# Universidad de las Ciencias Informáticas

**“Facultad # 6”**


# Plugin para la gestión de la seguridad de los esquemas de Procesamiento Analítico en Línea.

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

**Autores:** Isenith Valdés Correoso
Antonio Miguel Pérez Aramillo

**Tutores:** Ing. Roberto Tellez Ibarra
Ing. Leonel Pérez Nieblas

La Habana, Junio 2013
“Año 55 de la Revolución”

***El futuro de nuestra patria tiene que ser necesariamente un futuro de hombres de ciencia, tiene que ser un futuro de hombres de pensamiento, porque precisamente es lo que más estamos sembrando; lo que más estamos sembrando son oportunidades a la inteligencia, ya que una parte considerable de nuestro pueblo no tenía acceso a la cultura, ni a la ciencia. Era una riqueza de la cual no podía nada esperarse porque no tenían la oportunidad.***

**La Habana, 15 de enero de 1960**

**DECLARACIÓN DE AUTORÍA**

Declaramos ser autores de la presente tesis que tiene por título: Plugin para la gestión de la seguridad de los esquemas de Procesamiento Analítico en Línea y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ____ días del mes de junio del año 2013.

**Isenith Valdés Correoso**

_____

Firma del Autor

**Antonio Miguel Pérez Aramillo**

_____

Firma del Autor

**Ing. Roberto Tellez Ibarra**

_____

Firma del Tutor

**Ing. Leonel Pérez Nieblas**

_____

Firma del Tutor

**DATOS DE CONTACTO**

Tutor: Ing. Roberto Tellez Ibarra

Email: rtibarra@uci.cu

Profesor Instructor

Graduado en la Universidad de las Ciencias Informáticas en 2009.

Tutor: Ing. Leonel Pérez Nieblas

Email: lnieblas@uci.cu

Graduado en la Universidad de las Ciencias Informáticas en 2011.

**AGRADECIMIENTOS**

***Isenith***

A mi mamita y mi papito porque gracias a su cariño, apoyo y sacrificio he llegado hasta aquí

A mi abuela Tela que ha sido como mí otra madre dándome siempre todo su amor incondicional.

A mi abuelo que aunque siempre ha sido una persona reservada ha sabido educarme para ser lo que he logrado ser hoy

A mis tíos Yovany y Lile, mejores que ellos ninguno, mis segundos padres, el segundo un poco resabioso, pero siempre han estado ahí para ayudarme y aconsejarme siempre que me ha hecho falta.

A mis tías Maydielis y Yurema, que de una forma u otra siempre me han apoyado y ayudado.

A mi tío Ernesto, que hace pocos años lo conozco pero me ha bridado su ayuda y su cariño como si nos conociéramos de toda la vida.

A mi abuela Nereida y mi abuelo Ernesto que aunque viven un poco lejos siempre se han preocupado por mi y me han brindado todo su cariño.

A mi familia en general, que han estado siempre al pendiente de mí.

A mi novio y compañero de tesis Tony, al que no se como devolverle lo que ha hecho por mi durante estos 5 años y que juntos hemos logrado llegar hasta aquí. Gracias por soportarme y sobre todo por tener paciencia conmigo y entenderme.

A su familia que me acogieron en la suya sin problemas y me han dado su apoyo siempre durante todo este tiempo.

A mis compañeras de apartamento y a todas mis amistades que han compartido conmigo estos 5 años.

A los profesores que me han ayudado y contribuido en mi formación durante todo este periodo, y han estado ahí en cualquier cosa que me halla hecho falta.

A mis tutores que gracias a su ayuda han permitido que esté hoy aquí.

**AGRADECIMIENTOS**

***Tony***

A mis padres y a mi hermano por siempre haber estado a mi lado, apoyándome y guiándome por el buen camino y que gracias a ellos hoy he podido llegar hasta aquí.

A mi novia Isenith, a quien quiero mucho, por haberme brindado su apoyo durante los 5 años, por los momentos que juntos compartimos, buenos o malos, pero siempre juntos.

A mi familia en general que siempre estuvo presente y que depositaron toda su confianza en mí.

A mi otra familia, que me ha apoyado mucho durante estos 5 años.

A mis tutores por ser mi guía durante el desarrollo de la tesis y además al profesor Yosbel, que de verdad siempre estuvo presente cuando lo necesitamos.

A mis compañeros de clase que me han brindado su apoyo incondicional siempre que los he necesitado.

A los profesores del departamento, que de una manera u otra contribuyeron con nuestra formación.

**DEDICATORIA**

A nuestras familia por el amor preocupación, paciencia, confianza y apoyo que nos han brindado y sobretodo por creer en nosotros.

Un beso grande.

## RESUMEN

La Universidad de la Ciencias Informáticas, está compuesta por varios centros de producción de software dentro de los cuales existe un departamento encargado de la elaboración de almacenes de datos. Este departamento posee un equipo de Inteligencia de Negocio el cual se centra en la presentación de los datos. Para este fin se utiliza la plataforma Pentaho BI Server que funciona como un sistema de gestión basado en la web.

El presente trabajo de diploma surge debido a las deficiencias de usabilidad que presenta el Mondrian a la hora de gestionar permisos a los roles para el Pentaho. Además es necesario garantizar la seguridad a nivel de registros en el sistema, así como buscar una solución para actualizar los permisos existentes de forma que el usuario pueda ver los permisos que se asignaron con anterioridad y actualizar el sistema con nuevos datos que se hayan introducido.

El trabajo tiene como objetivo el desarrollo de un plugin que al estar integrado al Pentaho BI Server, permita la gestión de la seguridad de los esquemas de Procesamiento Analítico en Línea (OLAP) publicados en el sistema.

Para ello se llevó a cabo el análisis, diseño, implementación y prueba del plugin propuesto. Conjuntamente se investigaron y emplearon varias herramientas y una metodología para guiar el desarrollo, obteniéndose como resultado, un plugin que cumple con las exigencias del cliente.



## PALABRAS CLAVE

esquemas, OLAP, pentaho, plugin, seguridad

**TABLA DE CONTENIDOS**

## ÍNDICE DE FIGURAS

## ÍNDICE DE TABLAS

## Introducción

En la actualidad la sociedad ha venido enfrentándose a varios cambios y a nuevos retos que han ido apareciendo con el transcurso del tiempo. Uno de estos nuevos cambios han sido las Tecnologías de la Información y Comunicación (TIC) las cuales se han ido insertando y provocando un gran impacto no solo en el ámbito social sino también en el económico, político y cultural. Las TICs se han convertido en un medio indispensable para la sociedad. El rápido avance de estas tecnologías brinda oportunidades sin precedentes para alcanzar niveles más elevados de desarrollo.

Cuba a pesar de las múltiples limitaciones que posee, tanto económicas como sociales se encuentra inmerso en un proceso de transformaciones con el objetivo de alcanzar mayor desarrollo en el campo de la informática y las comunicaciones haciendo uso de las TIC. Estas se han convertido en un elemento indispensable para establecer las líneas de desarrollo de la sociedad cubana, siendo un salto vertiginoso en el desarrollo científico técnico a escala mundial. Su auge alcanzado en el mundo, traza nuevas pautas en función de la informatización del país, produciendo importantes transformaciones a nivel nacional. Se ha ido enfocando su aplicación en la salud pública, en la ciencia, la cultura, la economía, los servicios públicos y la educación; siendo esta última la rama de mayor importancia. Para poner en vigencia estas transformaciones existen un conjunto de instituciones cubanas que potencian este desarrollo.

La Universidad de las Ciencias Informáticas (UCI), se encuentra involucrada en el desarrollo de la sociedad cubana en lo referente a este campo. Esta tiene como misión dirigir la formación del Ingeniero en Ciencias Informáticas con conocimientos, habilidades y valores sólidos, que estén comprometidos con su patria y que actúen como verdaderos profesionales.

La universidad combina el estudio con la producción y la investigación, por lo que en ella existen varios centros de desarrollo de software, siendo uno de ellos el Centro de Tecnologías de Gestión de Datos (DATEC), en el cual se desarrollan soluciones de almacenes de datos, para esto existen varias líneas de desarrollo como el equipo de Inteligencia de Negocio o Business Intelligence (BI), el cual se centra en las técnicas que permiten convertir los datos en información y la información en conocimiento para apoyar el proceso de toma de decisiones. Con este fin se utilizan varias plataformas, pero la empleada por el departamento es el Pentaho BI Server que funciona como un sistema de gestión basado en la web.

El BI Server es una solución de BI completa y fácil de usar, perteneciente a la Suite de Pentaho, mediante

la cual se pueden mostrar entre otras cosas, perspectivas de análisis de información para apoyar el proceso de toma de decisiones en la inteligencia de negocio. Sin embargo, el motor multidimensional que utiliza, presenta algunas deficiencias de usabilidad a la hora de gestionar permisos a los roles en los esquemas OLAP (Procesamiento Analítico en Línea), lo que trae consigo que aquellos usuarios que no tengan dominio de la sintaxis empleada en el esquema XML no pueden definir los permisos que se almacenan en estos, provocando que una vez que un usuario entra al sistema, sin importar el rol que posea tendrá acceso a toda la información modelada en los esquemas OLAP, siendo vulnerables al robo, modificación o pérdida de los datos.

Otra deficiencia, es que un usuario del Pentaho Schema Workbench no puede acceder a los datos a nivel de registros cuando se diseñan los esquemas, lo que imposibilita asegurar estos datos y garantizar que se asignen los permisos con integridad, evitando definir niveles de acceso a datos que no existan en las fuentes modeladas. Además, si se comete algún error al asignar un permiso puede quedar inútil el esquema en que se está trabajando.

Por otro lado, el sistema carece de la manera de actualizar los permisos existentes de forma que el usuario pueda ver los permisos que se asignaron con anterioridad, así como tampoco permite actualizar el sistema con nuevos datos que se hayan introducido.

De la situación antes planteada surge el siguiente problema:

¿Cómo contribuir a la seguridad de los esquemas OLAP del servidor de Inteligencia de Negocios de la Suite de Pentaho?

Por lo tanto el **objeto de estudio** de investigación se centra en la gestión de la seguridad de los esquemas OLAP, defendiéndose como **campo de acción** la gestión de la seguridad en los esquemas OLAP en el Pentaho BI Server.

Con el propósito de dar solución al problema anteriormente planteado, se ha trazado como **objetivo general** de esta investigación, desarrollar un plugin para gestionar la seguridad en los esquemas OLAP del servidor de Inteligencia de Negocios de la Suite de Pentaho.

Teniéndose como **objetivos específicos**:

1. Realizar un estudio sobre las herramientas existentes que aseguran la seguridad de los esquemas OLAP del servidor de Inteligencia de Negocios.
2. Realizar el análisis y diseño del plugin de seguridad de los esquemas OLAP del servidor de Inteligencia de Negocios.
3. Implementar el plugin de seguridad.
4. Realizar pruebas a los resultados obtenidos.

Para cumplir con los objetivos propuestos y dar solución a la problemática planteada en esta investigación, se proponen las siguientes **tareas de investigación**:

1. Estudio del estado del arte de las herramientas existentes para asegurar la seguridad en los cubos OLAP.
2. Estudio y selección de las principales metodologías de desarrollo de software, herramientas y tecnologías para la implementación del sistema.
3. Levantamiento de requisitos del subsistema de gestión de permisos a los roles.
4. Selección de los patrones de diseño y arquitectónico a utilizar en el sistema.
5. Realización del modelo de diseño.
6. Elaboración del diagrama de componentes.
7. Implementación de los componentes.
8. Realización de pruebas.

Con el desarrollo de este trabajo se espera como un posible **resultado:**

- Un plugin que contribuya a la seguridad de los esquemas OLAP del servidor de Inteligencia de Negocios de la Suite de Pentaho en el cual se puedan asignar permisos a los roles dentro de los esquemas ya publicados en el Pentaho BI Server.

El presente trabajo se ha estructurado por Resumen, Introducción, tres capítulos Conclusiones, Recomendaciones, Referencias Bibliográficas, Bibliografía y Anexos. Para una mejor comprensión del trabajo se realizará una breve descripción de los tres capítulos donde se abarca toda la investigación realizada:

**Capítulo 1: Fundamentos teóricos.** En este capítulo se hace énfasis en los conceptos y definiciones necesarios para desarrollar un plugin. Se realiza un análisis de algunas herramientas y metodologías para definir la más apropiada.

**Capítulo 2: Análisis y diseño del sistema.** En este capítulo se presenta la propuesta de la solución que se quiere implementar y se realiza todo lo vinculado con la fase de análisis y diseño del plugin según la metodología seleccionada. Además se desarrolló la captura de requisitos funcionales y no funcionales que el sistema debe cumplir para lograr una mejor comprensión de la funcionalidad e integridad del sistema.

**Capítulo 3: Implementación y pruebas.** En este capítulo se explica todo lo relacionado con la implementación, se elaboran los diagramas de despliegue y de componentes y se muestran distintos elementos del desarrollo del plugin. Además, se diseñarán y aplicarán las pruebas a la solución propuesta.

# Capítulo 1: Fundamentos Teóricos

## Introducción

En el presente capítulo se realizará un estudio del estado del arte acerca de las herramientas que aseguran la seguridad de los esquemas OLAP. Además se darán a conocer los principales conceptos que contribuyen a un mejor entendimiento del problema en cuestión, así como la caracterización y selección de las diferentes metodologías y tecnologías actuales a utilizar en el desarrollo de la solución.

## 1.1. Cubos OLAP

En el mundo de las soluciones para BI, una de las herramientas más utilizadas por las empresas son las aplicaciones OLAP, las cuales permiten procesar grandes volúmenes de información, en campos bien definidos, y con un acceso inmediato a los datos para su consulta y posterior análisis. Para asegurar su funcionamiento, estas aplicaciones utilizan un tipo de base de datos que posee la peculiaridad de ser multidimensional, denominada comúnmente Cubos OLAP, los cuales son una base de datos que posee diversas dimensiones, ampliando las posibilidades que hasta el momento ofrecían las conocidas hojas de cálculo. (1)

Hasta la llegada de este nuevo término solo se utilizaban bases de datos relacionales para el proceso de la información, con sistemas tales como el ROLAP (Relacional OLAP). Gracias a la incorporación de las bases de datos de tipo multidimensional y el nacimiento del nuevo concepto, las herramientas de soluciones para sistemas BI han avanzado notablemente en cuanto a las prestaciones que estas aplicaciones brindan a las empresas, donde la información confiable, precisa y en el momento oportuno, son uno de los bienes más preciados. Cabe destacar que los Cubos OLAP son vectores en los cuales se dispone la información y gracias a esta ordenada jerarquía es posible llevar a cabo un análisis rápido de los datos. (1)

Cada una de las dimensiones que posee la base de datos incorpora un campo determinado para un tipo de dato específico, que luego podrá ser comparado con la información contenida en el resto de dimensiones, para hacer posible la evaluación y posteriores informes de la información realmente relevante para una compañía. (1)

Una base de datos multidimensional puede contener varios cubos o vectores que extenderán las posibilidades del sistema OLAP con el cual se trabaja. Por ello, si bien en general los sistemas OLAP

suelen estar compuestos por tres dimensiones, lo cierto es que existe la posibilidad de que el sistema OLAP albergue más de tres dimensiones mediante la utilización de estos Cubos OLAP. (1)

Para tener una idea más simple de la función de los Cubos OLAP dentro de una base de datos multidimensional, cabe destacar que cada una de las dimensiones o escalas del cubo corresponde básicamente a una jerarquía de datos. Un ejemplo claro de ello podría ser el siguiente caso: dentro de una escala temporal para incluir datos determinados a un periodo de tiempo, que llevará el nombre de "Enero de 2012", seguramente incluirá una dimensión denominada "Primer Trimestre de 2012", la cual además incluirá otra dimensión llamada "Año 2012" y así sucesivamente, de acuerdo a las necesidades de cada empresa. (1)

A pesar de las grandes ventajas que presenta este tipo de base de datos multidimensional que incluye Cubos OLAP, la cual permite obtener mayor rapidez en las consultas y en el procesamiento de la información, lo cierto es que su gran falla reside en la imposibilidad de realizar cambios en su estructura. Debido a su forma de funcionamiento y almacenamiento de la información, cuando los usuarios requieren realizar modificaciones en la estructura de este tipo de base de datos, deben rediseñar el Cubo OLAP, sin posibilidades de poder utilizar la estructura en la que se trabajó hasta el momento. (1)

Para los cubos, se ofrece tres formas de almacenar su información: (2)

- MOLAP - Multidimensional OLAP.
- ROLAP - Relacional OLAP.
- HOLAP - OLAP híbrido.

*MOLAP*

Los datos fuente del cubo son almacenados junto con sus agregaciones en una estructura multidimensional de alto rendimiento. El almacenaje de MOLAP, provee excelente rendimiento y compresión de datos. Como se dice, todo va en el cubo. Tiene el mejor tiempo de respuesta, dependiendo solo del porcentaje y diseño de las agregaciones del cubo. En general este método, es muy apropiado para cubos con uso frecuente por su rápida respuesta (2).

*ROLAP*

Toda la información del cubo, sus datos, su agregación, sumas, etc., son almacenados en una base de datos relacional. ROLAP no almacena copia de la base de datos, sino que accede a las tablas originales

cuando necesita responder consultas, es generalmente, mucho más lenta que las otras dos estrategias de almacenaje. Típicamente ROLAP se usa para largos conjuntos de datos que no son frecuentemente buscados, tales como datos históricos de los años más recientes (2).

*HOLAP*

Combina atributos de MOLAP y ROLAP, la agregación de datos es almacenada en una estructura multidimensional usada por MOLAP, y la base de datos fuente, se almacena en una base de datos relacional. Para procedimientos de búsqueda que acceden a datos sumarizados, HOLAP es equivalente a MOLAP, por el contrario si estos procesos accedieran a datos fuentes como los *drill down*, estos deben de buscar los datos en la base de datos relacional y esto no es tan rápido comparado a si los datos estuvieran almacenados en una estructura MOLAP. Los cubos almacenados como HOLAP, son más pequeños que los MOLAP y responden más rápidos que los ROLAP.

HOLAP es generalmente usado para cubos que requieren rápida respuesta, para sumarizaciones basadas en una gran cantidad de datos (2).

### 1.1.1. Seguridad

A la hora de diseñar el modelo multidimensional, es fundamental definir la seguridad adecuada sobre los diferentes componentes y niveles de la solución, debido a lo sensible que puede ser para la organización la información que suelen manejar este tipo de aplicaciones. Al igual que ocurre con las bases de datos de los sistemas transaccionales, en OLAP pueden manejarse distintos niveles de seguridad. La seguridad en OLAP tiene una arquitectura jerárquica, partiendo del cubo y llegando al nivel de medida dentro del cubo. (3)

De este modo, se definen los permisos a nivel de:

- Cubo
- Dimensión
- Medida

**Cubo**: esta restricción de seguridad se realiza sobre todo el cubo, se puede permitir o denegar el acceso al cubo.

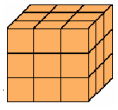

**Figura 1: Permitido**

**Figura 2: Denegado**

**Dimensión:** Se puede permitir que el usuario vea la dimensión, que acceda solo a una parte de ella, o que no tenga permiso de visualizarla.

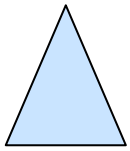

**Figura 3: Permitido**

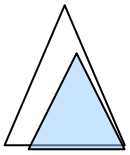

**Figura 4: Solo una parte**

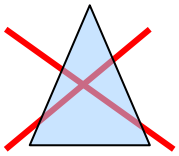

**Figura 5: Denegado**

**Medida:** En una celda o medida se puede permitir el acceso, o bien personalizarlo utilizando expresiones que verifiquen alguna condición para acceder a los datos.

Otra opción para limitar los accesos puede ser el uso de cubos virtuales. Podríamos crear un cubo virtual solo con las medidas que deseamos que tenga acceso el usuario y luego otorgar los permisos sobre el cubo virtual, y denegar o no los permisos sobre el cubo original.

Por ejemplo, si solo un grupo de usuarios puede visualizar el importe de los sueldos de los empleados, entonces podríamos definir una restricción de acceso a nivel celda, sobre la medida Sueldo o crear un cubo virtual que no muestre esta medida. (3)

En la actualidad existen algunas herramientas que permiten asegurar la seguridad de los cubos OLAP, a continuación se muestran algunas de ellas:

- **Microsoft SQL Server OLAP services**

Usando las facilidades de seguridad manejadas por Microsoft SQL Server OLAP Services, se puede controlar quien acceda a los datos y los tipos de operaciones que los usuarios pueden ejecutar con los datos. OLAP Services soporta el sistema de seguridad integrado que ofrece el sistema operativo Windows NT y permite que se asignen permisos de acceso, a la base de datos y al cubo incluyendo a los cubos virtuales(2).

La seguridad es manejada mediante los derechos de control de acceso manejados por los roles, estos determinan el tipo de acceso a los datos y definen categorías de usuarios con los mismos controles de acceso (2).

- **Suite de Pentaho**

Pentaho se define a sí mismo como una plataforma de BI “orientada a la solución” y “centrada en procesos” que incluye los principales componentes requeridos para implementar soluciones basadas en procesos para lo que ha sido concebido desde el principio (4).

Las soluciones de Pentaho están escritas en Java y tienen un ambiente de implementación también basado en el mismo lenguaje, permitiendo que sea una solución muy flexible para cubrir una gran cantidad de necesidades empresariales. El marco proporciona los servicios básicos, incluidos autenticación, registro, auditoría, servicios web y motor de reglas. La plataforma también incluye un motor de solución que integra reportes, análisis, tableros de comandos y componentes de minería de datos. Además posee una arquitectura basada en plugin lo que permite a toda o parte de la plataforma estar inmersa en aplicaciones de terceros por los usuarios finales, así como fabricantes de equipos originales.

**Pentaho BI Server**

Una de las aplicaciones más conocidas de la plataforma de Pentaho es BI Server, esta proporciona el servidor y la plataforma web del usuario final, el cual podrá interactuar con la solución de BI previamente creada. También funciona como un sistema basado en administración web de informes, el servidor de integración de aplicaciones y un motor de flujo de trabajo ligero. Está diseñado para integrarse fácilmente en cualquier proceso de negocios y cuenta con una interfaz de usuario donde se encuentran disponibles todos los informes, vistas OLAP, cuadros de mando y accesos. (5)

**Pentaho Analysis Services (Mondrian)**

Mondrian ahora rebautizado como Pentaho Analysis Services es el motor OLAP integrado en la suite de Pentaho. Es un servidor de código abierto que gestiona la comunicación entre una aplicación que maneja información multidimensional mediante el uso de OLAP y una base de datos relacional. Además permite a los usuarios analizar grandes cantidades de información en tiempo real y puede ayudar a definir la seguridad de los cubos OLAP por roles. Mondrian es un motor ROLAP con caché, lo cual lo sitúa cerca del concepto de Híbrido OLAP (6).

**Pentaho Schema Workbench**

El elemento principal de Mondrian son los ficheros XML donde se representan los esquemas dimensionales. Para construir estos ficheros XML, se puede utilizar cualquier editor de texto o XML, o bien la herramienta que nos ofrece Pentaho, denominada Schema Workbench.

Pentaho Schema Workbench es la herramienta gráfica que permite la construcción de los esquemas de Mondrian y además permite publicarlos en el servidor de BI para que puedan ser utilizados en los análisis por los usuarios de la plataforma. (7)

En los ficheros de esquema XML, se describen las relaciones entre las dimensiones y medidas del cubo OLAP con las tablas y campos de la base de datos, a nivel relacional. Este mapeo se utiliza para ayudar la traducción de las querys MDX (que es el lenguaje con el que trabaja Mondrian), y para transformar los resultados recibidos de las consultas SQL a un formato dimensional. (7)

Como se mencionó anteriormente el Schema Workbench permite crear esquemas y cubos visuales. Un **esquema** es un contenedor de cubos donde se podrá crear tantos cubos como se desee, el cual tendrá un único fichero XML. Las propiedades que se pueden indicar al crear un esquema son un nombre, la descripción, un nombre para la dimensión que agrupará a las medidas y un rol por defecto para utilizar en las conexiones de base de datos. Una vez creado el esquema se procede a la creación de los cubos. (7)

Al crear el **cubo**, se le indica un nombre y una descripción. Se seleccionan las dimensiones que se quieren incluir en el cubo. El cubo hereda todas las características que se hayan incluido en la dimensión, incluyendo todas las jerarquías con sus correspondientes niveles. (7)

Como último paso en la creación del cubo, se definen en las medidas, que van a ser los valores de análisis. Se tienen aquellas que se calculan directamente con campos de la base de datos, y los miembros calculados**,** que son fórmulas en las que utilizan otras medidas. (7)

### 1.2. Plugin

Debido a que Pentaho BI Server posee una arquitectura basada en plugin hace más fácil el trabajo en cuestión debido a que se puede insertar exitosamente el complemento a desarrollar. Un plugin se puede definir como:

Un módulo de hardware o software que puede anexarse a un sistema, generalmente más grande, para aumentar sus funcionalidades. Esta aplicación adicional es ejecutada por la aplicación principal e interactúan por medio de una Interfaz de Programación de Aplicaciones (API). También se lo conoce como complemento, *add-on* (agregado), conector o extensión. (8)

**Funcionamiento de los plugin**

La aplicación principal o *host* proporciona servicios que el complemento puede utilizar, incluyendo un

método para que los complementos se registren a sí mismos y un protocolo para el intercambio de datos. Los complementos dependen de los servicios prestados por la aplicación de acogida y no suelen funcionar por sí mismos. Por el contrario, la aplicación principal funciona independientemente de ellos, lo que permite a los usuarios finales añadir y actualizar los complementos de forma dinámica sin necesidad de hacer cambios a la aplicación principal (8).

Las APIs proporcionan una interfaz estándar, lo que permite a terceros crear complementos que interactúan con la aplicación principal. Un API estable permite que complementos de terceros funcionen como la versión original y amplíen el ciclo de vida de las aplicaciones obsoletas (8).

### 1.3. Metodología de desarrollo de software

Lograr la construcción de un software eficiente, que cumpla con los requerimientos planteados, es una tarea realmente intensa y sobre todo difícil de cumplir. Las metodologías imponen un proceso disciplinado sobre el desarrollo de software con el fin de hacerlo más predecible y eficiente.

Una metodología de desarrollo de software es un conjunto de pasos, procedimientos, técnicas, herramientas y un soporte documental que ayuda a obtener un producto, o sea, el software. Estas suelen seguir uno o varios modelos del ciclo de vida, el cual indica lo que hay que obtener a lo largo del desarrollo del proyecto pero no cómo hacerlo (9).

Las metodologías tienen como principal objetivo aumentar la calidad del software que se produce en todas y cada una de sus fases de desarrollo. Toda metodología debe ser adaptada a las características de cada proyecto o equipo de desarrollo de forma que el proceso sea configurable (9).

Existen dos grupos de metodologías en este tema, las tradicionales y las ágiles, no existe una metodología universal para cada tipo de proyecto, pues esta generalmente se define según las características del equipo de desarrollo, el dominio de aplicación, la complejidad y envergadura del proyecto, todos estos factores hacen necesario adaptar una metodología de desarrollo.

**Metodologías Tradicionales**

Estas metodologías imponen una disciplina de trabajo sobre el proceso de desarrollo del software, con el fin de conseguir un producto más eficiente. Para ello, se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto software. Se centran especialmente en el control del proceso, mediante una rigurosa definición de roles,

actividades, artefactos, herramientas y notaciones para el modelado y documentación detallada. Además, las metodologías tradicionales no se adaptan adecuadamente a los cambios, por lo que no son métodos adecuados cuando se trabaja en un entorno donde los requisitos no pueden predecirse o bien pueden variar (10).

**Metodologías Ágiles**

Este tipo de metodología nace en febrero del 2001 en una reunión celebrada en Utah, EEUU. Constituyen un nuevo enfoque en el desarrollo de software debido a que están especialmente indicadas en proyectos con requisitos poco definidos o cambiantes, lo que exige que el proceso sea adaptable, permitiendo realizar cambios de último momento. Además están orientadas al resultado del producto y no a la documentación; se pueden aplicar bien en equipos pequeños que resuelven problemas concretos donde clientes y desarrolladores trabajan constantemente juntos, estableciéndose así una estrecha comunicación y permitiendo conocer mejor el producto a desarrollar para obtener un software que funcione desde el principio (11).

Analizando los tipos de metodologías antes planteados se decidió usar una metodología ágil teniendo en cuenta que las mismas están preparadas para cambios durante el proyecto, están enfocadas al trabajo en equipo donde se encuentra incluido el cliente y se pueden aplicar bien en equipos pequeños trabajando en el mismo sitio, además de poseer menos artefactos y roles.

### *1.3.1.* Open UP

Open Up es un marco de trabajo centrado en la arquitectura para procesos de desarrollo de software. Está basado en Rational Unified Process (RUP), desarrollado por International Business Machines (IBM). Aplica un enfoque iterativo e incremental dentro de su estructura del ciclo de vida y puede adaptarse para desarrollar diversos tipos de proyectos. Además está dirigido por casos de uso y es aplicable a un conjunto amplio de plataformas y aplicaciones de desarrollo (12).

**Ventajas de Open UP** (12)

- Es apropiado para proyectos pequeños y de bajos recursos ya que permite disminuir las probabilidades de fracaso en los mismos e incrementar las probabilidades de éxito.
- Permite detectar errores tempranos a través de un ciclo iterativo.

- Evita la elaboración de documentación, diagramas e iteraciones innecesarios requeridos en la metodología RUP.
- Tiene un enfoque centrado al cliente y con iteraciones cortas.

### *1.3.2.* Scrum

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar en equipo y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene su origen en un estudio enfocado en la manera de trabajo de equipos altamente productivos.

Más que una metodología de desarrollo, es una forma de auto-gestión para los equipos de programadores. Un grupo de programadores decide cómo hacer sus tareas y cuánto van a tardar en ello. Esta metodología ayuda a que trabajen todos juntos, en la misma dirección, con un objetivo definido. Permite además seguir de forma precisa el avance de las tareas a realizar, de modo que los jefes puedan ver día a día cómo progresa el trabajo.

Usando Scrum se realizan entregas parciales y regulares del resultado final del proyecto, priorizadas por el beneficio que aportan al receptor del mismo. Por ello, está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos y donde la innovación, la competitividad y la productividad son fundamentales.

Esta metodología tiene algunas desventajas, como el hecho de que genera muy poca documentación en comparación con otras metodologías, no es recomendable para todos los proyectos y en muchas ocasiones es necesario completarla con otros procesos de XP (Programación Extrema) (13).

### 1.3.3. Programación Extrema o XP (*Extreme Programming*)

Es una metodología ágil para el desarrollo de software y consiste básicamente en ajustarse estrictamente a una serie de reglas que se centran en las necesidades del cliente para lograr un producto de buena calidad en poco tiempo.

Es una metodología centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo del software. Promueve el trabajo en equipo, preocupándose en todo momento por el aprendizaje de los desarrolladores y estableciendo un buen clima de trabajo.

Este tipo de método se fundamenta en una retroalimentación continuada entre el cliente y el equipo de desarrollo con una comunicación fluida entre todos los participantes, también busca simplificar las soluciones implementadas y coraje para los múltiples cambios. Este tipo de programación es la adecuada para los proyectos con requisitos imprecisos, muy cambiantes y con un riesgo técnico excesivo (13).

La metodología de XP se apoya en (13):

- Pruebas Unitarias: son las pruebas realizadas a los principales procesos, de tal manera que se adelanta en algo hacia el futuro permitiendo hacer pruebas de las fallas que pudieran ocurrir. Es como si se adelantara a obtener los posibles errores.
- Re-fabricación: se desarrolla en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.
- Programación en pares: propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento.

Analizando las metodologías antes plantadas, se decidió usar Open UP, la cual es una metodología ágil, idónea para equipos pequeños de trabajo. Se destaca por ser un proceso iterativo de desarrollo de software simplificado, completo y extensible. Está basada en las mejores prácticas de RUP, las cuales ya han probado su efectividad y se centra en articular la arquitectura para facilitar la colaboración técnica, reducir el riesgo y minimizar el sobreesfuerzo de desarrollo.

## 1.4. Tecnologías y Lenguajes usados en el desarrollo del plugin

Para la elaboración de este plugin se realizó un estudio de las tecnologías y lenguajes usados en la actualidad, con el objetivo de desarrollar el software con la calidad requerida. A continuación se presentan las tecnologías y lenguajes escogidos.

### 1.4.1. Lenguaje de modelado

Como lenguaje de modelado se decide utilizar el UML, el cual es el lenguaje gráfico de modelado orientado a objetos de la industria, para especificar, visualizar, construir y documentar los elementos de los sistemas de software. Proporciona una forma estándar de escribir los planos de un sistema, cubriendo tanto los aspectos conceptuales, (procesos del negocio y funciones del sistema), como los aspectos concretos (las clases escritas en un lenguaje de programación específico, esquemas de bases de datos y

componentes de software reutilizables). Simplifica el proceso complejo de análisis y diseño de software, facilitando un plano para la construcción. También se utiliza para especificar o para describir métodos o procesos así como para definir un sistema, para detallar los artefactos en el mismo y para documentar y construir. En otras palabras, es el lenguaje en que está descrito el modelo (14).

Dentro de las partes de UML se pueden encontrar:

- **Vistas:** Muestran diferentes aspectos de los sistemas que son modelados. Una vista no es un gráfico, pero es una abstracción que consiste en una serie de diagramas. Solamente definiendo una serie de vistas, cada una mostrando un aspecto particular, puede ser construida una imagen completa del sistema. Las vistas también enlazan el lenguaje de modelaje al proceso/método escogido para el desarrollo.
- **Diagramas:** Son los gráficos que describen los contenidos en una vista. El UML tiene nueve tipos de diagramas que son utilizados en combinación para proporcionar todas las vistas del sistema.
- **Mecanismos generales:** Los mecanismos generales proporcionan comentarios extras, información, o semántica acerca de un elemento del modelo; ellos proporcionan también mecanismos de extensión para adaptar o extender el UML a un método, proceso, organización o usuario específico.

**Ventajas de UML** (14)**:**

- Presenta notación específica para modelar procesos de negocio y gracias al metamodelo de UML, los modelos son más robustos y portables.
- Ayuda al reconocimiento rápido de errores, ahorro de tiempo en el desarrollo de software si se ha conseguido modelar un proceso de negocio que represente los objetivos establecidos.
- Facilita la extensibilidad y las modificaciones del modelo, lo que favorece la comunicación de los programadores y modeladores.
- Es un lenguaje ya consolidado y ampliamente conocido, ya que es considerado el estándar del factor para modelado de software y que, como se mencionaba antes, permite modelar con distintos diagramas todo el sistema software completo (estructura y comportamiento) y no solo los procesos de negocio (con diagramas de actividad) que contienen el sistema software a implementar. Esta

capacidad podría hacer que fuera más sencilla la integración de los modelos de negocio en un entorno de desarrollo de software dirigido por modelos.

### 1.4.2. Herramientas de modelado

Existen varias herramientas para el modelado de procesos. A estas también se les conocen como herramientas CASE (*Computer Aided Software Engineering*) las cuales se puede definir como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores, durante todos los pasos del ciclo de vida de desarrollo de un software. Este puede ser generalmente aplicado a cualquier sistema o colección de herramientas que ayudan a automatizar el proceso de diseño y desarrollo de software (15).

***Visual Paradigm***

Visual Paradigm es una de las herramientas UML CASE del mercado, considerada como muy completa y fácil de usar, con soporte multiplataforma y que proporciona excelentes facilidades de interoperabilidad con otras aplicaciones.

Fue creada para el ciclo vital completo del desarrollo de software que lo automatiza y acelera, permitiendo la captura de requisitos, análisis, diseño e implementación. Tiene la capacidad de crear el esquema de clases a partir de una base de datos y viceversa.

Permite invertir código fuente de programas, archivos ejecutables y binarios en modelos UML al instante, creando de manera simple toda la documentación. Está diseñada para usuarios interesados en sistemas de software de gran escala con el uso del acercamiento orientado a objetos. Además apoya los estándares más recientes de las notaciones de Java y de UML. Incorpora el soporte para trabajo en equipo que permite que varios desarrolladores trabajen a la vez en el mismo diagrama y vean en tiempo real los cambios hechos por sus compañeros (15).

**Características del Visual Paradigm** (15) **:**

- Utiliza varios idiomas.
- Resulta fácil de instalar y actualizar.
- Permite la compatibilidad entre ediciones.
- Se integra con distintos IDE como por ejemplo el NetBeans.

- Apoya todo lo básico en cuanto a artefactos generados en las etapas de definición de requerimientos y de especificación de componentes.
- Tiene apoyo adicional en cuanto a la generación de artefactos automáticamente.
- Permite la generación de documentación en formatos HTML y PDF.
- Brinda disponibilidad en múltiples plataformas como Microsoft Windows (98, 2000, XP, o Vista), Linux, Mac OS X, Solaris o Java.
- Brinda la posibilidad de intercambiar información mediante la importación y exportación de ficheros con aplicaciones como por ejemplo Visio y Rational Rose.
- Permite la generación de código e ingeniería inversa.
- Permite la generación de documentación.

Se decide utilizar Visual Paradigm en su versión 8.0 como herramienta CASE para el modelado del sistema a desarrollar teniendo en cuenta que es fácil de usar y es multiplataforma. Además es capaz de integrase con distintos IDEs como el Netbeans. Soporta múltiples usuarios trabajando sobre el mismo proyecto; genera la documentación del proyecto automáticamente en varios formatos y permite control de versiones. Es muy eficiente ya que su uso posibilita una construcción de aplicaciones rápida, con buena calidad y a un menor coste.

### 1.4.3. Lenguajes de programación

Un lenguaje de programación es un lenguaje diseñado para describir el conjunto de acciones consecutivas que un equipo debe ejecutar. Consiste en un conjunto de reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos, respectivamente (16).

- **Java**

Se toma la decisión de utilizar Java debido a que es el lenguaje estándar de Pentaho BI Server, además asegura la integración eficiente con las API que posee.

Java se puede definir como un lenguaje de programación de alto nivel orientado a objetos, creado por Sun Microsystems, que está sustentado en cinco pilares: la programación orientada a objetos, la posibilidad de ejecutar un mismo programa en diversos sistemas operativos, la inclusión por defecto de soporte para trabajo en red, la opción de ejecutar del código en sistemas remotos de manera segura y la facilidad de

uso.

Es un lenguaje que evoluciona a partir de C y C++, bien estructurado y fácil de aprender. Las aplicaciones son fiables, puede controlarse su seguridad frente al acceso a recursos del sistema y es capaz de gestionar permisos y criptografía. Además es un lenguaje compilado e interpretado y actualmente es uno de los más usados para la programación en todo el mundo. Se utiliza en una gran variedad de dispositivos móviles, como teléfonos y pequeños electrodomésticos, en los navegadores web, donde permite desarrollar pequeñas aplicaciones conocidas como applets que se incrustan en el código HTML de las páginas. También funciona perfectamente en red, aprovecha las características de la mayoría de los lenguajes modernos evitando sus inconvenientes, en particular los del C++. Tiene una gran funcionalidad gracias a sus librerías, genera aplicaciones con pocos errores y permite la ejecución de tareas concurrentes dentro de un mismo programa (17).

- **Java script**

Para programar la navegación en el lado del cliente de forma tal que la interfaz sea más dinámica, se decide utilizar JavaScript el cual se puede definir como un lenguaje multiplataforma, *open source,* interpretado y basado en objetos, que se utiliza para construir sitios Web más interactivos. Fue desarrollado por Netscape, a partir del lenguaje Java aunque no están relacionados. Nace debido a la necesidad de permitir a los desarrolladores de sitio web crear páginas que permitieran intercambiar con los usuarios, debido a que HTML solo permitía crear páginas estáticas donde solo se podían mostrar textos con estilos y se necesitaba crear webs de mayor complejidad.

Es un lenguaje pequeño y ligero; no es útil como un lenguaje independiente, más bien está diseñado para una fácil incrustación en otros productos y aplicaciones, tales como los navegadores Web.

El lenguaje JavaScript se inserta en documentos HTML, de forma que su código queda reflejado en la propia página y no es llamado o cargado de ninguna fuente externa. Es sensible a mayúsculas, aunque algunas implementaciones ignoran en parte este último extremo y no requiere de compilación ya que el lenguaje funciona del lado del cliente, los navegadores son los encargados de interpretar estos códigos. (18)

### 1.4.4. Lenguajes de marcado.

Un lenguaje de marcado es una forma de codificar un documento que, junto con el texto,

incorpora etiquetas o marcas que contienen información adicional acerca de la estructura del texto o su presentación. Los lenguajes de marcado suelen confundirse con lenguajes de programación, sin embargo, no son lo mismo, ya que el lenguaje de marcado no tiene funciones aritméticas o variables, que sí poseen los lenguajes de programación.

- **HTML**

Para diseñar y estructurar las páginas web del plugin a desarrollar, se decide utilizar HTML el cual se puede definir como un lenguaje de marcado orientado a la publicación de documentos en Internet. Es una aplicación del Lenguaje de Etiquetado Generalizado Estándar (SGML) y es considerado como el lenguaje de publicación estándar de la *Word Wide Web (WWW).*

El lenguaje HTML nace en 1991 de manos de Tim Bernes-Lee, fue concebido como un lenguaje para el intercambio de documentos científicos y técnicos adaptado para su uso por no especialistas en tratamiento de documentos. Resolvió el problema de la complejidad de SGML sirviéndose de un reducido conjunto de etiquetas estructurales y semánticas apropiadas para la realización de documentos relativamente simples. Pero, además de simplificar la estructura de los documentos, soportaba el hipertexto.

El código HTML permite insertar menús, tablas, imágenes en los documentos, pero no permite al usuario que maneje esos datos como mejor le convenga con la poderosa ayuda del ordenador (19).

- **XML** (Lenguaje de Etiquetado Extensible)

XML es un lenguaje muy simple, creado al amparo del *Word Wide Web Consortium* (W3C) organismo que vela por el desarrollo de WWW partiendo de las amplias especificaciones de SGML. No es más que un conjunto de reglas para definir etiquetas semánticas que nos organizan un documento en diferentes partes. Es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados. Este lenguaje es abierto, derivado del SGML, optimizado para su uso en la Web y que permite describir el sentido de los datos (20).

Es un lenguaje muy similar a HTML pero su función principal es describir datos y no mostrarlos como es el caso de HTML. Permite que los diseñadores creen sus propias etiquetas, permitiendo la definición, transmisión, validación e interpretación de datos entre aplicaciones y entre organizaciones.

### *1.4.5.* Framework de desarrollo

- **JQuery**

JQuery es una biblioteca JavaScript rápida, pequeña y rica en funciones. Permite simplificar la manera de interactuar con los documentos HTML, además el recorrido y la manipulación, el control de eventos, animación y Ajax son mucho más simples debido a una API fácil de usar que funciona a través de una multitud de navegadores. Con una combinación de versatilidad y capacidad de ampliación, jQuery ha cambiado la forma en que millones de personas escriben JavaScript (21).

- **YAML**

YAML ("*Yet Another Multicolumn Layout*" o Lenguaje de Marcas Extensible) es un framework de desarrollo para CSS de código abierto. Ofrece una estructura extremadamente versátil. Ayuda a los diseñadores a realizar hojas de estilo de una manera más ágil y estándar para los múltiples navegadores. Se encuentra centrado en estándares web y en su accesibilidad. Posee modelos de diseño para formularios, tipografía, microformatos, etc (22).

### 1.4.6. Entorno de Desarrollo Integrado (IDE)

Un Entorno de Desarrollo Integrado (IDE), es un programa compuesto por un conjunto de herramientas que proveen facilidades a los programadores para agilizar el proceso de desarrollo de software. Es un entorno de programación que ha sido empaquetado como un programa de aplicación. Consiste en un editor de código, un compilador, un depurador y un Entorno de Interfaz Gráfica de Usuario (GUI), algunas veces contiene también un sistema para llevar a cabo el control de versiones (23).

Muchos de los IDE modernos llevan integrados además un navegador de clases, un inspector de objetos y diagramas de herencia de clases para ser usados en el desarrollo orientado a objetos.

**Principales características de los IDE** (23)**:**

- Multiplataforma.
- Soporte para diversos lenguajes de programación.
- Integración con sistemas de control de versiones.
- Reconocimiento de sintaxis.
- Extensiones y componentes para el IDE.

- Integración con framework populares.
- Depurador.
- Importación y exportación de proyectos.
- Múltiples idiomas.
- Manual de usuarios y ayuda.

#### 1.4.6.1. Eclipse

El Eclipse es un entorno de desarrollo integrado que facilita enormemente las tareas de edición, compilación y ejecución de programas durante su fase de desarrollo. Es una potente y completa plataforma de programación de código abierto y multiplataforma creada por IBM. Posee una interfaz fácil y agradable de usar donde se encuentran todas las herramientas y funciones necesarias para desarrollar un software. Sirve como IDE de Java aunque también da soporte a otros lenguajes de programación, como son C/C++, Cobol, Fortran, PHP o Python. En el caso de Java dispone del Java Development Kit (JDK) para poder compilar y ejecutar las aplicaciones desarrolladas (24).

**Objetivos de la plataforma Eclipse** (25)**:**

La plataforma Eclipse está diseñada para afrontar las siguientes necesidades:

- Soportar la construcción de gran variedad de herramientas de desarrollo.
- Soportar herramientas proporcionadas por diferentes fabricantes de software independientes.
- Soportar herramientas que permitan manipular diferentes contenidos (HTML, Java, C/C++, JSP, XML...).
- Facilitar una integración transparente entre todas las herramientas y tipos de contenidos sin tener en cuenta al proveedor.
- Proporcionar interfaces gráficas y no gráficas de usuario.
- Compatibilizar con una gran variedad de sistemas operativos, incluyendo Windows y Linux.
- Fomentar el uso de Java para el desarrollo de nuevos plugins.

El principal objetivo de la Plataforma Eclipse es proporcionar mecanismos y reglas que permitan a los desarrolladores integrar de manera transparente sus herramientas. Estos mecanismos son

proporcionados a través de APIs (25).

#### 1.4.6.2. Netbeans

Es un entorno de desarrollo integrado, hecho principalmente para el lenguaje Java (y todos sus derivados) aunque una vez instalado se pueden descargar los complementos para programar en C++, además ofrece un excelente entorno para programar en PHP. Es un producto libre y gratuito, sin restricciones de uso, de gran éxito y con una extensa base de usuarios. Tiene un excelente balance entre una interfaz con múltiples opciones y un aceptable completamiento de código y todas las funciones del IDE son provistas por módulos.

**Ventajas del Netbeans** (26)**:**

Permite trabajar con un gran número de lenguajes como: C/C++, Ruby, PHP, JSP, entre otros.

- Posee una interfaz muy sencilla de usar.
- Es de instalación y actualización simple.
- Tiene un rápido diseño de interfaz para desarrollo.
- Es multiplataforma.
- Permite la reutilización de módulos.

**Desventajas del Netbeans** (26)**:**

- Demanda más consumo de memoria que otras IDE.
- Existe poca disponibilidad de plugins para esta plataforma.

Después de ser analizados los IDE anteriores se propone utilizar Netbeans en su versión 7.1 para el desarrollo de la solución, aunque ambos son de gran potencia y muy utilizados en el ámbito mundial. Netbeans es completamente compatible con el lenguaje que se va a usar para la implementación del plugin, permitiendo completamiento de código y la integración con el entorno de desarrollo. Además es multiplataforma y ha sido utilizado en múltiples ocasiones por los desarrolladores, lo que trae consigo un mejor dominio del mismo.

**Conclusiones del capítulo**

En este capítulo se realizó un análisis detallado de los conceptos fundamentales para darle solución a la problemática planteada, además de un estudio de las herramientas que permiten asegurar la seguridad de

los esquemas OLAP en la actualidad. También se realizó un estudio de las metodologías, los lenguajes y tecnologías de desarrollo que se pueden usar para hacer más eficaz la solución, obteniendo como resultado:

- Para el desarrollo del plugin se propone como metodología de desarrollo la Open UP, una metodología ágil basada en las mejores prácticas de RUP e idónea para equipos pequeños de trabajo.
- El lenguaje de modelado elegido fue el UML, este es un lenguaje gráfico para visualizar, especificar, construir y documentar cada una de las partes que comprende el desarrollo del proyecto.
- Como herramienta para realizar el modelado de procesos del negocio se eligió el Visual Paradigm para UML en su versión 8.0, el cual es multiplataforma y es considerado una herramienta muy potente.
- Por último, para desarrollar el plugin se escogió el IDE NetBeans en su versión 7.1, debido que soporta e integra completamente el lenguaje Java el cual será el utilizado para el desarrollo del plugin.

# Capítulo 2: Análisis y Diseño

### Introducción

En este capítulo se muestra todo lo relacionado con el flujo de trabajo análisis y diseño. Se realiza además el levantamiento de los requerimientos funcionales y no funcionales que el sistema debe cumplir y se lleva a cabo la representación y descripción de caso de uso del sistema. También se muestra el diseño de la solución propuesta, se realiza un análisis de los patrones de diseño y arquitectónico a usar, para de esta forma sentar las bases de la implementación.

### 2.1. Propuesta de la solución

Como propuesta de solución se presenta la creación de un plugin que permita asegurar la seguridad de los esquemas OLAP publicados en Pentaho, permitiendo asignar diferentes permisos a los usuarios de acuerdo al rol que posea, lo cual hace menos engorroso el trabajo y menos costoso en tiempo esta tarea.

El plugin debe mostrar un listado de roles y uno de catálogos existentes en el Pentaho, además debe ser capaz de visualizar los permisos que posee un rol seleccionado sobre un catálogo específico, con el fin de poder gestionar dichos permisos y guardar los cambios de forma tal que perduren en el sistema.

### 2.2. Modelo de Dominio

El Modelo de Dominio o Modelo Conceptual es una representación visual de los conceptos u objetos del mundo real, significativos para un problema o área de interés, representa clases conceptuales del dominio del problema. Puede utilizarse para capturar y expresar el entendimiento obtenido en un área bajo análisis como paso previo al diseño de un sistema, ya sea de software o de otro tipo (27).

Para la elaboración del modelo de dominio se definieron varias entidades las cuales serán explicadas a continuación:

**Administrador:** Rol que debe poseer el usuario que acceda al Pentaho BI Server y quiera acceder al plugin.

**Pentaho BI Server:** Herramienta de BI sobre la que va a estar integrada la solución a desarrollar.

**Plugin:** Funcionalidad desarrollada que permite dar solución al problema planteado inicialmente.

**Roles:** Listado donde se encuentran todos los roles que pueden poseer los usuario dentro del sistema.

**Esquemas:** Son los contenedores de los cubos OLAP, un esquema puede contener la cantidad de cubos que se desee, estos cubos poseen dimensiones y medidas, dentro de las dimensiones se encuentran las jerarquías y dentro la jerarquías los niveles.

**Esquema XML:** Fichero donde se guarda los esquemas con los permisos asignados y los roles que tienen acceso a la información que poseen dichos esquemas.

**Permisos de acceso:** Son los permisos asignados a un determinado esquema permitiendo que solo se pueda ver la información que se desee.

**Nivel de registro:** Cuando ser carga los esquemas solo se puede acceder hasta los niveles del cubo, sin embargo estos niveles poseen miembros, los cuales pueden tener hijos y de esta forma seguir desencadenando una larga cadena. A esta información se le conoce como datos a nivel de registro.

**Nuevos cambios:** Una vez gestionados los permisos es necesario actualizarlos en todo el sistema, por lo que se refresca el repositorio y se vacía la caché de los esquemas de Mondrian.

**Vista de Análisis:** Es donde se visualiza toda la información referente a un cubo elegido dentro de un esquema seleccionado.

En la figura 1 se evidencia como el Administrador del sistema mediante el Pentaho BI Server es el único que puede acceder al plugin. El Pentaho BI Server contiene los roles de los usuarios que pueden acceder al sistema y los esquemas con los cubos OLAP modelados. Además posee un esquema XML donde se encuentran almacenados los permisos de acceso que tienen concedidos estos esquemas. Estos permisos de acceso se encuentran asignados hasta nivel de registro y son gestionados mediante el plugin permitiendo crear nuevos permisos, modificar o eliminar los existentes. El plugin también proporciona actualizar los nuevos cambios en el sistema posibilitando hacer visible en el Pentaho BI Server los cambios realizados mediante la Vista de Análisis.

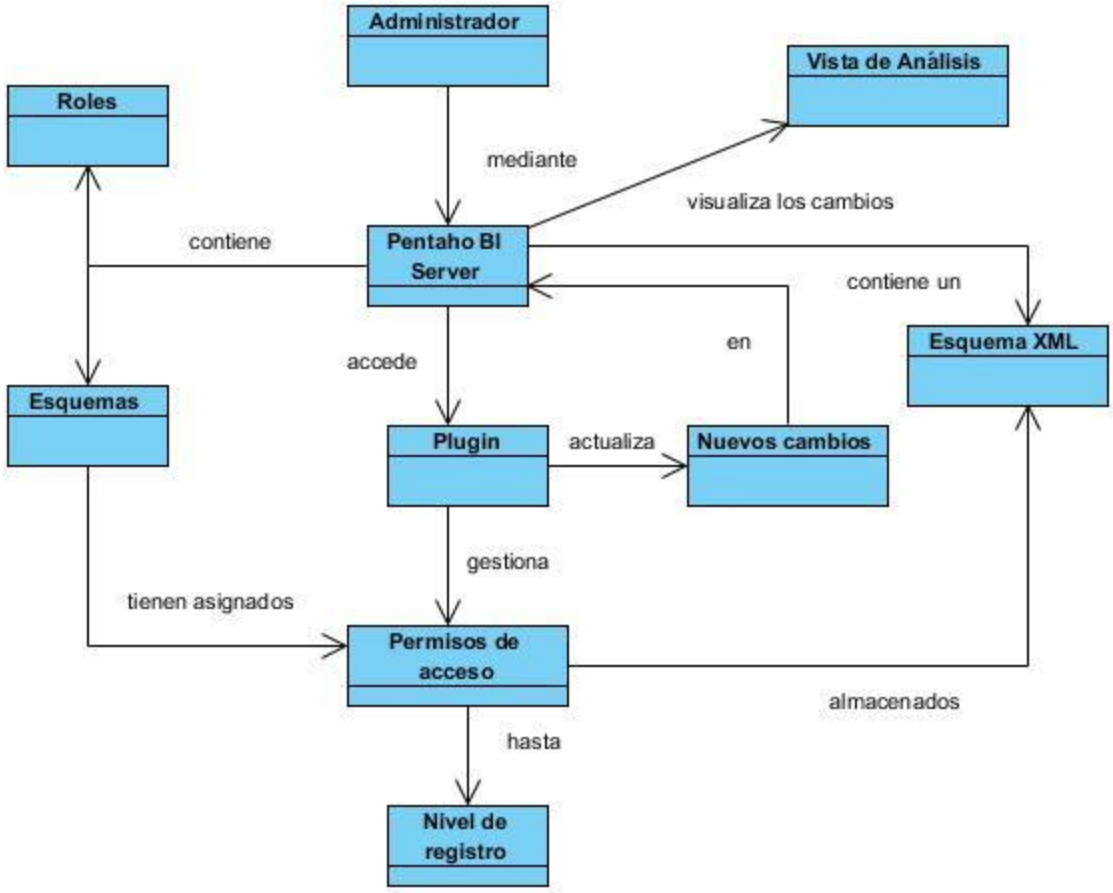


**Figura 6: Diagrama de Dominio del sistema**

### 2.3. Especificación de los requisitos del sistema

En la especificación de requisitos del software se realiza una descripción completa del comportamiento del sistema a desarrollar. Contiene los requisitos funcionales (Rf) y no funcionales (Rnf) que debe cumplir el sistema y los casos de uso que describen todas las interacciones que se prevean que los usuarios tendrán con el software.

#### 2.3.1 Requisitos Funcionales

Los requisitos funcionales no son más que los encargados de describir las funcionalidades o servicios que proveerá el sistema (28), o sea las propiedades o cualidades que se relacionen. En vista al desarrollo del presente trabajo se obtuvieron los siguientes:

**Rf 1** Obtener roles

**Rf 2** Obtener catálogos.

**Rf 3** Obtener permisos a nivel de registro.

**Rf 4** Crear permisos

**Rf 5** Eliminar permisos

**Rf 6** Modificar permisos

**Rf 7** Refrescar el repositorio del Pentaho BI Server

**Rf 8** Vaciar caché de los esquemas de Mondrian

**Rf 9** Restablecer permisos del sistema

### 2.3.2 Requisitos no funcionales

Los requerimientos no funcionales son las propiedades emergentes que va a poseer el sistema, (28) estos harán del mismo un producto seguro, confiable y de respuesta rápida.

**Rnf1.** ***Requisito de Usabilidad***

- El funcionamiento del sistema será intuitivo y requerirá de conocimientos mínimos para su uso.

**Rnf2.** ***Requisito de Interfaz***

- La interfaz de usuario debe ser ligera y de fácil uso.

**Rnf3.** ***Requisito de Disponibilidad:***

- La información y las funcionalidades estarán disponibles y el administrador podrá acceder a ellas en todo momento.

**Rnf4.** ***Requisito de legalidad***

- Las herramientas seleccionadas están respaldadas por licencias libres, bajo las condiciones de software libre.

**Rnf5.** ***Seguridad***

- Solo accederán al sistema los usuarios que posean el rol de administrador.

**Rnf6.** ***Requisito de Hardware***

- Para el cliente:

  Requerimientos mínimos: Procesador Pentium III con 512MB de memoria RAM.

- Para el servidor:

  Requerimientos mínimos: Procesador Pentium Dual-Core y 2 GB de memoria RAM.

**Rnf7.** ***Requisito de Software***

- Para el cliente:
  - Usar preferiblemente el Navegador Web Firefox
- Para el servidor
  - Tener instalada una Máquina Virtual de Java (JDK)

### 2.3.3 Actores del sistema

Los actores representan a personas, software o hardware fuera del sistema que interactúan con él. En el sistema que se describe se identificaron los siguientes actores.

**Tabla 1: Actores del Sistema**

| *Actor* | *Objetivo* |
|---|---|
| Administrador | Representa al usuario que va a hacer uso del sistema, teniendo la posibilidad de interactuar con las funcionalidades que brinda el plugin. |

### 2.3.4 Diagrama de casos de uso del sistema

Para la realización del diseño se deben conocer las características principales del sistema especificando las diferentes acciones que debe permitir y los actores que intervienen en las mismas. Las relaciones que se establecen entre estos dos elementos se reflejan en el siguiente diagrama de casos de uso del sistema. Este representa gráficamente a los procesos y su interacción con los actores.

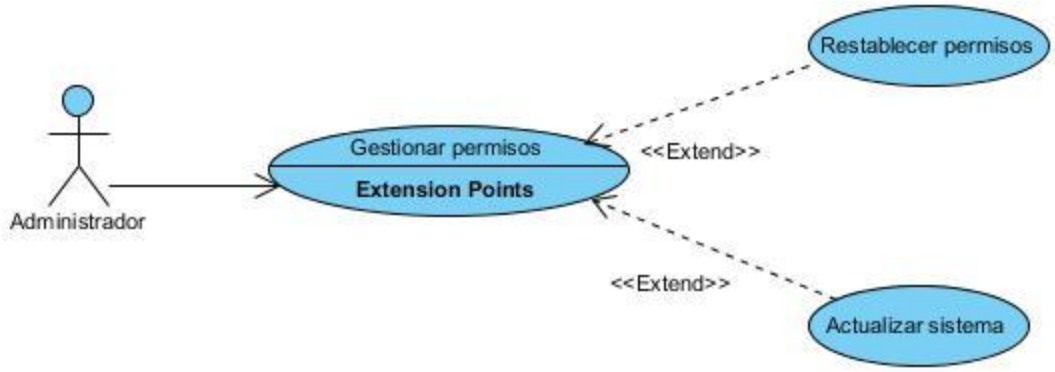


**Figura 7: Diagrama de Casos de Uso del Sistema**

### 2.3.5 Descripción de los casos de uso

Especificar los casos de usos (CU) permite que se adquieran detalles de estos para comprender las necesidades del cliente y provee un conocimiento para comenzar a desarrollar el sistema. A continuación se describirá el CU ***Gestionar permisos*** por ser uno de los que más impacto tiene en la arquitectura del sistema. Para ver las descripciones de los restantes CU se debe consultar el Anexo A.

**Tabla 2: Descripción del CU Gestionar permisos**

| | |
|---|---|
| **Objetivo** | El usuario puede gestionar los permisos de acceso de los roles existentes a los cubos OLAP. |
| **Actores** | Administrador |
| **Resumen** | El Caso de Uso inicia cuando el sistema lista los roles y catálogos existentes en el Pentaho BI Server. Luego el usuario puede gestionar los permisos y el caso de uso termina cuando el sistema guarda los cambios. |
| **Complejidad** | Alta |
| **Prioridad** | Crítico |
| **Precondiciones** | El usuario debe estar autenticado como administrador en el sistema. |

| **Poscondiciones** | | |
|---|---|---|
| **Flujo de eventos** | | |
| **Flujo básico: Gestionar permisos** | | |
| | **Actor** | **Sistema** |
| 1. | | a. El sistema muestra un combo box con un listado de roles. |
| 2. | | a. El sistema muestra un combo box con un listado de catálogos. |
| 3. | a. El usuario selecciona un rol y selecciona un catálogo. | b. El sistema muestra el esquema seleccionado y además muestra los permisos que posee. |
| 4. | *a.* El usuario mediante checkboxs puede adicionar nuevos permisos o, modificar o eliminar los existentes, luego selecciona el botón **"**Asignar Permisos"**.** | *b.* El sistema verifica que se haya seleccionado un rol, *c.* El sistema verifica que existan permisos asignados y manda un mensaje informando sobre su existencia y si desea modificarlos. |
| 5 | a. El usuario acepta el mensaje mostrado. | b. El sistema guarda los datos y emite un mensaje informando que se guardaron correctamente los permisos asignados, terminándose así el CU. |
| **Flujos alternos** | | |

| 4.b. No se ha seleccionado un rol | | |
|---|---|---|
| | **Actor** | **Sistema** |
| 1. | | Se emite un mensaje para que seleccione un rol de la lista de roles existentes y regresa a la acción 1.a. |

### 2.4. Diagrama de clases del diseño

El diagrama de clases es el diagrama principal para el análisis y diseño. Un diagrama de clases presenta las clases del sistema con sus relaciones estructurales y de herencia. La definición de clase incluye definiciones para atributos y operaciones. El modelo de casos de uso aporta información para establecer las clases, objetos, atributos y operaciones. (29)

En la figura 3 se evidencia el diagrama de clases del diseño correspondiente al CU Gestionar permisos, donde la clase controladora GSEO_ContentGenerator es la encargada de construir la clase cliente index, esta posee tres formularios: Roles, Catálogos y Permisos, quienes envían la información contenida a la clase GSEO_ContentGenerator. Esta consulta a las clases MondrianHelper, Leer_XML y Escribir_XML obteniendo la información necesaria para realizar el CU.

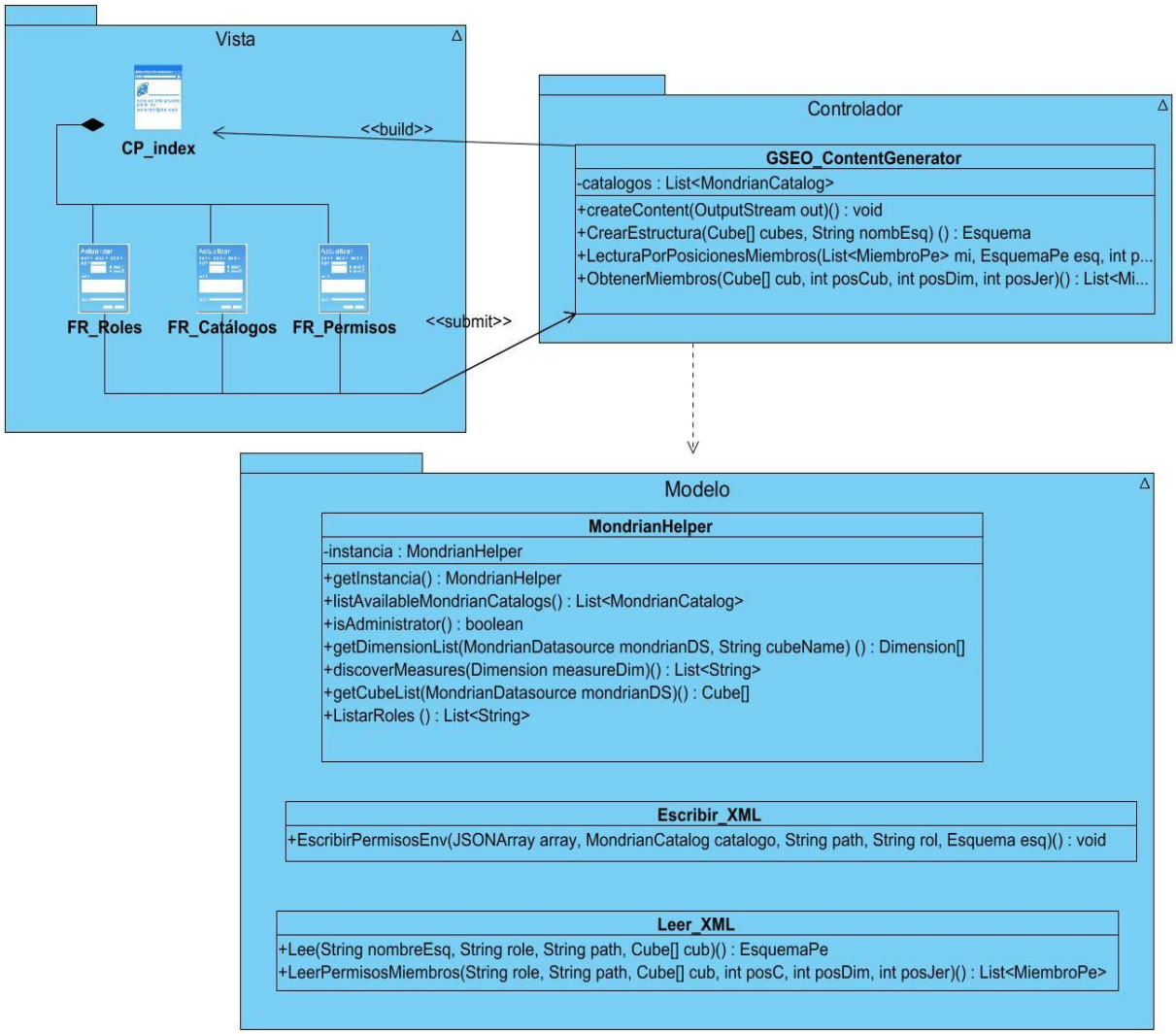


**Figura 8: Diagrama de clases del diseño CU Gestionar Permisos**

A continuación se explican las principales clases:

**GSEO_ContentGenerator:** Es la clase encargada de controlar toda la información tanto del lado del cliente como la del lado del servidor.

**MondrianHelper:** Permite obtener los roles, catálogos, y toda la información de los esquemas.

**Leer_XML:** Lee del esquema XML los permisos que posee un esquema.

**Escribir_XML:** Escribe en el esquema XML los permisos asignados a un determinado esquema.

## 2.5. Patrones arquitectónicos

Los patrones arquitectónicos se encargan de definir la estructura de un sistema, estos a su vez se componen de subsistemas con sus responsabilidades, también tienen una serie de directivas para organizar los componentes de la misma aplicación, con el objetivo de facilitar la tarea del diseño del sistema. Un patrón arquitectónico se enfoca a dar solución a un problema en específico, de un atributo de calidad, y abarca solo parte de la arquitectura. (30)

### 2.5.1. Modelo Vista Controlador (MVC)

El estilo arquitectónico que se definió en el desarrollo del plugin de Seguridad fue el Modelo Vista Controlador (MVC), el cual es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos usualmente. (31)

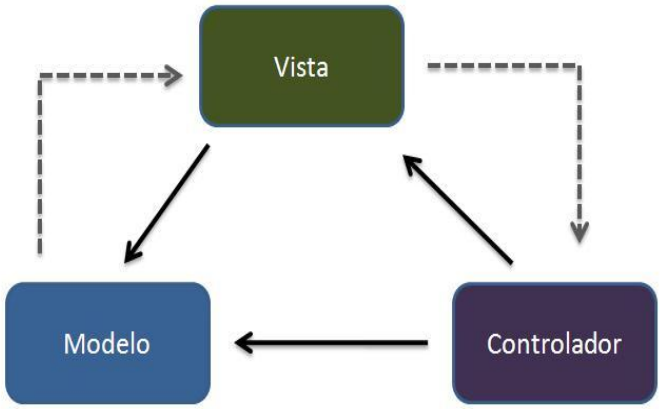


**Figura 9: Patrón Arquitectónico MVC**

A continuación se explica con mayor claridad la funcionalidad de cada componente:

**Modelo:** Es la representación de la información en el sistema. Envía a la vista aquella parte de la información que en cada momento se le solicita para que sea mostrada. Las peticiones de acceso o manipulación de la información llegan al modelo a través del controlador.

**Vista:** Es la que presenta al modelo en un formato adecuado para que el usuario pueda interactuar con él, casi siempre es la interfaz de usuario.

**Controlador:** Es el elemento más abstracto. Hace de intermediario entre la vista y el modelo. Recibe, trata y responde los eventos enviados por el usuario o por la propia aplicación.

## 2.6. Patrones de diseño

Un patrón es una solución de diseño de software a un problema, aceptada como correcta, a la que se ha dado un nombre y que puede ser aplicada en otros contextos (32).

"Cada patrón describe un problema que ocurre una y otra vez en nuestro entorno, para describir después el núcleo de la solución a ese problema, de tal manera que esa solución pueda ser usada más de un millón de veces sin hacerlo ni siquiera dos veces de la misma forma". (32)

Son un mecanismo popular para describir soluciones generales de problemas de diseño que pueden ser reutilizadas en la construcción de aplicaciones. Cada patrón prescribe una estructura de clases, sus roles y colaboraciones, y una adecuada asignación de métodos para resolver un problema de diseño en una manera flexible y adaptable (32). Entre los patrones que se utilizaron en el diseño de la solución se encuentran:

### 2.6.1. Patrones GOF

Los patrones de diseño GOF son conocidos como la pandilla de los cuatro, formada por Erich Gamma, Richard Helms, Ralph Johnson y John Vlissides, autores del famoso libro "Design Patterns: Elements of Reusable Object Oriented Software los cuales clasificaron los patrones en tres grandes categorías: (33)

- **Creacionales**: que abstraen el proceso de creación de instancias.
- **Estructurales:** se ocupan de cómo clases y objetos son utilizados para componer estructuras de mayor tamaño.
- **Comportamiento:** se centran en la comunicación entre las distintas clases y objetos.

En la solución en desarrollo se hizo uso del patrón creacional **Singleton** el cual garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia. Este patrón se pone de manifiesto en la clase MondrianHelper.

### 2.6.2. Patrones GRASP

Los patrones GRASP (Patrones para Asignar Responsabilidades) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma estructurada. (34) Entre los patrones GRASP que se utilizan en el diseño de la solución están:

- **Experto en información**

Es utilizado fundamentalmente en el diseño orientado a objetos. Con él se pretende asignar la responsabilidad de realizar una tarea determinada a aquel objeto que tiene la información necesaria para ello. Este patrón se evidencia mediante las clases GSEO_ContentGenerator, MondrianHelper, Leer_XML y Escribir_XML, mostrado en la Figura 5.

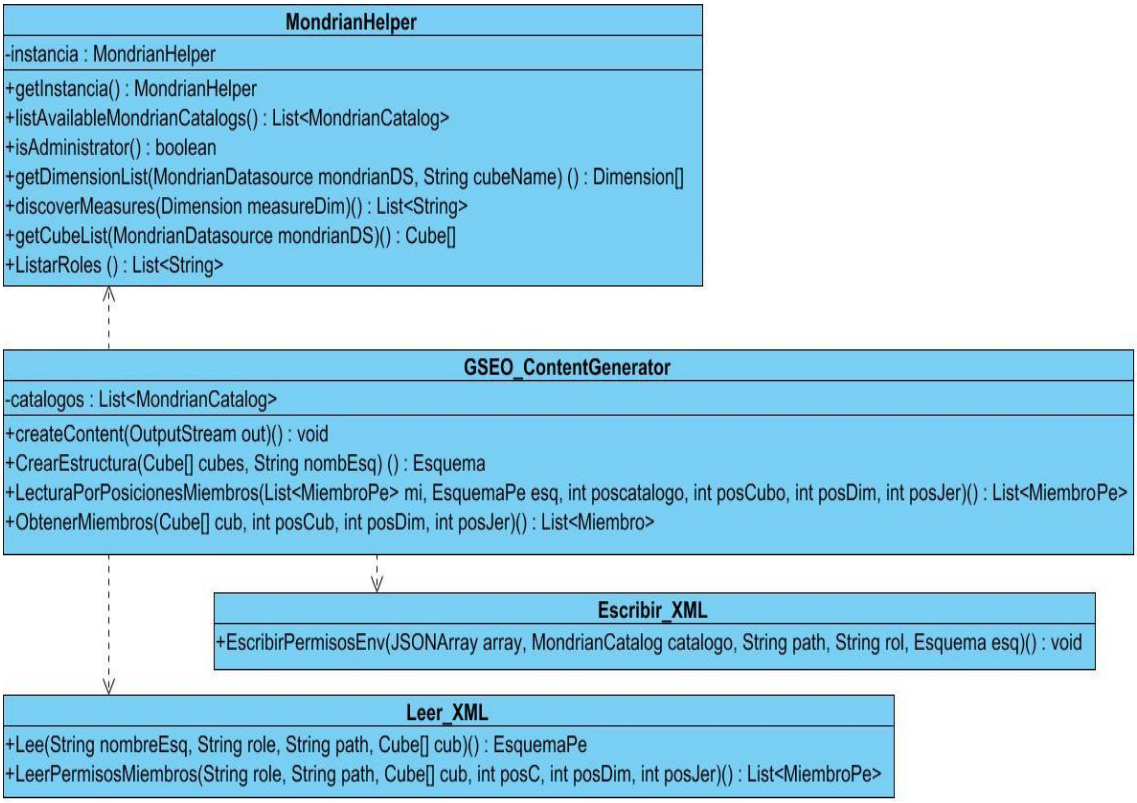


**Figura 10: Patrón GRAPS Experto en Información**

- **Creador**

  Este patrón maneja la asignación de responsabilidades relacionadas con la creación de objetos. El principal objetivo de este patrón es encontrar un creador que se debe conectar con el objeto producido en un evento cualquiera, lo que se puede ver mediante la clase GSEO_ContentGenerator, en la cual se crean las instancias para acceder a las clases de MondrianHelper, Leer XML y escribir XML.

- **Controlador**

Este patrón se asocia con las operaciones del sistema, es el encargado de asignar la responsabilidad del

manejo de los eventos de un sistema a una clase que represente un sistema global, en el caso de la solución presentada. Normalmente un controlador delega en otros objetos, el trabajo que se necesita hacer; coordina o controla la actividad. En el diseño de la solución la clase GSEO_ContentGenerator es la encargada de controlar toda la información recibida tanto del lado cliente como del lado del servidor, lo cual se evidencia a través de la figura 5.

- **Alta cohesión**

La cohesión es una medida de la fuerza con la que se relacionan las clases y el grado de focalización de las responsabilidades de un elemento. Cada elemento del diseño debe realizar una labor única dentro del sistema no desempeñado por el resto de los elementos y auto-identificable. Una clase con baja cohesión hace muchas cosas no relacionadas o hace demasiado trabajo. Este patrón se evidencia mediante la Figura 5 donde cada clase tiene la responsabilidad de controlar la información que posee.

### 2.7. Diagramas de Interacción

Los diagramas de interacción describen secuencias de intercambios de mensajes entre los roles que implementan el comportamiento de un sistema. Explican gráficamente las interacciones existentes entre las instancias y las clases del sistema. proporcionando una vista integral de su comportamiento. (34) Existen dos tipos de estos diagramas, ambos basados en la misma información pero cada uno enfatizando un aspecto particular, ellos son:

- Diagrama de Secuencias
- Diagrama de Colaboración

#### 2.7.1. Diagramas de Secuencia

El diagrama de secuencia es uno de los diagramas más efectivos para modelar interacción entre objetos en un sistema. El mismo muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada método de la clase. Este contiene detalles de implementación del escenario, incluyendo los objetos y clases que se usan para implementarlo. (34)

Para modelar la interacción entre los objetos del plugin se le realizó el diagrama de secuencia correspondiente al CU Gestionar permisos, el cual se puede encontrar en los Anexos, Figura 19.

**Conclusiones del capítulo**

En el presente capítulo se definieron los elementos fundamentales del análisis y diseño, se analizó la propuesta de solución basada en el modelo del dominio y se definieron los requisitos funcionales y no funcionales que debe cumplir el plugin. Se identificaron los actores involucrados y se describieron los casos de uso del sistema. Además se seleccionaron los patrones de diseño y arquitectónicos a utilizar, definiéndose como patrón arquitectónico el MVC, logrando con esto sentar las bases para la implementación de la solución. Por último se definió el diagrama de clases del diseño y los diagramas de interacción para el CU: “Gestionar Permisos” el cual es considerado un caso de uso arquitectónicamente significativo.

# Capítulo 3: Implementación y Prueba

## Introducción

En este capítulo se expone todo lo relacionado con los flujos de trabajo implementación y pruebas. Se describirá la implementación del sistema en términos de componentes y la manera en que estos componentes serán desplegados, además de los estándares de codificación propuestos. También se mostrarán las pruebas realizadas al software desarrollado, con el objetivo de comprobar las funcionalidades del plugin en los diferentes escenarios, verificando en todos los casos que los resultados de las pruebas sean los esperados.

## 3.1. Diagrama de componentes

Los diagramas de componente modelan la vista estática de un sistema y describen los elementos físicos, los mismos pueden ser archivos, librerías, módulos, ejecutables, o paquetes y pueden ser usados para modelar y documentar cualquier arquitectura de sistema. Además en los diagramas de componentes se muestran los elementos de diseño de un sistema y por otra parte permiten 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. (35)

A continuación se muestra el diagrama de componentes del caso de uso Gestionar permisos donde se definieron tres paquetes los cuales se mostrarán seguidamente de forma individual.

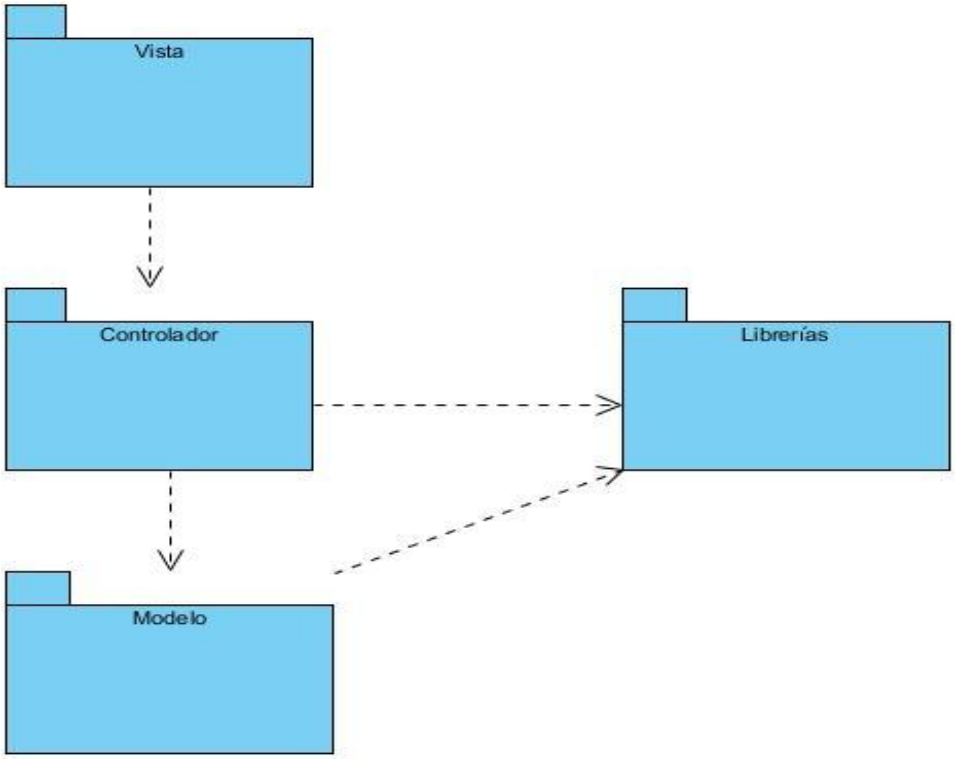


**Figura 11: Diagrama de componentes**

En la figura 7 se muestra el paquete Vista el cual está compuesto por los siguientes componentes:

**Index.html:** Clase principal sobre la que se va a mostrar toda la información.

**Arbol.js:** Clase que muestra un formulario con la estructura del esquema seleccionado en formal de árbol.

**Mensaje.js:** Clase encargada de mostrar todos los mensajes generados en el sistema.

**Refrescar-caché.js:** Clase encargada de vaciar la caché y reiniciar el sistema con la nueva información.

**Jquery-1.7.2.min.js**: Envía las peticiones del cliente hacia la clase controladora y se encarga de garantizar el acceso de los elementos html, así como de personalizar la interfaz.

**roles.js:** Esta clase muestra el listado de roles existentes en el Pentaho BI Server.

**catalogo.js:** Esta clase muestra el listado de catálogos existentes en el Pentaho BI Server.

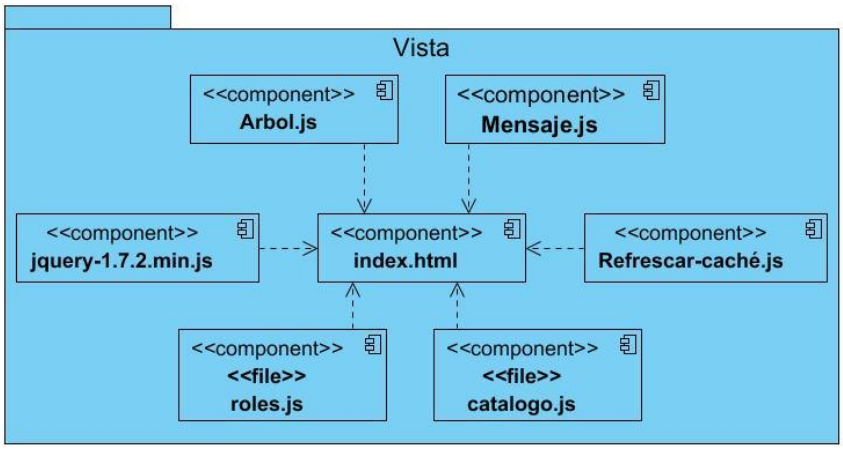


**Figura 12: Componentes del paquete Vista**

El paquete Controlador se puede evidenciar en la figura 8 el cual está compuesto por la clase controladora **GSEO_ContentGenerator.java,** la cual implementa los métodos necesarios para dar las respuestas al cliente, para esto hace uso de las siguientes librerías:

- mondrian-3.4.1.jar
- pentaho-bi-plataform-plugin-action-4.5.0-stable.jar
- xstream-1.4.4.jar
- json-simple-1.1.1.jar

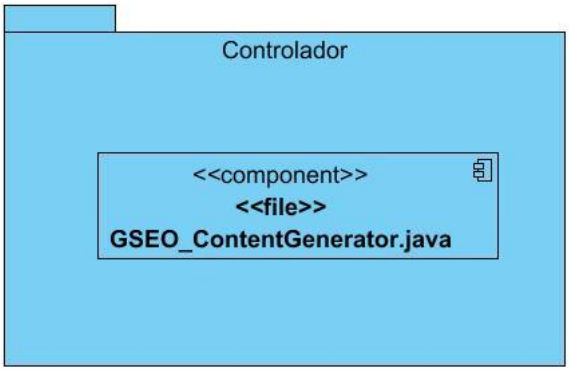


**Figura 13: Componentes del paquete Controlador**

En la figura 9 se evidencia el paquete Modelo el cual está compuesto por los siguientes componentes:

**MondrianHelper.java** la cual es la encargada de obtener los roles y esquemas, para esto hace uso de las

siguientes librerías

- commons-lang-2.4.jar
- pentaho-bi-plataform-plugin-service-4.5.0-stable.jar
- pentaho-bi-plataform-engine-core-4.5.0-stable.jar
- pentaho-bi-plataform-api-4.5-SNAPSHOT.jar
- mondrian-3.4.1.jar
- eigenbase-xom-1.3.0.13768.jar
- pentaho-bi-plataform-engine-service-4.5.0-stable.jar
- pentaho-bi-plataform-plugin-action-4.5.0-stable.jar

**Escribir_XML:** Implementa los métodos para escribir los permisos en el esquema XML usando como ayuda la librería javax.servlet-5.1.12.jar

**Leer_XML:** Implementa los métodos para leer los permisos del XML usando como ayuda la librería javax.servlet-5.1.12.jar.

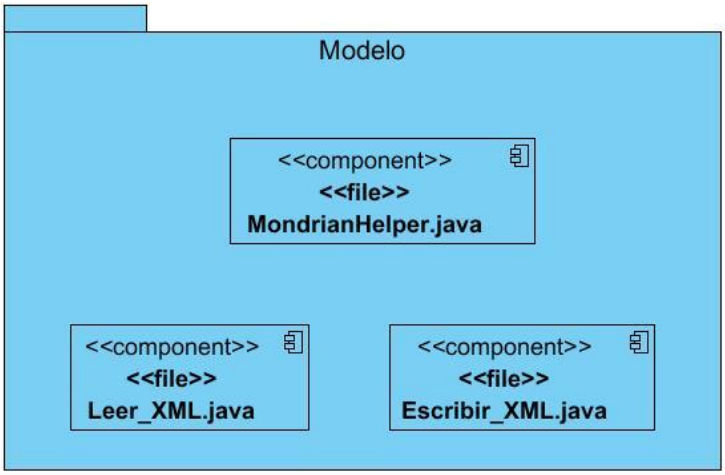


**Figura 14: Componentes del paquete Modelo**

En la figura 10 se muestra el paquete Librería el cual está compuesto por todas las librerías que van a ser usadas por las clases .java para realizar la implementación de la solución.

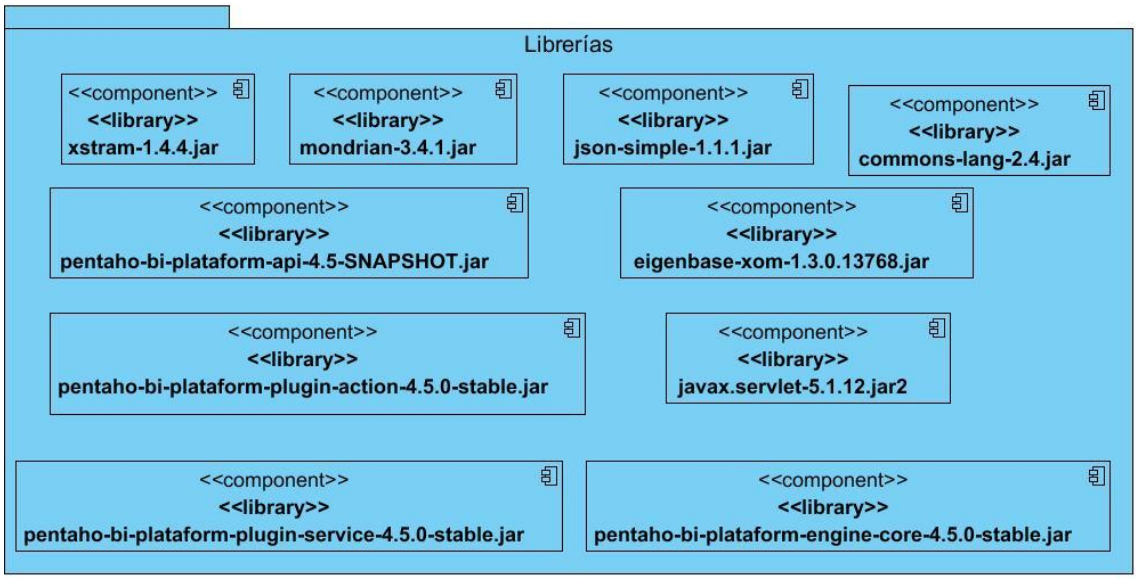


**Figura 15: Componentes del paquete Librería**

### 3.2. Modelo de despliegue

El Modelo de Despliegue define la arquitectura física del sistema por medio de nodos interconectados. Estos nodos son elementos de hardware sobre los cuales pueden ejecutarse los elementos del software. (36)

En la figura 11 se muestra el modelo de despliegue del sistema, el cual está compuesto por 2 nodos que se conectan mediante soportes bidireccionales.


**Figura 16: Diagrama de despliegue**

A continuación se explican las responsabilidades de cada nodo:

- La **PC Cliente** es el nodo encargado de acceder al Servidor Web mediante un navegador, haciendo uso del protocolo HTTP.

- El **Servidor Web** es el Apache Tomcat 6.0 que contiene la aplicación y permite realizar conexiones bidireccionales, unidireccionales, síncronas y asíncronas con el cliente, generando o cediendo una respuesta.

### 3.3. Estándares de codificación

En general, un estándar de codificación es un conjunto de reglas que se siguen para la escritura del código fuente, de tal manera que otros programadores se les facilite entender el código.

En particular el de Java, tiene reglas como: Las clases inician en mayúsculas, los atributos y métodos inician con minúsculas, las constantes son todas en mayúsculas, entre otros. A continuación se mostrarán algunos de los estándares de codificación que se tuvieron en cuenta en la elaboración de la solución.

#### Comentarios

Permiten describir ciertas líneas de código o deshabilitar código no necesario. El estilo de los comentarios debe ser (/* */ o //). Éstos deben ser elaborados con un lenguaje formal siempre teniendo en cuenta la ortografía.

```
//Devolver listado de catálogos

/*
 Obtener la ubicación del fichero que contiene el esquema al que se le
 modificaran los permisos;
 */
```

#### Declaraciones de variables

Las variables son declaradas en una sola línea y el nombre cuando es simple será escrito en minúscula y cuando es compuesto, tendrá la primera letra de las restantes palabras que lo formen con mayúscula.

```
String nombre;
List<Miembro> miembrosHijos;
```

#### Declaraciones de clases

En la declaración de una clase, la llave de apertura comienza al lado de la línea de definición. Además, el nombre de la clase cuando es compuesto o no, tendrá la primera letra de cada palabra que lo forma en mayúscula.

```
public class Cubo {
      .
      .
      .
}
```

```
public class EsquemaPe {
      .
      .
      .
}
```

**Declaración de una función**

En la declaración de una función, la llave de apertura comienza al lado de la línea de definición. Además, el nombre de la función cuando es simple será escrito en mayúscula y cuando es compuesto, tendrá la primera palabra con minúscula y la primera letra de las restantes palabras que lo forman con mayúscula.

```
public Dimensiones[] getDimensiones() {
      .
      .
      .
}
```

## 3.4. Interfaces del Sistema

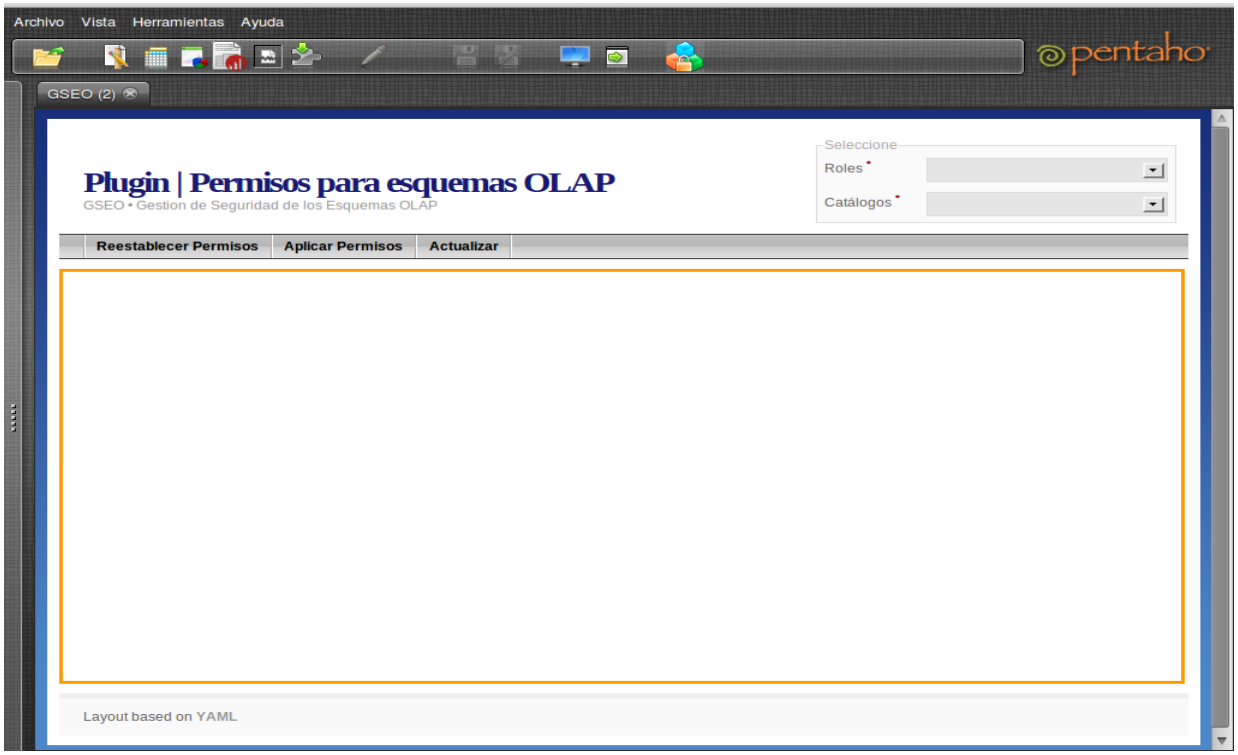


**Figura 17: Interfaz inicial**

Inicialmente el sistema muestra una interfaz vacía como se muestra en la Figura 12. De esta se selecciona un rol y un catálogo, mostrándose los permisos que posee el esquema seleccionado como se evidencia en la Figura 13.

La interfaz posee tres botones encargados de realizar las operaciones en la aplicación donde:

**Restablecer permisos**: Permite obtener nuevamente los permisos que posee el catálogo, en caso de que se cometa algún error y se quieran ver los permisos originales.

**Aplicar permisos:** Es el botón encargado de escribir los nuevos permisos en el esquema XML, si lo que se está realizando es una modificación o eliminación de permisos, muestra un mensaje verificando que desea modificar los definidos anteriormente, una vez que se acepte, se muestra otro mensaje asegurando que se asignaron los permisos correctamente.

**Actualizar:** Es el encargado de reiniciar el sistema, limpiando la caché y cargando los nuevos datos introducidos.

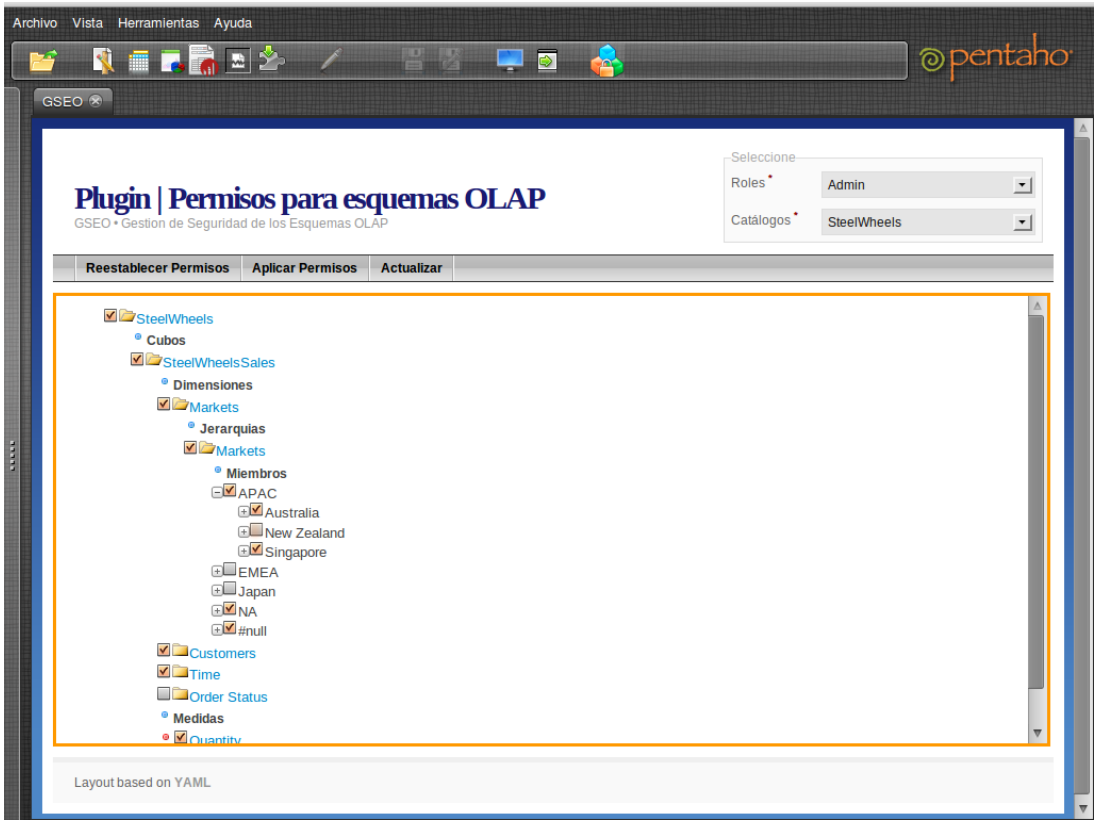


**Figura 18: Interfaz principal (Permisos)**

## 3.5. Pruebas de Software

Las aplicaciones informáticas no están exentas de errores por lo que a todas ellas es necesario aplicarles un conjunto de pruebas para garantizar que cuentan con la calidad requerida. Este es el factor fundamental para lograr entregar un software que cumpla o supere las expectativas del cliente.

Para ello se aplican una serie de técnicas de pruebas del software. Estas técnicas facilitan una guía sistemática para diseñar pruebas que comprueben la lógica interna de los componentes del software y verifiquen los dominios de entrada y salida del programa para descubrir errores en la funcionalidad, el comportamiento y rendimiento. (28).

Para la realización del proceso de pruebas en la solución obtenida se decidió realizar

pruebas unitarias aplicando el método de caja blanca haciendo uso de la técnica camino básico y pruebas del sistema aplicando casos de prueba mediante la técnica partición de equivalencia del método caja negra.

### 3.5.1 Prueba de caja blanca

La prueba de caja blanca, denominada a veces prueba de caja de cristal es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para obtener los casos de prueba. (28)

Mediante los métodos de prueba de caja blanca, el ingeniero del software puede obtener casos de prueba que garanticen que se ejerciten:

- por lo menos una vez todos los caminos independientes de cada módulo.
- todas las decisiones lógicas en sus vertientes verdadera y falsa.
- todos los bucles en sus límites y con sus límites operacionales.
- las estructuras internas de datos para asegurar su validez. (28)

Una de las técnicas de prueba de caja blanca es el camino básico, la cual determina la complejidad ciclomática de una porción de código. Este método permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. Los casos de prueba obtenidos del conjunto

básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa. (28)

Esta complejidad se puede calcular de tres formas:
El número de regiones del grafo de flujo coincide con la complejidad ciclomática.

1. La complejidad ciclomática, V (G), de un grafo de flujo G, se define como: V (G)=A-N+2. Donde A es el número de aristas del grafo de flujo y N es el número de nodos del mismo.
2. La complejidad ciclomática, V (G), de un grafo de flujo G también se define como V (G) = P +1. Donde P es el número de nodos predicados contenidos en el grafo de flujo G. (28)

Para la función " CrearEstructura " se identificaron los bloques de ejecución y se numeraron para identificarlos (Figura 14). Se obtuvieron nueve bloques y se determinó el camino básico ilustrado en la Figura 15.

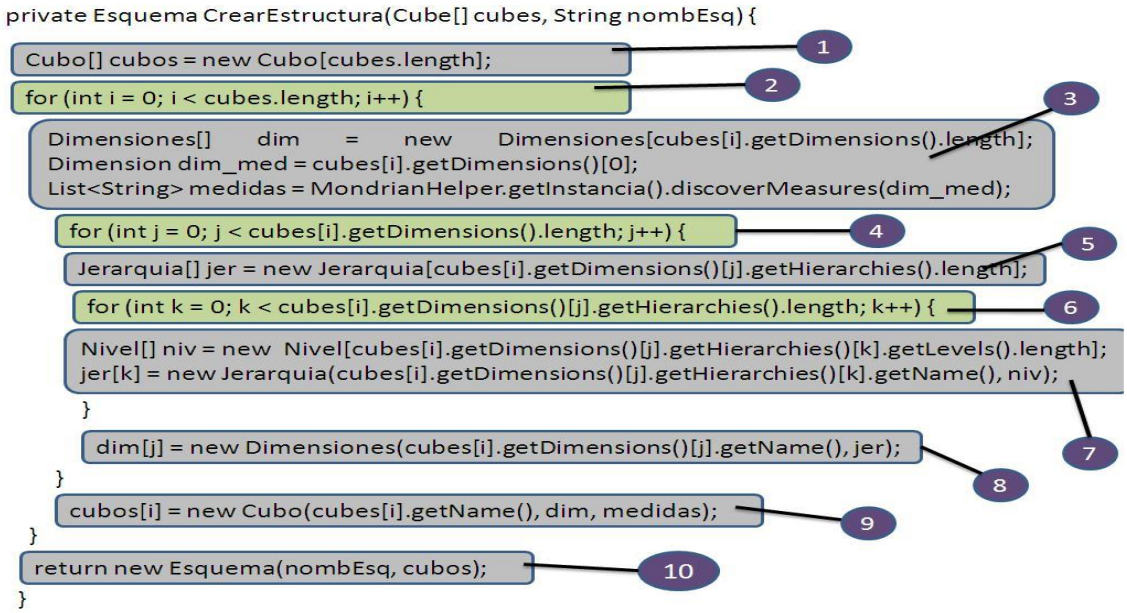


**Figura 19: Función que cargar la estructura del catálogo seleccionado sin incluir los permisos. El código se divide por bloques de ejecución, los cuáles están enumerados y constituyen los nodos del camino básico.**

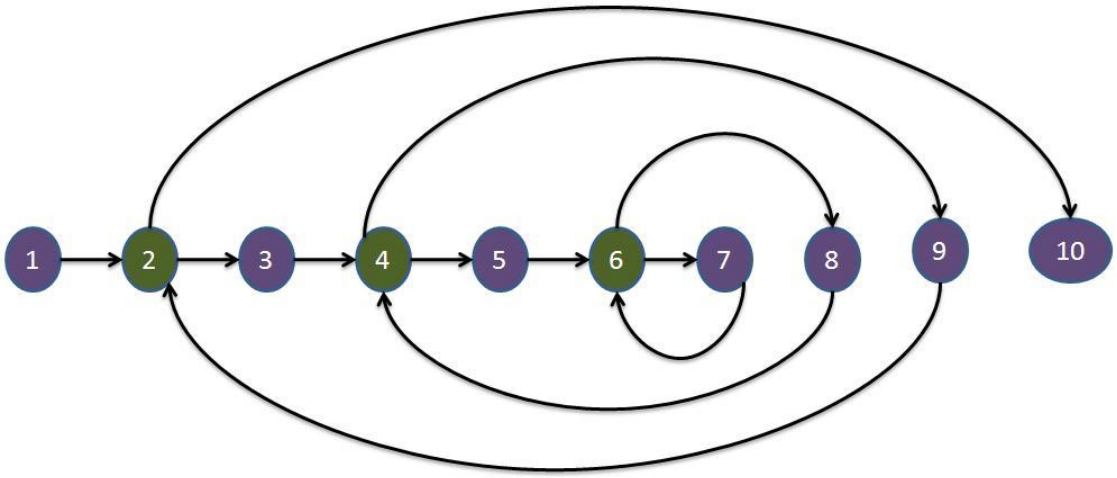


**Figura 20: Camino básico de la función CrearEstructura. Los nodos resaltados en color verde corresponden a las condicionales del código de la función y por tanto son nodos predicados. Las aristas indican los posibles caminos a seguir a partir del nodo correspondiente.**

Con el camino básico determinado, se aplica una de las tres formas para calcular la complejidad ciclomática. Para esto se utilizó la fórmula V (G) = A-N + 2, para la cual se obtuvo doce artistas y diez nodos, por lo tanto: V (G) = 12-10 + 2, quedando V (G) = 4. De la misma forma se pueden comprobar que las otras variantes explicadas para calcular la complejidad ciclomática arriban al mismo valor. Como resultado se obtiene la existencia de 4 caminos independientes y la definición de casos de prueba que garanticen que se ejerciten por lo menos una vez todos los caminos independientes.

### 2.5.2. Prueba de caja negra

Las pruebas de caja negra, también denominada prueba de comportamiento, se centran en los requisitos funcionales del software, o sea, permite al ingeniero del software obtener un conjunto de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. (28)

La prueba de caja negra no es una alternativa a las técnicas de prueba de caja blanca, más bien se trata de un enfoque complementario que intenta descubrir diferentes tipos de errores que los descubiertos por esta. La prueba de caja negra intenta encontrar 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. (28)

Dentro de la prueba de caja negra existen varias técnicas entre las que se pueden mencionar:

- Partición de Equivalencia**:** Divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software.
- Análisis de Valores Límites: Prueba la habilidad del programa para manejar datos que se encuentran en los límites aceptables.
- Grafos de Causa-Efecto: Permite al encargado de la prueba validar complejos conjuntos de acciones y condiciones.

Para verificar que la aplicación se comporte según los requerimientos establecidos por el cliente, se diseña 1 caso de prueba usando la técnica de partición de equivalencia de caja negra.

Partiendo de la descripción de los casos de uso se diseñó un caso de prueba para el CU Gestionar Permisos. A partir de esta, se detallan las variables encontradas en la interfaz que se asocian a este CU definiéndose los siguientes campos:

- **No:** Se enumeran todas las variables descritas en el CU.
- **Nombre del campo:** Se especifica el nombre del campo de entrada.
- **Clasificación:** Se clasifica según el componente de diseño utilizado (ejemplo: campo de texto, lista desplegable, combo box, entre otros).
- **Puede ser nulo:** Se especifica si el campo puede ser nulo o no, mediante un **Sí** o un **No**.
- **Descripción:** Se describen brevemente los datos que deberían introducirse.

El la siguiente tabla se muestra la descripción de las variables para el CU Gestionar permisos.

**Tabla 3: Descripción de variables**

| No | Nombre del campo | Clasificación | Valor nulo | Descripción |
|---|---|---|---|---|
| 1 | roles | Lista de selección | No | Se escoge el rol al que se le va asignar los permisos. |
| 2 | catálogos | Lista de selección | No | Se escoge el esquema al que se le va a crear, modificar o eliminar los permisos. |

Esta descripción posibilitó que se realizara el caso de prueba para el CU mencionado anteriormente, donde se definieron dos escenarios: Gestionar permisos correctamente y Gestionar permisos con campos vacíos, mediante los cuales se evaluó y probó la validez de cada uno de los datos introducidos en el sistema, obteniéndose la respuesta del sistema la cual indica si la prueba es satisfactoria o no. Para mayor información dirigirse al artefacto Diseño de Casos de Prueba donde viene explicado los casos de prueba realizados en el sistema.

**Tabla 4: Caso de prueba para el CU Gestionar permisos**

| **Escenario** | | **Gestionar permisos correctamente** | **Gestionar permisos con campos vacíos** | | |
|---|---|---|---|---|---|
| **Variables** | **roles** | V | I | V | I |
| | **catálogos** | V | V | I | I |
| **Respuesta del sistema** | | El sistema guarda los permisos asignados.<br>El sistema emite un mensaje de que se asignaron los permisos satisfactoriamente | El sistema emite un mensaje de error de que debe seleccionar un rol y un esquema | | |
| **Resultado de la prueba** | | Satisfactorio | Satisfactorio | | |

**Resultados de las pruebas de caja negra**

En la primera iteración se detectaron errores como :

- Error ortográfico en la interfaz principal, ejemplo: Gestion
- Uso excesivo de mayúscula en el texto de la descripción del botón Reestablecer Permisos, Aplicar

Permisos, Actualizar Permisos y en el mensaje Acceso Denegado.

- Error de interfaz. El botoń Cancel de la opción Actualizar Permisos aparecen en inglés, debe ser Cancelar, lo mismo sucede con el botoń Ok del mensaje Acceso denegado, debe ser Aceptar.

En la segunda iteración se solucionaron todas las deficiencias detectadas en la iteración anterior.

**Tabla 5: No conformidades**

| Iteración | No conformidades | Cerrada | No procede |
|---|---|---|---|
| 1ra | 7 | 7 | 0 |
| 2da | 0 | 0 | - |

**Conclusiones del capítulo**

En este capítulo se confeccionó el diagrama de componentes representándose la relación existente entre las clases del diseño descritas. Además, se elaboró el diagrama de despliegue donde se presentó la distribución de los nodos físicos del sistema unidos por conexiones de comunicación y se describió el estándar de codificación que se utilizó para la implementación del sistema.

Se analizaron las principales técnicas y métodos de pruebas de software, tanto de caja blanca como de caja negra. Se determinó aplicar el método del camino básico, correspondiente al método de caja blanca, a la función encargada de cargar la estructura del esquema que seleccione el administrador la cual arrojó un valor de 4 como complejidad ciclomática de la misma. Por otro lado se definió un caso de prueba de caja negra para la técnica de partición de equivalencia al CU: **“**Gestionar permisos**”.**

## Conclusiones Generales

Como resultado de la investigación se desarrolló un plugin para gestionar la seguridad en los cubos OLAP del Pentaho BI Server, dando con esto cumplimiento a los objetivos específicos y tareas trazadas. Se puede concluir con el desarrollo del mismo:

- El análisis de los principales conceptos relacionados con las herramientas encargadas de asegurar la seguridad en los cubos OLAP proporcionó los elementos teóricos necesarios para guiar el proceso de desarrollo. Además el estudio realizado de las metodologías, herramientas y tecnologías sentó las bases para el desarrollo de la solución.
- Los artefactos generados en el flujo de análisis y diseño permitieron implementar de forma satisfactoria la solución.
- La implementación del plugin para la gestión de la seguridad de los cubos OLAP permitió dar cumplimiento a los requisitos funcionales identificados con anterioridad, satisfaciendo las necesidades del cliente.
- Las pruebas realizadas permitieron evaluar los resultados obtenidos y mejorar la calidad del producto final.

## Recomendaciones

Con la culminación del presente trabajo se obtuvo un plugin para gestionar la seguridad de los cubos OLAP en el Pentaho BI Server. Los desarrolladores de dicho plugin recomiendan:

- El uso del plugin en los proyectos del departamento, para garantizar el nivel de seguridad de las soluciones que se desarrollen.
- Mostrar el plugin solo a los administradores del sistema.
- Aumentar la agilidad del plugin enviando solo al servidor los permisos modificados.

## Referencias Bibliográficas


1. **InformaticaHoy, Derechos reservado.** InformaticaHoy. [En línea] 2010. [Citado el: 06 de diciembre de 2012.] http://www.informatica-hoy.com.ar/telefonos-celulares/Cubo-OLAP-una-base-de-datos-multidimensional.php.

2. **Orol, Alfredo Martínez.** GestioPolis. [En línea] Febrero de 2007. [Citado el: 6 de 12 de 2012.] http://www.gestiopolis.com/canales8/ger/olap-online-analytic-processing.htm.

3. **Velazques, Dionicio.** sideshare. [En línea] 14 de Obtubre de 2010. [Citado el: 12 de enero de 2013.] http://www.slideshare.net/dvelasquezt/academia-latinoamericana-de-business-intelligence-albi.

4. **Todo BI, Derechos reservados.** Todo BI<business intelligence>. [En línea] 21 de 5 de 2011. [Citado el: 22 de 11 de 2012.] http://todobi.blogspot.com/2006/05/pentaho-la-solucion-open-source.html.

5. **summan, Derechos Reservados.** Summan. [En línea] 2008. [Citado el: 22 de 11 de 2012.] www.summan.com/pentaho/pentaho-bi-platform-server.

6. **Pentaho, Derechos reservados.** Pentaho Open Source Business Intelligence. [En línea] 2006-2011. [Citado el: 6 de 12 de 2012.] http://pentaho.almacen-datos.com/mondrian.html.

7. **Espinosa, Roberto.** El Ricon del BI. [En línea] 4 de Julio de 2010. [Citado el: 16 de febrero de 2013.] http://churriwifi.wordpress.com/2010/07/04/17-3-preparando-el-analisis-dimensional-definicion-de-cubos-utilizando-schema-workbench/.

8. **Hiperlink, Derechos reservados.** hiperlink0plugin. [En línea] 7 de 12 de 2010. [Citado el: 15 de 11 de 2012.] http://hiperlink0plugin.wordpress.com/page/2/ referencia3.

9. **Méndez, Alejandra Virrueta.** Metodologias de Desarrollo de Software. [En línea] 2010. [Citado el: 7 de diciembre de 2012.] www.monografias.com/.../metodologias-de-desarrollo-software/metodologias-de-desarrollo-software.pdf.

10. **Eumed, Derechos reservados.** Eumed.net. [En línea] [Citado el: 19 de 11 de 2012.] http://www.eumed.net/libros-gratis/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm ref8.

11. **Amaro Calderón, Sarah Dámaris.** Metodologías Ágiles. [En línea] 2008. [Citado el: 21 de 11 de 2012.] www.seccperu.org/files/Metodologias%20Agiles.pdf ref9.

12. **Levano, Daniel.** Articulos. [En línea] 18 de Noviembre de 2008. [Citado el: 6 de diciembre de 2012.] http://www.articulosya.com/article/437/Open_UP_Como_modelo_alternativo_para_el_cumplimiento_del_proceso_de_desarrollo_de_la_NTP-ISO/IEC_122.aspx.

13. Ecured. [En línea] 2012. [Citado el: 2 de 12 de 2012.] http://www.ecured.cu/index.php/Metodologías_de_desarrollo_de_software.

14. Ecured. [En línea] 2012. [Citado el: 4 de 12 de 2012.] http://www.ecured.cu/index.php/UML.

15. Ecured. [En línea] 2012. [Citado el: 4 de 12 de 2012.] http://www.ecured.cu/index.php/CASE.

16. **Miranda, Jocelyn Robledo.** [En línea] 4 de Diciembre de 2012. [Citado el: 8 de enero de 2013.] http://tecnologiajocelyn.bligoo.com/tag/tecnologia.

17. Definicion.De. [En línea] [Citado el: 8 de diciembre de 2012.] http://definicion.de/java/.

18. **Hanze, Heredia.** [En línea] 2010. [Citado el: 06 de diciembre de 2012.] dspace.espoch.edu.ec/bitstream/123456789/96/1/18T00372.pdf.

19. **Lapuente, María Jesús Lamarca.** HTML. [En línea] 2012. [Citado el: 5 de diciembre de 2012.] www.hipertexto.info/documentos/html.htm.

20. w3c world wide web. [En línea] [Citado el: 5 de diciembre de 2012.] www.w3c.es/Divulgacion/GuiasBreves/TecnologiasXML.

21. **Foundation, The jQuery.** jQuery. [En línea] 2013. [Citado el: 20 de febrero de 2013.] http://jquery.com/.

22. [En línea] [Citado el: 15 de mayo de 2013.] http://www.creativasfera.com/15-frameworks-para-desarrollo-web.

23. Ecured. [En línea] 2012. [Citado el: 22 de 11 de 2012.] http://www.ecured.cu/index.php/IDE_de_Programaci%C3%B3n .

24. **Joaquín Seoane Pascual, Jesús M. González Barahona y Gregorio Robles.** Introducción al software Libre. [En línea] 2008. [Citado el: 7 de diciembre de 2012.] http://curso-sobre.berlios.de/introsobre/2.0.1/sobre.html/eclipse.html.

25. **Pozo, Laura Fontanillo Fontanillo y Rubén González del.** [En línea] [Citado el: 26 de noviembre de 2012.] zarza.usal.es/~fgarcia/docencia/poo/04-05/Trabajo/Eclipse.pdf.

26. **Gimenez, Maria Alejandra.** NETBEANS accesible. [En línea] 2012. [Citado el: 21 de noviembre de 2012.] netbeansaccesible.blogspot.com.

27. Tecnologia y Sinergix. [En línea] 10 de 07 de 2008. [Citado el: 15 de 02 de 2013.] http://synergix.wordpress.com/2008/07/10/modelo-de-dominio/.

28. **Pressman, Roger S.** *Ingeniería de Software:Un Enfoque Práctico.* s.l. : Mc Graw Hill, 2001.

29. **Arizaca, Eliza.** [En línea] [Citado el: 9 de 1 de 2013.] virtual.usalesiana.edu.bo/web/practica/archiv/clases_2.doc.

30. **Velasquez, Key.** BuenasTareas. [En línea] Noviembre de 2010. [Citado el: 10 de enero de 2013.] http://www.buenastareas.com/ensayos/Patrones-Arquitectonicos/1069013.html.

31. **Sebastian, Juan.** Seguridad de la información software y tecnología Comusoft. [En línea] 13 de Noviembre de 2010. [Citado el: 16 de Febrero de 2013.] http://www.comusoft.com/modelo-vista-controlador-definicion-y-caracteristicas.

32. **reservados, Derechos.** mi granito de java. [En línea] Mayo de 2011. [Citado el: 16 de febrero de 2013.] http://migranitodejava.com/2011/05/patrones-de-diseno-de-gof.

33. —. CiberAula. [En línea] 2010. [Citado el: 8 de enero de 2013.] http://www.ciberaula.com/articulo/diseno_patrones_j2ee.

34. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* Mexico : Prentice Hall, 1999. 970-17-0261-1.

35. **reservados, Derechos.** msdn. [En línea] 2012. [Citado el: 2 de mayo de 2013.] http://msdn.microsoft.com/es-es/library/dd409390.aspx.

36. **Ivar Jacobson, Grady Booch, James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Addison Wesley, 2000. 0-201-57169-2.

## Bibliografía


1. **InformaticaHoy, Derechos reservado.** InformaticaHoy. [En línea] 2010. [Citado el: 06 de diciembre de 2012.] http://www.informatica-hoy.com.ar/telefonos-celulares/Cubo-OLAP-una-base-de-datos-multidimensional.php.

2. **Orol, Alfredo Martínez.** GestioPolis. [En línea] Febrero de 2007. [Citado el: 6 de 12 de 2012.] http://www.gestiopolis.com/canales8/ger/olap-online-analytic-processing.htm.

3. **Velazques, Dionicio.** sideshare. [En línea] 14 de Obtubre de 2010. [Citado el: 12 de enero de 2013.] http://www.slideshare.net/dvelasquezt/academia-latinoamericana-de-business-intelligence-albi.

4. **Todo BI, Derechos reservados.** Todo BI<business intelligence>. [En línea] 21 de 5 de 2011. [Citado el: 22 de 11 de 2012.] http://todobi.blogspot.com/2006/05/pentaho-la-solucion-open-source.html.

5. **summan, Derechos Reservados.** Summan. [En línea] 2008. [Citado el: 22 de 11 de 2012.] www.summan.com/pentaho/pentaho-bi-platform-server.

6. **Pentaho, Derechos reservados.** Pentaho Open Source Business Intelligence. [En línea] 2006-2011. [Citado el: 6 de 12 de 2012.] http://pentaho.almacen-datos.com/mondrian.html.

7. **Espinosa, Roberto.** El Ricon del BI. [En línea] 4 de Julio de 2010. [Citado el: 16 de febrero de 2013.] http://churriwifi.wordpress.com/2010/07/04/17-3-preparando-el-analisis-dimensional-definicion-de-cubos-utilizando-schema-workbench/.

8. **Hiperlink, Derechos reservados.** hiperlink0plugin. [En línea] 7 de 12 de 2010. [Citado el: 15 de 11 de 2012.] http://hiperlink0plugin.wordpress.com/page/2/ referencia3.

9. **Méndez, Alejandra Virrueta.** Metodologias de Desarrollo de Software. [En línea] 2010. [Citado el: 7 de diciembre de 2012.] www.monografias.com/.../metodologias-de-desarrollo-software/metodologias-de-desarrollo-software.pdf.

10. **Eumed, Derechos reservados.** Eumed.net. [En línea] [Citado el: 19 de 11 de 2012.] http://www.eumed.net/libros-gratis/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm ref8.

11. **Amaro Calderón, Sarah Dámaris.** Metodologías Ágiles. [En línea] 2008. [Citado el: 21 de 11 de 2012.] www.seccperu.org/files/Metodologias%20Agiles.pdf .

12. **Levano, Daniel.** Articulos. [En línea] 18 de Noviembre de 2008. [Citado el: 6 de diciembre de 2012.] http://www.articulosya.com/article/437/Open_UP_Como_modelo_alternativo_para_el_cumplimiento_del_proceso_de_desarrollo_de_la_NTP-ISO/IEC_122.aspx.

13. Ecured. [En línea] 2012. [Citado el: 2 de 12 de 2012.] http://www.ecured.cu/index.php/Metodologías_de_desarrollo_de_software.

14. Ecured. [En línea] 2012. [Citado el: 4 de 12 de 2012.] http://www.ecured.cu/index.php/UML.

15. Ecured. [En línea] 2012. [Citado el: 4 de 12 de 2012.] http://www.ecured.cu/index.php/CASE.

16. **Miranda, Jocelyn Robledo.** [En línea] 4 de Diciembre de 2012. [Citado el: 8 de enero de 2013.] http://tecnologiajocelyn.bligoo.com/tag/tecnologia.

17. Definicion.De. [En línea] [Citado el: 8 de diciembre de 2012.] http://definicion.de/java/.

18. **Hanze, Heredia.** [En línea] 2010. [Citado el: 06 de diciembre de 2012.] dspace.espoch.edu.ec/bitstream/123456789/96/1/18T00372.pdf.

19. **Lapuente, María Jesús Lamarca.** HTML. [En línea] 2012. [Citado el: 5 de diciembre de 2012.] www.hipertexto.info/documentos/html.htm.

20. w3c world wide web. [En línea] [Citado el: 5 de diciembre de 2012.] www.w3c.es/Divulgacion/GuiasBreves/TecnologiasXML.

21. **Foundation, The jQuery.** jQuery. [En línea] 2013. [Citado el: 20 de febrero de 2013.] http://jquery.com/.

22. [En línea] [Citado el: 15 de mayo de 2013.] http://www.creativasfera.com/15-frameworks-para-desarrollo-web.

23. Ecured. [En línea] 2012. [Citado el: 22 de 11 de 2012.] http://www.ecured.cu/index.php/IDE_de_Programaci%C3%B3n .

24. **Joaquín Seoane Pascual, Jesús M. González Barahona y Gregorio Robles.** Introducción al software Libre. [En línea] 2008. [Citado el: 7 de diciembre de 2012.] http://curso-sobre.berlios.de/introsobre/2.0.1/sobre.html/eclipse.html.

25. **Pozo, Laura Fontanillo Fontanillo y Rubén González del.** [En línea] [Citado el: 26 de noviembre de 2012.] zarza.usal.es/~fgarcia/docencia/poo/04-05/Trabajo/Eclipse.pdf.

26. **Gimenez, Maria Alejandra.** NETBEANS accesible. [En línea] 2012. [Citado el: 21 de noviembre de

2012.] netbeansaccesible.blogspot.com.

27. Tecnologia y Sinergix. [En línea] 10 de 07 de 2008. [Citado el: 15 de 02 de 2013.] http://synergix.wordpress.com/2008/07/10/modelo-de-dominio/.

28. **Pressman, Roger S.** *Ingeniería de Software:Un Enfoque Práctico.* s.l. : Mc Graw Hill, 2001.

29. **arizaca, Eliza.** [En línea] [Citado el: 9 de 1 de 2013.] virtual.usalesiana.edu.bo/web/practica/archiv/clases_2.doc.

30. **Velasquez, Key.** BuenasTareas. [En línea] Noviembre de 2010. [Citado el: 10 de enero de 2013.] http://www.buenastareas.com/ensayos/Patrones-Arquitectonicos/1069013.html.

31. **Sebastian, Juan.** Seguridad de la información software y tecnología Comusoft. [En línea] 13 de Noviembre de 2010. [Citado el: 16 de Febrero de 2013.] http://www.comusoft.com/modelo-vista-controlador-definicion-y-caracteristicas.

32. **reservados, Derechos.** mi granito de java. [En línea] Mayo de 2011. [Citado el: 16 de febrero de 2013.] http://migranitodejava.com/2011/05/patrones-de-diseno-de-gof.

33. —. CiberAula. [En línea] 2010. [Citado el: 8 de enero de 2013.] http://www.ciberaula.com/articulo/diseno_patrones_j2ee.

34. **Larman, Craig.** *UML y Patrones. Introducción al análisis y diseño orientado a objetos.* Mexico : Prentice Hall, 1999. 970-17-0261-1.

35. **Reservados, Derechos.** msdn. [En línea] 2012. [Citado el: 2 de mayo de 2013.] http://msdn.microsoft.com/es-es/library/dd409390.aspx.

36. **Ivar Jacobson, Grady Booch, James Rumbaugh.** *El Proceso Unificado de Desarrollo de Software.* Madrid : Addison Wesley, 2000. 0-201-57169-2.

37. Internet2.0. [En línea] [Citado el: 17 de 11 de 2012.] www.psicologia2000.com/.../32909-complemento-informatica.html .

37. Google Driver . [En línea] [Citado el: 15 de 11 de 2012.] https://docs.google.com/document/d/1...eTNylgncCCO9i8g/edit.

38. **Jimenez, Eliseo Castro.** slideshare. [En línea] 28 de 11 de 2008. [Citado el: 21 de 11 de 2012.] http://www.slideshare.net/ecastrojimenez/uml-lenguaje-de-modelamiento-unificado-presentation

39. Inei. [En línea] [Citado el: 23 de noviembre de 2012.] www.inei.gob.pe/biblioineipub/bancopub/Inf/Lib5103/Libro.pdf .

40. Redinertho. [En línea] 22 de junio de 2012. [Citado el: 26 de noviembre de 2012.] http://redinertho.wordpress.com/2012/06/22/ventajas-en-la-utilizacion-de-eclipse/.

41. Universidad de Oriente. [En línea] 13 de julio de 2012. [Citado el: 4 de diciembre de 2012.] http://wiki.monagas.udo.edu.ve/index.php/Metodolog%C3%ADas_SCRUM_y_XP.

42. **Solarte, Mario.** [En línea] 15 de Junio de 2010. [Citado el: 17 de febrero de 2013.] http://www.slideshare.net/Enf2IST/introduccin-a-los-patrones-de-diseo .

43. **Torrijos, Ricard Lou.** Programacion en castellano. [En línea] [Citado el: 18 de febrero de 2013.] http://www.programacion.com/articulo/catalogo_de_patrones_de_diseno_j2ee__i__capa_de_presentacion_240/4.

Anexos

**Tabla 6: Descripción CU Restablecer permisos**

| | | |
|---|---|---|
| **Objetivo** | El caso de uso permite al usuario obtener todos los permisos que posee el esquema seleccionado hasta nivel de registro. | |
| **Actores** | Administrador | |
| **Resumen** | El caso de uso inicia cuando el usuario selecciona la opción Restablecer permisos (esto ocurre cuando el usuario luego de hacer alguna modificación en los permisos quiere ver los permisos originales que tiene asignado el esquema). El caso de uso termina luego de realizar esta acción. | |
| **Complejidad** | Media | |
| **Prioridad** | Crítico | |
| **Precondiciones** | El usuario debe estar autenticado como administrador en el sistema. | |
| **Poscondiciones** | | |
| **Flujo de eventos** | | |
| **Flujo básico: Gestionar permisos** | | |
| | **Actor** | **Sistema** |
| 1. | a. El usuario selecciona el botón Restablecer permisos | b. El sistema muestra el catálogo seleccionado con los permisos existentes, finalizándose el CU. |
| **Flujos alternos** | | |

| 1.a. Existen campos vacíos | | |
|---|---|---|
| | **Actor** | **Sistema** |
| 1. | | Se emite un mensaje para que seleccione un rol y un esquema de la lista de roles y catálogos existentes y regresa a la acción 1.a. |

**Tabla 7: Descripción CU Actualizar sistema**

| | |
|---|---|
| **Objetivo** | El caso de uso permite que el sistema luego de guardados los cambios y siempre que el usuario lo desea, refresque el repositorio del Pentaho BI Server y vacíe la caché de los esquemas de Mondrian permitiendo actualizar todos los cambios en la plataforma. |
| **Actores** | Administrador |
| **Resumen** | El caso de uso inicia cuando se selecciona el botón Aplicar Cambios. El caso de uso finaliza cuando el sistema reinicia la caché con la nueva información guardada. |
| **Complejidad** | Media |
| **Prioridad** | Crítico |
| **Precondiciones** | El usuario debe estar autenticado como administrador en el sistema. |
| **Postcondiciones** | |
| **Flujo de eventos** | |

| Flujo básico Refrescar caché | | |
|---|---|---|
| | **Actor** | **Sistema** |
| 1. | a. El usuario selecciona el botón Aplicar Cambios | b. El sistema refresca el repositorio del Pentaho BI Server.<br>c. El sistema vacía la caché de los esquemas de Mondrian.<br>d. El sistema emite un mensaje informando que ya se ha refrescado el repositorio y vaciado la caché de los esquemas de Mondrian, terminándose así el CU. |

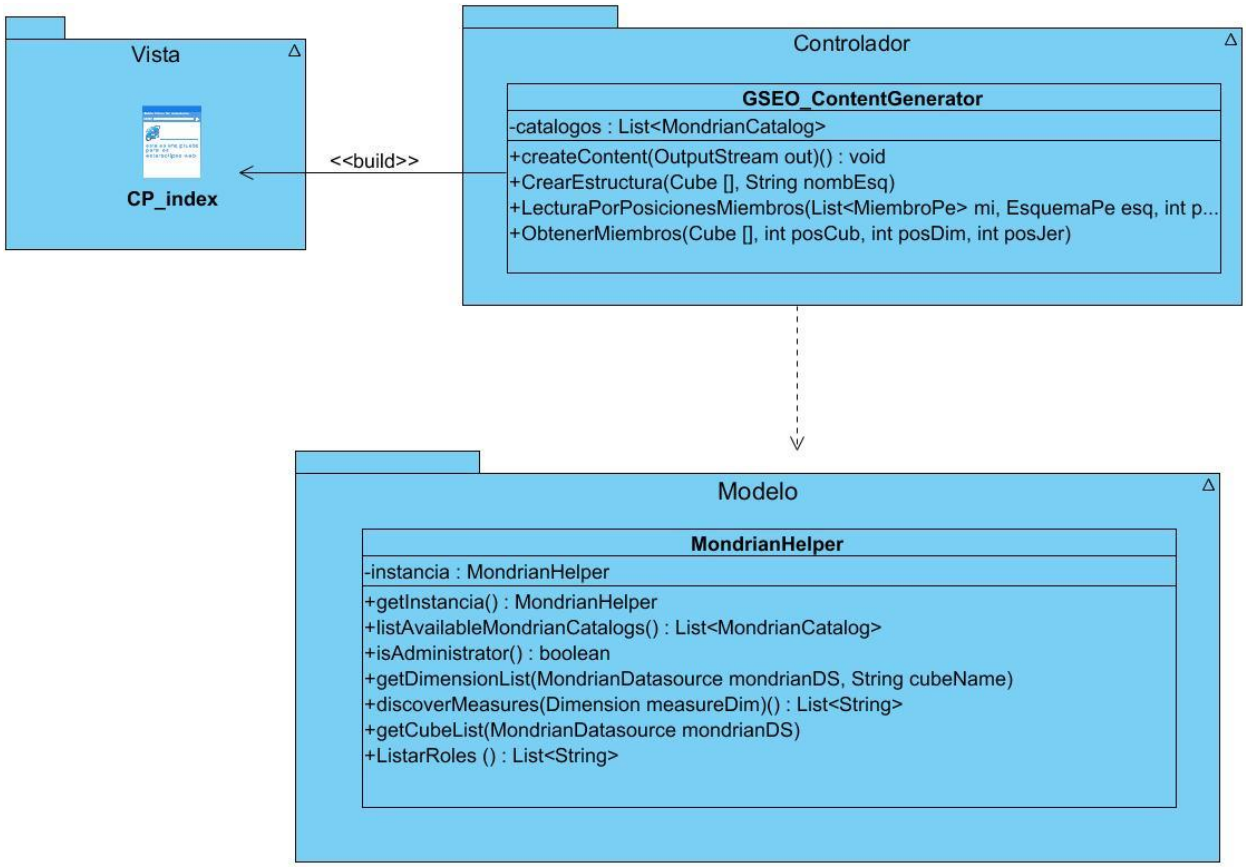


**Figura 21: Diagrama de clases del diseño CU Actualizar sistema**

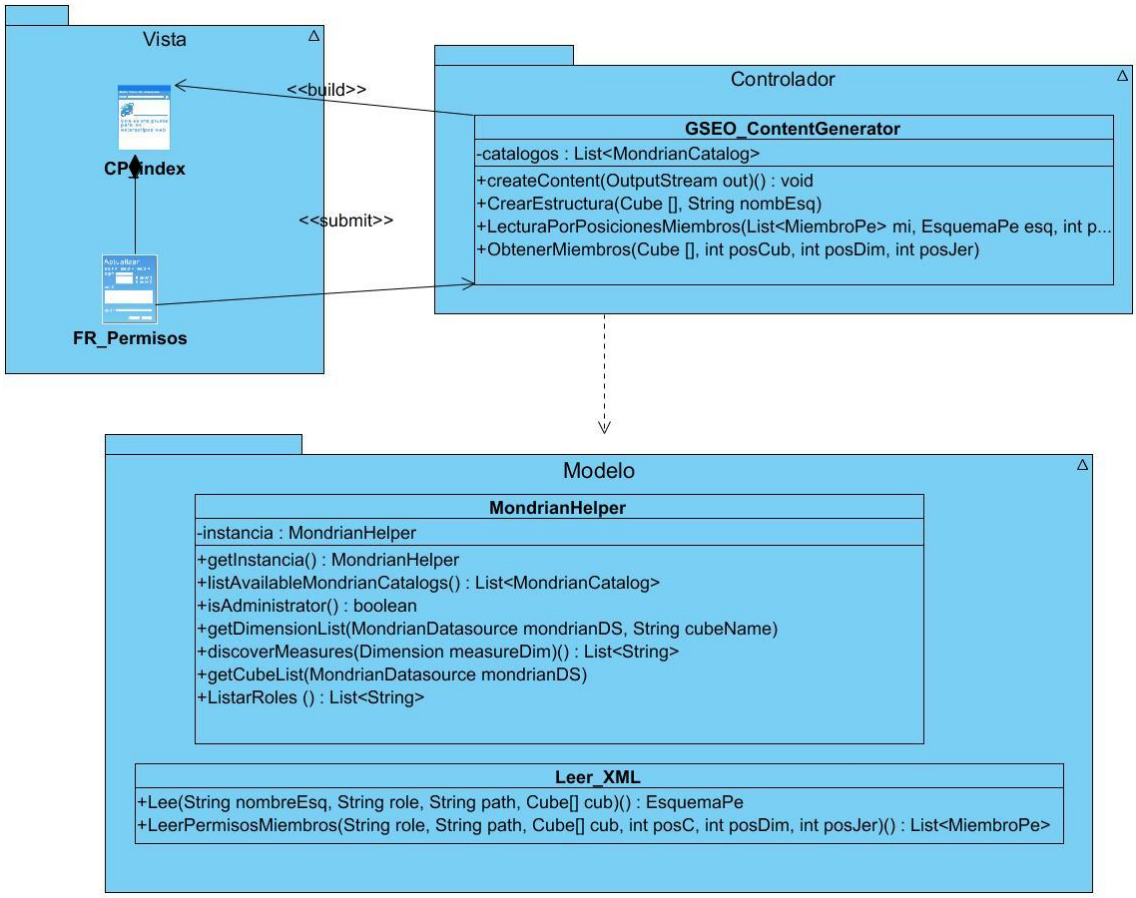


**Figura 22: Diagrama de clases del diseño CU Restablecer permisos**

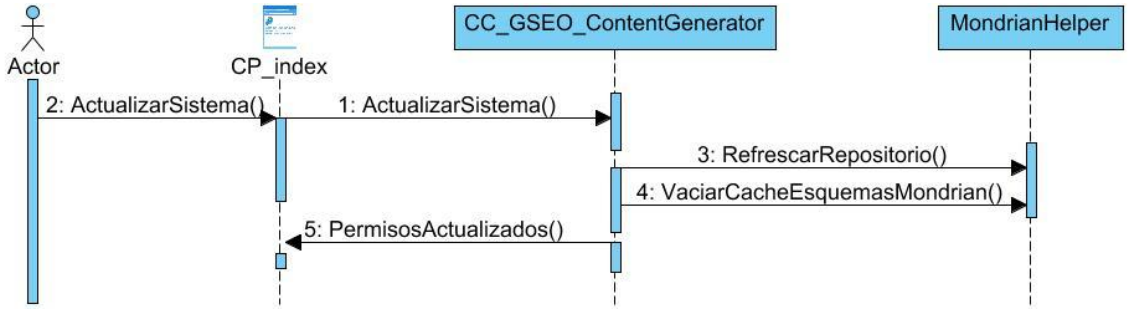


**Figura 23: Diagrama de secuencia CU Actualizar sistema**

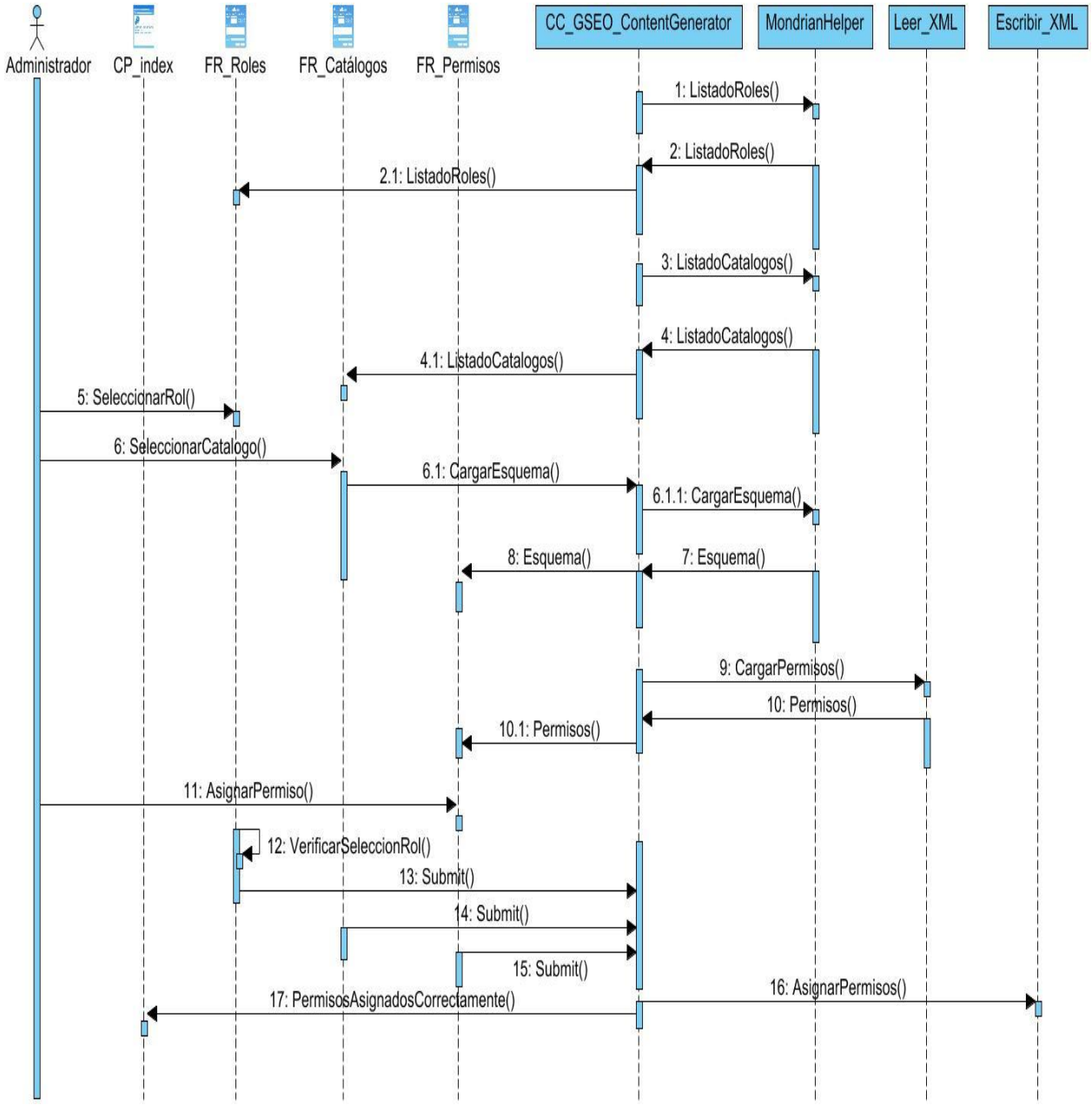


**Figura 24: Diagrama de secuencia CU Gestionar permisos**

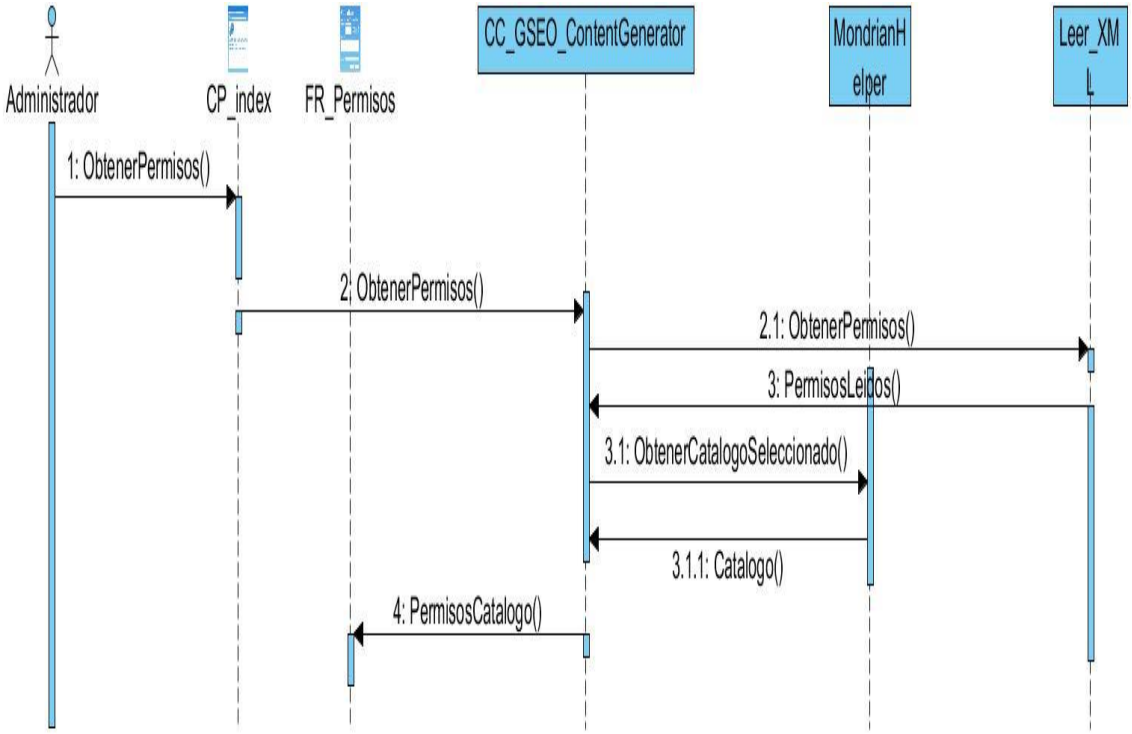


**Figura 25: Diagrama de secuencia CU Restablecer permisos**