aux2;
    }
return line;
}
```

Comentarios

Se utilizará dos tipos de comentarios, los de implementación y los de documentación, estos últimos existen solo en Java y se encuentran limitados por `/** ...*/`.

Ejemplo:

```
/**
    Clase para configurar Postgres
*/
private class Postgres() {
```

Declaraciones

Se realizará una declaración por línea, debido a que facilita los comentarios.

Ejemplo:

```
public class XML {
    private File xml;
    private String user;
    private String password;
```

Inicialización

Se inicializará las variables locales donde se declaran. La única razón para no inicializar una variable donde se declara es si el valor inicial depende de algunos cálculos que deben ocurrir.

Colocación

Las declaraciones solo se pondrán al principio de los bloques (un bloque es cualquier código encerrado por llaves "{" y "}"). Con la excepción de los ciclos.

Métodos

Se separarán los métodos con una línea en blanco.

Ejemplo:

```
private String setAtributo(String line, String atr,String value) {
    int pos=line.indexOf(atr);
    if(pos!=-1){
        String aux1=line.substring(0,pos);
        String aux2=line.substring(pos+atr.length(),line.length());
        return aux1+value+aux2;
    }
    return line;
}

public File getXml() {
    return xml;
}
```

Convenciones de nombres para las variables

Los nombres de las variables empezarán con minúscula en caso de que sean palabras compuestas deben ser sin espacios y con la primera letra de la segunda palabra en mayúscula. Los nombres de variables de un solo carácter solo se utilizarán para variables índices temporales.

Ejemplo:

```
String nombre="";  
int valor=0;
```

Convenciones de nombres para los métodos

Los métodos cuando son compuestos tendrán la primera letra en minúscula, y la primera letra de las siguientes palabras que lo forma en mayúscula.

Ejemplo:

```
private String setAtributo(String line, String atr,String value) {  
    int pos=line.indexOf(atr);  
    if(pos!=-1){  
        String aux1=line.substring(0,pos);  
        String aux2=line.substring(pos+atr.length(),line.length());  
        return aux1+value+aux2;  
    }  
    return line;  
}  
  
public File getXml() {  
    return xml;  
}
```

Convenciones de nombres para las clases

Se emplearán sustantivos para los nombres de las clases, en caso de tener un nombre compuesto, estos tendrán la primera letra de cada palabra que lo forma en mayúscula. No se utilizará acrónimos.

Ejemplo:

```
public class EstablecerConfiguracion()  
public class Postgres()
```

3.4 Mapa de navegación

El siguiente mapa de navegación se elaboró con el objetivo de brindar una mejor orientación a los usuarios que necesiten utilizar la herramienta de configuración de la seguridad del Pentaho BI.

En la figura 14 se muestra la página principal con todos los vínculos que existen a las diferentes páginas de la herramienta. Al usuario dar clic en un determinado vínculo, directamente este mostrará la página correspondiente en el área de trabajo que se encuentra en la página principal.

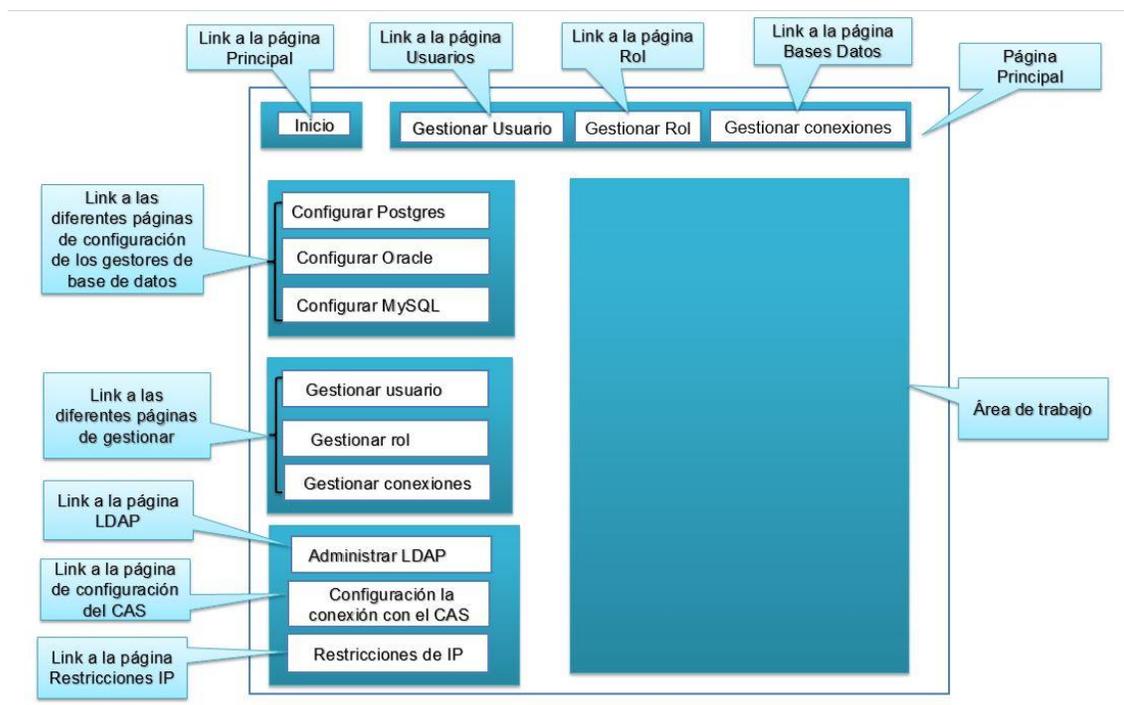


Figura 14: Mapa de navegación de la herramienta.

3.5 Pruebas de software

Un software debe ser probado para descubrir y corregir el máximo de errores posibles antes de su entrega al cliente, por lo que se hace necesario aplicarle una serie de pruebas.

La prueba es un proceso de ejecución de un programa con la intención de descubrir errores. Las pruebas definen el grado de aceptación del sistema que pueden tener los clientes del mismo, e incluso determinar si lo aceptan o no. Para ser más eficaces, las pruebas deben ser realizadas por extraños, que puedan

probarlo de forma despiadada, esto garantiza pruebas con alta probabilidad de encontrar errores. Las pruebas no pueden asegurar la ausencia de errores; sólo puede demostrar que existen defectos en el software (4).

3.5.1 Niveles de prueba

La Prueba es aplicada para diferentes tipos de objetivos, en diferentes escenarios o niveles de trabajo.

Se distinguen los siguientes niveles de prueba (4):

- *Prueba unitaria:* son diseñadas y ejecutadas por el desarrollador una vez concluido el desarrollo de cada componente. Es la prueba que valida si los componentes individuales de un programa están funcionando adecuadamente.
- *Prueba de sistema:* son ejecutadas por el desarrollador o equipo de pruebas en un ambiente controlado. Por consiguiente, este tipo de prueba determina el correcto funcionamiento de un sistema y su cumplimiento contra las especificaciones del producto. Permite detectar discrepancias entre el comportamiento del sistema construido y su especificación.
- *Pruebas de aceptación:* son pruebas funcionales que se realizan directamente con el cliente para validar los requerimientos y su conformidad con el producto. Son aquellas que demuestran al cliente que la aplicación está terminada y funciona correctamente.
- *Prueba de integración:* son diseñadas y ejecutadas por el desarrollador cuando la aplicación está completa. Es la prueba que determina la integración de los componentes dentro de un sistema.

3.5.2 Método de pruebas de caja blanca

La prueba de la caja blanca es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivar los casos de prueba (4).

Las pruebas de caja blanca intentan garantizar que: se ejercite por lo menos una vez todos los caminos independientes de cada módulo; ejerciten todas las decisiones lógicas en sus vertientes verdaderas y falsas; ejecuten todos los bucles en sus límites; y ejerciten las estructuras internas de datos para asegurar su validez (4).

3.5.2.1 Técnica camino básico

Una de las técnicas de prueba de caja blanca más usada es el camino básico, la cual determina la complejidad ciclomática de una porción de código. La complejidad ciclomática es una métrica del software que proporciona una medición cuantitativa de la complejidad lógica de un programa (4).

Cuando se usa el camino básico, el valor calculado como complejidad ciclomática define el número de caminos independientes del conjunto básico de un programa y nos da un límite superior para el número de pruebas que se deben realizar. Esta complejidad se puede calcular de tres formas que se pueden encontrar en (4).

3.5.3 Método de pruebas de caja negra

La prueba de la caja negra es un método de diseño de casos de prueba que se centra en los requisitos funcionales del software, por lo que se llevan a cabo sobre la interfaz del software. El objetivo es demostrar que las funciones del software son operativas, que las entradas se acepten de forma adecuada y que la integridad de la información externa se mantenga (4).

La prueba de caja negra intenta identificar errores de las siguientes categorías: funciones incorrectas o ausentes, errores de interfaz, errores en estructuras de datos o en accesos a bases de datos externas, errores de rendimiento y errores de inicialización y de terminación (4).

3.5.3.1 Técnica particiones equivalentes

Una de las técnicas de prueba de caja negra más usada es la partición equivalente, la cual divide el campo de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. La partición equivalente se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar (4).

3.5.4 Diseño de casos de prueba

Se trata de diseñar pruebas que tengan la mayor probabilidad de encontrar el mayor número de errores con la mínima cantidad de esfuerzo y de tiempo (4).

En la prueba de caja blanca se realiza un examen minucioso de los detalles procedimentales, comprobando los caminos lógicos del programa, comprobando los bucles y condiciones, y examinado el estado del programa en varios puntos (4).

En la prueba de la caja negra, los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta (4).

3.5.5 Proceso de prueba de la herramienta

Con la intención de poder descubrir y corregir el máximo de errores posibles en la herramienta desarrollada, se decidieron realizar las pruebas que se muestran en la figura 15; incluidas en el proceso de prueba, para de esta manera validar la herramienta de configuración de la seguridad del Pentaho BI:

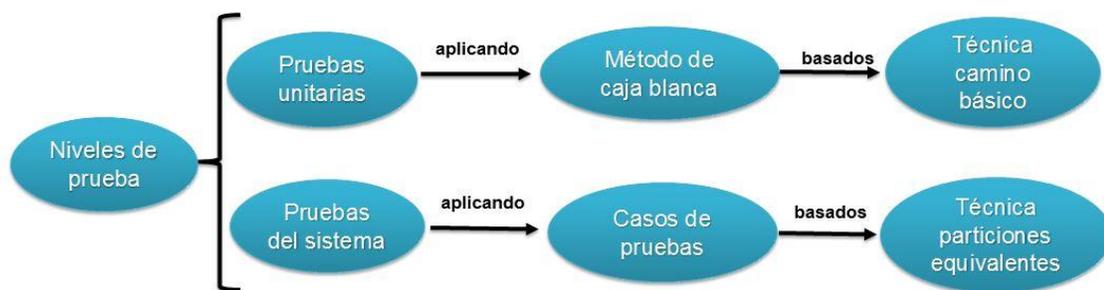


Figura 15: Proceso de prueba para la herramienta.

3.5.6 Aplicación de pruebas de caja blanca

Como se mencionó en acápites anteriores, una de las técnicas de prueba de caja blanca más usada es la de camino básico, que se aplica a fragmentos de código. En el proceso de prueba para validar la herramienta desarrollada se decidió aplicar esta técnica a unos de los métodos del caso de uso significativo “Configurar gestores de bases de datos”.

Al aplicar la técnica de camino básico al código del método *setAtributo* se identificaron 5 bloques de ejecución, los cuales se enumeraron (Véase, figura 16) para que puedan ser reconocidos. Además se determinó el camino básico que se muestra en la figura 17, donde existe un nodo predicado del cual se

derivan más de un camino a seguir, tal es el caso del nodo 2 y las aristas indican los posibles caminos a seguir a partir del nodo correspondiente.

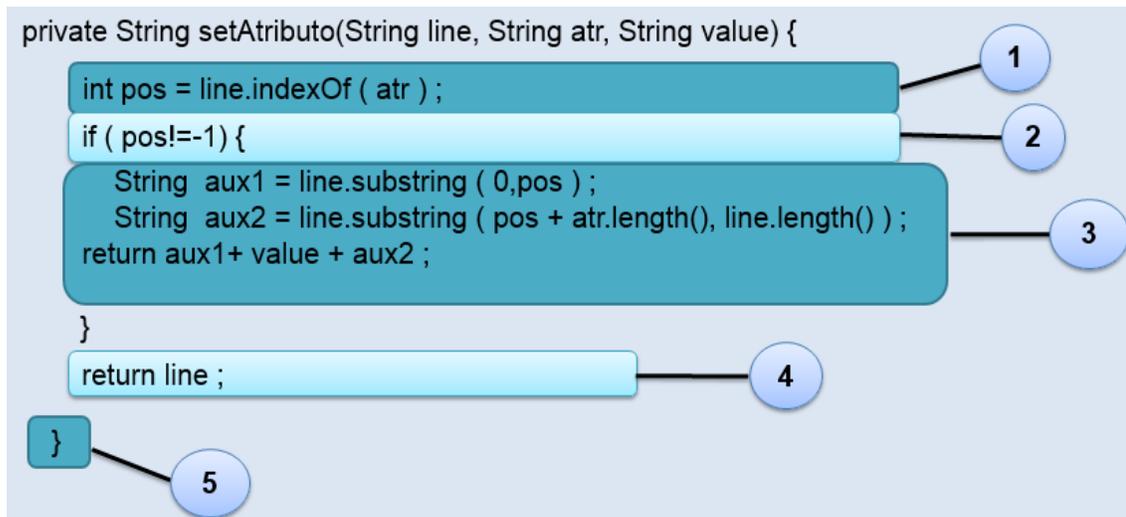


Figura 16: Enumeración de código para el método setAtributo.

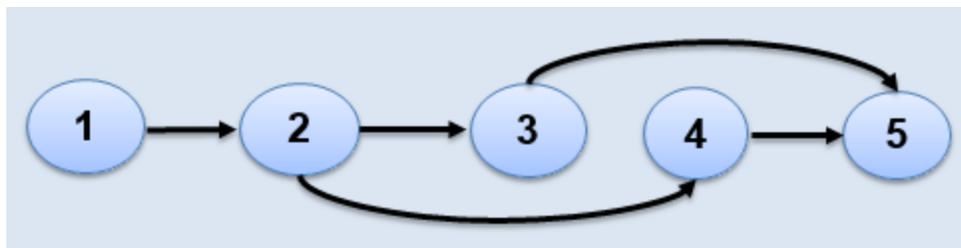


Figura 17: Grafo de flujo para el método setAtributo.

Después de representado el grafo de flujo se aplicaron las tres formas para calcular la complejidad ciclomática. Se utilizó la fórmula $V(G) = A - N + 2$, para la cual se identificaron cinco artistas y cinco nodos, por lo tanto: $V(G) = 5 - 5 + 2$, quedando $V(G) = 2$. De la misma forma se pueden comprobar que las otras variantes explicadas de calcular la complejidad ciclomática arriban al mismo resultado.

3.5.7 Casos de pruebas de caja negra

Para aplicar las pruebas al sistema se determinaron los casos de pruebas, para verificar el cumplimiento de un objetivo en particular.

Un caso de prueba se diseña según las funcionalidades descritas en los casos de uso. El propósito que se persigue con este artefacto es lograr una comprensión común de las condiciones específicas que la solución debe cumplir. Se parte de la descripción de los casos de uso del sistema, como apoyo para las revisiones. Cada planilla de caso de prueba recoge la especificación de un caso de uso, dividido en secciones y escenarios, detallando las funcionalidades descritas en él y describiendo cada variable que recoge el caso de uso en cuestión. Se efectuaron los casos de pruebas a los 10 casos de uso del sistema, plasmándose en la documentación del proyecto. (Ver planilla Diseño de Casos de Prueba)

Para el caso de uso: “Configurar gestores de base de datos” se definieron cuatro variables que se describen en la tabla que se muestra a continuación.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	nombre	String	No	Muestra el nombre del usuario que se va a conectar a la base de datos.
2	contraseña	String	No	Muestra la contraseña del usuario que se va a conectar a la base de datos.
3	URL	String	No	Muestra la dirección de la URL de la base de datos.
5	driver	String	No	Muestra el driver de la conexión a la base de datos

Tabla 6: Variables del caso de uso “Configurar gestores de base de datos”.

Escenario	Descripción	Variables de Entrada	Variables de Salida	Respuesta del sistema	Flujo central				
EC 1 Configuración existosa a PostgreSQL.	El administrador introduce los datos de la configuración del biserver correctamente y se realiza la configuración,terminándose el CU.	<table border="1"> <tr><td>nombre</td></tr> <tr><td>URL</td></tr> <tr><td>driver</td></tr> <tr><td>contraseña</td></tr> </table>	nombre	URL	driver	contraseña	Se configuró el biserver satisfactoriamente.	El sistema configura el biserver para que se conecte al gestor de base de datos.	Se selecciona la opción Migrar a Postgres situada en la parte izquierda del navegador y se muestra el formulario para que se llenen los datos de la configuración. Posteriormente se introducen los datos y se selecciona la opción aceptar.
nombre									
URL									
driver									
contraseña									

Tabla 7: Caso de prueba para el escenario 1 “Configuración exitosa a Postgres”.

Escenario	Descripción	Variables de Entrada	Variables de Salida	Respuesta del sistema	Flujo central
EC 2 Configuración con presencia de errores a PostgreSQL	El administrador introduce los datos de la configuración del biserver con errores y se muestra un mensaje de error,terminándose el CU.	nombre	Se muestran los errores que tienen los datos.	El sistema muestra un mensaje de error.	Se selecciona la opción Migrar a Postgres situada en la parte izquierda del navegador y se muestra el formulario para que se llenen los datos de la configuración. Posteriormente se introducen los datos y se selecciona la opción aceptar.
		URL			
		driver			
		contraseña			

Tabla 8: Caso de prueba para el escenario 2 “Configuración con presencia de errores a Postgres”.

La aplicación de los casos de prueba permitió verificar el cumplimiento de los requisitos funcionales del sistema.

3.5.8 Resultado de las pruebas

El sistema desarrollado después de aplicarle las pruebas arrojó en la primera iteración 29 no conformidades (ver figura 17), clasificándolas en 13 de nivel bajo, 6 de nivel medio y 10 de nivel alto (ver figura 18), las cuales fueron resultas, culminando el presente trabajo con resultados satisfactorios.



Figura 18: Cantidad de no conformidades.

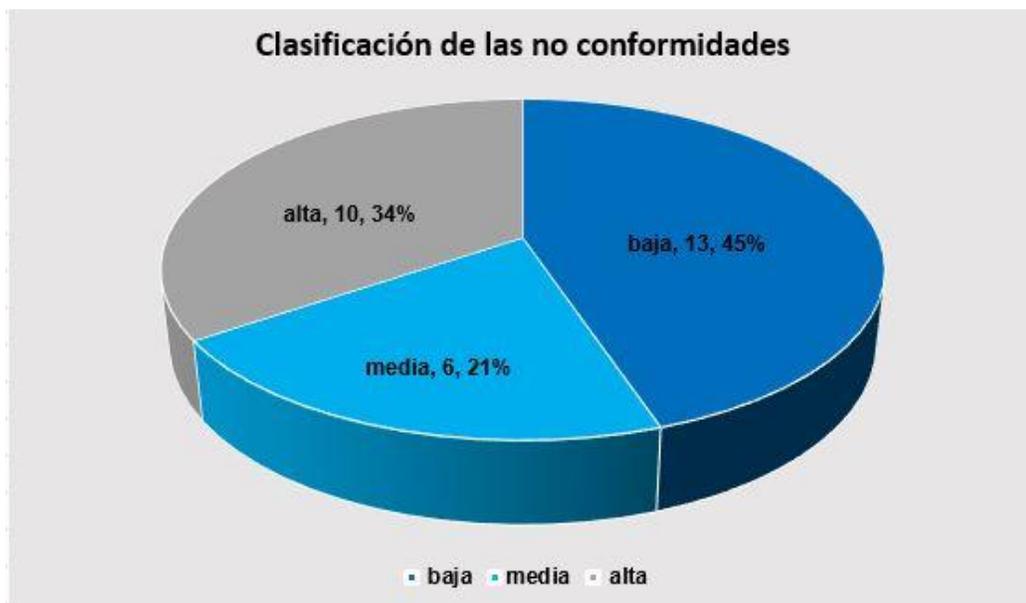


Figura 19: Clasificación de las no conformidades.

Conclusiones del capítulo

Como resultado de este capítulo se obtuvo la implementación del sistema en términos de componentes, proporcionando la solución necesaria a los requisitos especificados en el capítulo anterior. Una vez concebida la estructura y la implementación del sistema, se realizaron ocho casos de pruebas para validar la completitud de los requerimientos, obteniéndose 32 no conformidades, clasificándolas en 13 de nivel bajo, 6 de nivel medio y 10 de nivel alto, las cuales fueron resultas, culminando el presente trabajo con resultados satisfactorios.

CONCLUSIONES GENERALES

Para el desarrollo de la herramienta de configuración de la seguridad del Pentaho BI Server:

- La elaboración del marco teórico permitió la selección de la metodología, las herramientas y tecnologías que se utilizaron durante la implementación de la herramienta.
- A partir de las funcionalidades identificadas se realizó el modelo de diseño basado en la arquitectura Modelo Vista Controlador, proporcionando la entrada apropiada y el punto de partida para las actividades de implementación.
- Se implementó la herramienta que posibilita automatizar la configuración de los diferentes mecanismos de autenticación del Pentaho BI Server en su versión comunitaria
- Las pruebas realizadas para validar las funcionalidades implementadas, demostraron que se desarrolló una herramienta con la calidad requerida.

RECOMENDACIONES

Luego de haber analizado los resultados del presente trabajo de diploma, surgen algunas ideas que podrían ser incorporadas en un futuro con el objetivo de fortalecer el sistema desarrollado, por lo que se recomienda:

- Garantizar la configuración del Pentaho BI Server solo con el LDAP.
- Hacer salvas de las configuraciones de las conexiones que se han establecido en el Pentaho BI Server.

REFERENCIAS BIBLIOGRÁFICAS

1. **Pentaho Corporation.** Pentaho. [Online] 2005. [Cited: 10 15, 2012.] <http://www.pentaho.com>.
2. **Jorge Alfredo Medina Soto .** GestioPolis. GestioPolis. [En línea] [Citado el: 13 de 02 de 2013.] <http://www.gestiopolis.com/recursos5/docs/ger/buconce.htm>.
3. **Pentaho Corporation.** Pentaho. [En línea] 1995-2012. [Citado el: 07 de 11 de 2012.] <http://community.pentaho.com>.
4. **Pressman, Roger S.** Ingeniería de Software.Un enfoque práctico. 6ta Edición. s.l. : McGraw-Hill Companies, 2007. ISBN: 8448132149.
5. **Pentaho Corporation.** The Pentaho Security Guide. 2011.
6. **Web Chuck.** CAS. [Online] 2009. [Cited: 11 02, 2012.] <http://www.jasig.org/cas>.
7. **EcuRed.** EcuRed. [En línea] 14 de 12 de 2010. [Citado el: 04 de 11 de 2012.] <http://www.ecured.cu/index.php/LDAP>.
8. EcuRed. [En línea] 14 de 12 de 2010. [Citado el: 05 de 11 de 2012.] <http://www.ecured.cu/index.php/OpenUp>.
9. EcuRed. [En línea] 14 de 12 de 2010. [Citado el: 05 de 11 de 2012.] http://www.ecured.cu/index.php/Sistema_Gestor_de_Base_de_Datos.
10. EcuRed. [En línea] 14 de 12 de 2010. [Citado el: 04 de 10 de 2012.] http://www.ecured.cu/index.php/Herramienta_CASE.
11. **UCI.** PosrgreSQL Cuba. [En línea] [Citado el: 14 de 11 de 2012.] <http://postgresql.uci.cu>.
12. **Dataprix.** Dataprix. Dataprix. [En línea] Creative Commons. [Citado el: 5 de 10 de 2012.] <http://www.dataprix.com/723-caracter-sticas-pentaho>.
13. **Visual Paradigm.** Visual Paradigm. [Online] [Cited: 12 06, 2012.] <http://www.visual-paradigm.com/>.
14. **Microsoft.** MSDN. [En línea] 2013. [Citado el: 05 de 02 de 2013.] <http://msdn.microsoft.com/es-es/library/syf5yeat.aspx>.
15. **Oracle.** Oracle. [En línea] 2012. [Citado el: 07 de 11 de 2012.] <http://www.oracle.com>.

16. **EcuRed.** EcuRed. [En línea] [Citado el: 05 de 02 de 2013.]
http://www.ecured.cu/index.php/Lenguajes_de_Programaci%C3%B3n_Web.
17. Minter, D Beginning Spring 2. From Novice to Professional. New York, E.E. U.U. Apress.2008. 298.
18. Eclipse (software). [En línea] [Consultado el: 19 de febrero de 2011] Disponible en:
<http://es.scribd.com/doc/55151939/Taller-de-Rails>.
19. JACOBSON, I.; G. BOOCH, et al. El proceso Unificado de Desarrollo de Software. p.xviii Addison Wesley Object Technology.
20. usuarios.multimania. usuarios.multimania. [En línea] [Citado el: 5 de Abril de 2013.]<http://usuarios.multimania.es/manualesjava/manuales/convencionescodigojava/convencionescodigojava.pdf>.

BIBLIOGRAFÍA

Jorge Alfredo Medina Soto . GestioPolis. *GestioPolis*. [En línea] [Citado el: 13 de 02 de 2013.] <http://www.gestipolis.com/recursos5/docs/ger/buconce.htm>.

Pentaho Corporation. Pentaho. [Online] 2005. [Cited: 10 15, 2012.] <http://www.pentaho.com>.

Pentaho Corporation. Pentaho. [En línea] 1995-2012. [Citado el: 07 de 11 de 2012.] <http://community.pentaho.com>.

Microsoft. MSDN. [En línea] 2013. [Citado el: 05 de 02 de 2013.] <http://msdn.microsoft.com/es-es/library/syf5yeat.aspx>.

Web Chuck. CAS. [Online] 2009. [Cited: 11 02, 2012.] <http://www.jasig.org/cas>.

EcuRed. EcuRed. [En línea] 14 de 12 de 2010. [Citado el: 04 de 11 de 2012.] <http://www.ecured.cu/index.php/LDAP>.

Pressman, Roger S. *Ingeniería de Software.Un enfoque práctico*. 6ta Edición. s.l. : McGraw-Hill Companies, 2007. ISBN: 8448132149.

Ecured. EcuRed. [En línea] 14 de 12 de 2010. [Citado el: 05 de 11 de 2012.] <http://www.ecured.cu/index.php/OpenUp>.

IbéricaS.A.AI. MicroStrategy. [Online] 2012. [Cited: 11 05, 2012.] <http://www.microstrategy.es>.

Oracle. Oracle. [En línea] 2012. [Citado el: 07 de 11 de 2012.] <http://www.oracle.com>.

Trademarks. Qlikview. [Online] 1993-2012. [Cited: 11 07, 2012.] <http://www.qlikview.com>.