Universidad de las Ciencias Informáticas
Facultad 6

Título: Herramienta para importar los reportes desde la versión 1.7 del Generador Dinámico de Reportes a su versión 2.0

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autora: Claudia García Suárez del Villar.

Tutores: Ing. Yanet Rosales Morales.

Ing. Aurelio Rodríguez Duran.

La Habana Junio 2012

"Año 54 de la Revolución"

DECLARACIÓN DE AUTORÍA

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

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

Claudia García Suárez del Villar.

_____

Firma del Autor

Aurelio Rodríguez Duran.

_____

Firma del Tutor

Yanet Rosales Morales.

_____

Firma de la Tutora

DATOS DE CONTACTO

Autora:
Claudia García Suárez del Villar.
Universidad de las Ciencias Informáticas
e-mail: cgarcias@estudiantes.uci.cu.

Tutor:
Aurelio Rodríguez Duran.
Ingeniero de las Ciencias Informáticas.
e-mail: arduran@uci.cu

Tutor:
Yanet Rosales Morales.
Ingeniera de las Ciencias Informáticas.
e-mail: yrmorales@estudiantes.uci.cu.

AGRADECIMIENTOS

Un sueño se hace realidad cuando se lucha día a día por él, cuando se pone todo el empeño y dedicación para alcanzarlo. Hoy se hizo realidad mi sueño, después de muchos años de sacrificio he alcanzado la meta que una vez me propuse y por eso quiero agradecer a todas aquellas personas que de una forma u otra han estado apoyándome en todo momento : profesores de ayer y hoy, amigos de ayer y hoy, estando siempre en las buenas y malas junto a mí, animándome a seguir adelante ante cada nuevo reto que la vida me ha presentado, a todos los que un día me dijeron cuenta conmigo, muchísimas gracias pero especialmente a :

A la mujer más buena y más comprensiva del mundo, mi mamá, *por todo lo que has hecho en la vida por mí, por su amor, confianza, dedicación, apoyo, por ser guía ejemplar y constante, por ser una madre excepcional, sin ser perfecta me has guiado por el buen camino. Te quiero mucho.*

Al que todo lo sabe, mi papá, *por guiar cada uno de mis pasos, por brindarme siempre su amor a cambio de nada, sus enseñanzas, sus años de sacrificio, su confianza, sus consejos y apoyo en cada momento, por ser el papá más lindo y especial de este mundo, por confiar tanto en mí y estar seguro que no lo defraudaría. Te quiero mucho.*

A mi familia *por esperar siempre lo mejor de mí y porque cada uno de ellos puso un granito de arena para que pudiera llegar a ser quien soy, en especial a mi abuela Teresa, mi hermana Elita, a mi tía Eloísa, mi tío Rafael, mi tía Nélida, mi primita Laura, mi primo Ramsés y mi abuelos Leonor y Troadio que aunque ya no están siempre me enseñaron muy buenos valores y la importancia del estudio.*

Mi novio Alexis, *por estar a mi lado en todos los momentos duros de la carrera y la tesis, por darme fuerzas cuando creía que todo estaba perdido, por regañarme cuando lloraba innecesariamente, por cambiar mis lágrimas por sonrisas, por darme ese infinito amor que alegra cada día de mi vida, por todos los momentos lindos que hemos vivido juntos y los que quedan por vivir, por enseñarme todas las cosas bellas que tiene el mundo, por ser la persona más especial que he conocido y por querernos tanto.*

Mis amigas, *por los buenos momentos que pasamos juntos y los recuerdos que quedaron grabados para siempre a* ***Ana Rosa*** *pues a pesar de que a veces pelea mucho, es una de las personas más especiales que he conocido y en estos cinco años ha sido una muy buena amiga, a* ***Marla*** *que aunque siempre está criticando, lo hace con el mayor cariño del mundo, para que seamos mejores personas, y muchas veces es nuestro paño de lágrimas y nuestra consejera, a* ***Margarita*** *y* ***Nelly*** *con las que hemos vivido miles de locuras y por eso ya forman parte del piquete más unido de la UCI.*

Este es un agradecimiento especial a la persona que más se parece a mí en su forma de pensar y actuar, a mi fiel compañera de estudios, de fiesta, de grandes locuras y gozaderas, con la que siempre he compartido todo durante estos 5 años, la que me ha soportado y ha estado ahí para mi, juntas hemos hecho maldades, hemos dado chucho, puesto nombretes, hemos estudiado, siempre preocupada por mí, aconsejándome, bridándome su apoyo, a mi buena amiga que ya en estos momentos es casi una hermana ***Juliet Mora Casares*** *mi gran agradecimiento por ser mi gran amiga de la universidad.*

Mis queridos tutores Yanet Rosales y Aurelio Rodríguez, *por ser cómplices de este gran esfuerzo, por el apoyo brindado, por ofrecerme sus conocimientos y ayudarme a estar hoy aquí, por sus sacrificios en algún tiempo incomprendido, por sus ejemplos de superación incasable, por su comprensión y confianza, por su amor y amistad incondicional, porque sin su apoyo no hubiera sido posible la culminación de mi carrera profesional y sobre todas las cosas por su esmerada dedicación.*

A todas aquellas personas *que no dudaron ni un minuto en dejar lo que estuviesen haciendo para prestarme su ayuda y su atención para aclararme una duda relacionada con la tesis y en ese sentido merece un agradecimiento especial mi amigo y compañero Julio Cesar Brito por ser tan especial, nunca hubo un no, cuando necesité de él, siempre con la mejor disposición de brindarme su ayuda a cambio de nada, también agradecer a Marcos, Yoandy, Michael, Reynaldo Nieblas, el profe Alberto Garnache a Lianet Cruz y Lianet Salazar a todos muchas gracias por todo su apoyo.*

A todos mis compañeros que estuvieron conmigo durante estos 5 años de carrera en los que pasamos buenos y malos momentos pero siempre juntos y contentos, en especial quiero mencionar a: Yonnis,

AGRADECIMIENTOS

Victor, Yohanca, Luis Miguel, Alejandro Navarro por ser tan un amigo muy especial, Alexis, las mimis, Mayelin, Mayeleiny y Ayeris, la gente del piquete…., a mis compañeras de apartamento por ser mi pequeña familia aquí en la universidad, a todos muchas gracias por soportarme y por cuidar siempre de mí. A mis amistades de la infancia en especial: Mónica, Raif Ernesto, Odelaysis, Lisandra y Lillanys. A todos los profesores que ayudaron en mi formación tanto profesional como en mi vida personal.

En general, agradezco a todos aquellos que granito de arena para que un día como hoy, pudiera dedicarle estas palabras. A todos ustedes, Gracias, Muchas Gracias.

DEDICATORIA

Entre mis grandes aspiraciones, mis metas y mis sueños siempre estuvo presente ser una graduada de nivel superior. En estos momentos aún no puedo creer que ya lo haya logrado, pero ya es un hecho, ya soy toda una ingeniera. Justo ahora me vienen muchas personas a la mente y es cuando me doy cuenta de que este gran logro va dedicado a:

En primer lugar a la persona que desde niña siempre me revisaba las tareas, me forraba las libretas y me ayudaba a estudiar las cosas para ir preparada a la próxima clase. Esa persona que fue la primera en inculcarme el amor por los estudios, los buenos valores y la importancia de tener un título. Me enseñó a ayudar siempre a los demás en los estudios, ella me decía que de esa forma yo también me preparaba mejor, me tuvo paciencia cuando me ponía majadera y siempre buscó la forma de de que el estudio no me pareciera una obligación, sino que lo disfrutara al máximo. Aunque no esté presente, ella siempre está conmigo, a mi primera y mejor maestra, mi gran abuela Leonor Placeres Calderón va dedicada esta tesis, gracias por lo que soy hoy.

En segundo lugar dedico esta tesis a mis padres Betsaida Suárez del Villar Placeres y Oscar García Bicet, pues sin su apoyo incondicional nada de esto fuese posible, por enseñarme todo lo que sé y prepararme para la vida en todos los sentidos, por decirme sigue adelante cuando algo no me salía como yo esperaba, por sus buenos consejos, por darme lo que tenían y lo que no tenían con tal de que yo no me detuviese nunca, por estar ahí siempre para mí. Por ser los padres más maravillosos del mundo este también es su logro.

En tercer lugar dedico esta tesis a todos los miembros de mi familia que siempre me han apoyado y me han dado ánimos para seguir adelante con mis sueños. Realmente tengo una familia increíble y pienso que al menos un pedacito de esta felicidad es gracias a ellos.

RESUMEN

Los generadores de reportes son herramientas complementarias de los sistemas de información que utilizan un lenguaje transparente al usuario, capaces de realizar consultas a la base de datos para obtener información de ella en forma de reportes. El departamento de Soluciones Integrales perteneciente al Centro de Tecnologías de Gestión de Datos en la Universidad de las Ciencias Informáticas, cuenta actualmente con el Generador Dinámico de Reportes en su versión 1.7. La herramienta permite generar reportes de forma dinámica partiendo de datos persistentes en algún origen de datos soportado por el sistema. Sin embargo, carece de funcionalidades, pues la tecnología que utiliza como motor de reporte PhpReport se encuentra en desventaja con respecto a las características y componentes que presentan otras de su tipo. Para lograr reducir estos inconvenientes se decidió desarrollar la versión 2.0, la cual contará con JasperReport como motor de reportes. Los reportes generados por estas herramientas son incompatibles, pues ambos motores tienen características diferentes. El objetivo del presente trabajo de diploma es desarrollar una herramienta que permita migrar los reportes de la versión 1.7 del Generador Dinámico de Reportes basado en PhpReport a su versión 2.0 que utilizará JasperReport como motor de reportes. Como resultado se desea obtener la integración de este componente de importación al Generador Dinámico de Reportes en su nueva versión, logrando transformar los reportes generados por el motor PhpReport de la versión 1.7 al JasperReport del Generador Dinámico de Reportes 2.0.

Palabras clave: generadores de reportes, migración, origen de datos.

ÍNDICE DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

En la actualidad existen empresas que cuentan con una gran cantidad de información, la cual si no es almacenada de manera cuidadosa, puede perderse o deteriorarse en momentos que sea necesaria su utilidad para un óptimo desempeño de la empresa. En la mayoría de los casos estas organizaciones requieren de una alternativa eficiente que les proporcione información útil en tiempo real, por esta razón es recomendable mantener los datos en forma de reportes. Los reportes son de gran utilidad para conservar, mostrar los análisis y resultados de las empresas, por lo que se consideran un importante requisito en el negocio.

Los generadores de reportes son herramientas de apoyo a los sistemas de información, que permiten a los usuarios acceder de forma simple y rápida a los datos o bases de datos guardados en forma de reportes. Algunas de las herramientas más usadas para realizar estas funciones son: Active Reports, iReport y Crystal Reports, pues ofrecen información en un formato particular y realizan varias operaciones a partir de los datos que se encuentran almacenados en una base de datos. Además permiten realizar consultas a las bases de datos obteniendo así la información en forma de reportes con un mejor dominio en el diseño y la visualización de los datos obtenidos.

Cuba no se encuentra ajena a los avances tecnológicos actuales, a pesar de no contar con un elevado desarrollo en el campo de la informática. El mismo pretende alcanzar una mejor evolución en la industria productora de software y convertirla en una fuente de ingresos para la economía del país. Bajo este principio surge la Universidad de las Ciencias Informáticas (UCI) la cual cuenta con una amplia infraestructura productiva. Dispone de varios centros de desarrollo, entre los que se encuentra el Centro de Tecnologías de Gestión de Datos (DATEC). Entre las líneas de desarrollo que lo componen se cuenta con Soluciones Integrales, cuya función principal consiste en el desarrollo de las herramientas necesarias para crear soluciones integrales de Análisis de Datos.

El Generador Dinámico de Reportes (GDR) en su versión 1.7 es uno de los sistemas desarrollados en dicho departamento, tiene como propósito garantizar la generación dinámica de reportes partiendo de datos persistentes en algún origen de datos soportado por el sistema (PostgreSQL, MySQL Server, SQLlite, Microsoft SQL Server, Oracle). Está compuesto por un conjunto de módulos que brindan funcionalidades para el trabajo con los reportes brindándoles soporte durante todo el ciclo de vida. El sistema aún carece de capacidades que son de importancia para su correcto funcionamiento. La tecnología que utiliza para la generación de reportes PhpReport se encuentra en desventaja con otras de su tipo, resultando muy complicado hacerle modificaciones para nuevas prestaciones.

Para lograr reducir estos inconvenientes surge la necesidad de desarrollar una nueva versión (2.0) del GDR, presentando como principal característica que sea lo suficientemente genérica como para ser utilizada en cualquier empresa que requiera los servicios de un sistema de este tipo. También debe contar con las premisas de confiabilidad, seguridad, eficiencia, las potencialidades necesarias para la obtención y creación de reportes que faciliten la toma de decisiones convirtiéndolo en un producto de excelencia. Esta nueva versión del GDR decide utilizar JasperReport como motor de reportes, el cual ofrece nuevas potencialidades para generar de reportes. Los motores de reportes de ambas versiones presentan características diferentes, por lo que son incompatibles los reportes que son generados por cada uno de ellos.

A raíz de la problemática planteada, surge el siguiente **Problema de la investigación:** ¿Cómo importar los reportes de la versión 1.7 del Sistema Generador Dinámico de Reportes a la versión 2.0? Este problema se define con el **Objeto de estudio:** Proceso de desarrollo de software de los sistemas generadores de reportes, el cual está delimitado por el **Campo de acción:** Herramienta para importar reportes en el Generador Dinámico de Reportes.
Para resolver el problema planteado surge como **Objetivo general:** Desarrollar una herramienta que permita migrar los reportes de la versión 1.7 del Generador Dinámico de Reportes a su versión 2.0.

Del **Objetivo general** se desglosan los siguientes **Objetivos Específicos:**

1) Realizar un análisis del marco conceptual relacionado con los motores de reportes utilizados en las versiones 1.7 y 2.0 del Sistema Generador Dinámico de Reportes.

2) Realizar el análisis y diseño de la herramienta para la migración de reportes desde la versión 1.7 del Generador Dinámico de Reportes a su versión 2.0.

3) Realizar la implementación y prueba de la herramienta para la migración de reportes desde la versión 1.7 del Generador Dinámico de Reportes a su versión 2.0.

Tareas de la investigación:

1. Análisis de los motores de reportes PhpReport y JasperReport.

2. Selección de las herramientas y metodología a utilizar.

3. Definición de los requisitos funcionales y no funcionales de la solución.

4. Análisis de la herramienta para la importación de reportes.
5. Diseño de la herramienta para la importación de reportes.
6. Implementación de los componentes de diseño.
7. Ejecución de las pruebas.
8. Documentación y corrección de las no conformidades identificadas en la ejecución de las pruebas.

Posibles resultados:

- Herramienta que permita importar los reportes de la versión 1.7 del Generador Dinámico de Reportes basado en PhpReport a su versión 2.0 que utilizará JasperReport como motor de reportes.

Estructuración del trabajo de diploma:

- **Capítulo 1: Fundamento Teórico.**

Este capítulo contiene los fundamentos teóricos para entender el problema a solucionar y conceptos fundamentales. Un análisis de las tecnologías, las herramientas, metodologías y tecnologías posibles a utilizar para desarrollar el sistema y la definición de cuáles serán las seleccionadas para lograr tal propósito.

- **Capítulo 2: Análisis y Diseño.**

Se presenta el desarrollo de la extensión a través de la discusión de los artefactos resultados de las disciplinas Análisis, Diseño, e Implementación requeridos para la elaboración y construcción de la misma.

- **Capítulo 3: Implementación y pruebas de la solución.**

Comienza con el resultado del diseño, implementando el sistema en términos de componentes, así como una revisión final de las especificaciones del diseño y de la codificación mediante la realización de pruebas, garantizando la calidad del software.

CAPÍTULO 1: FUNDAMENTO TEÓRICO

1. Introducción

En este capítulo se describen los principales conceptos relacionados con las herramientas de generación de reportes JasperReport y PhpReport las cuales son de vital importancia para darle solución al problema científico planteado anteriormente. Se abordarán sus principales características y se realizará un análisis crítico de las mismas a través de la comparación tomando en cuenta los aspectos más relevantes. Además se establecen las herramientas, metodología y lenguajes a utilizar argumentando el por qué de su elección.

1.1 Motores de Reportes

Los sistemas de generación de reportes están generalmente compuestos por dos elementos básicos: un diseñador de reportes y un motor de reportes. Este último es el encargado de construir el informe con los datos a partir de una plantilla previamente elaborada desde el diseñador de reportes. Entre sus capacidades se encuentra la obtención de reportes en diferentes formatos de salida. A continuación se analizan dos motores de reportes reconocidos en el ámbito nacional e internacional.

1.1.1 PhpReport

PhpReport es un motor de generación de reportes para PHP, es software libre y provee un alto nivel de productividad en este tipo de aplicaciones. Un reporte en PhpReport está dividido en capas que contienen unas a otras. De esta forma un reporte sería un documento, contiene un encabezado, un pie y está formado por páginas, las que a su vez cuentan con un encabezado, un pie de página y puede contener grupos anidados en los que se cargarán los campos obtenidos de la consulta a la base de datos. (1)

La biblioteca base de reportes PhpReport es la librería que utiliza actualmente la versión 1.7 del GDR para la generación de reportes en PHP. La asimilación y adaptación de esta biblioteca comienza desde los inicios del proyecto Paquete de Ayuda para la Toma de Decisiones (PATDSI) desarrollado en el Centro de Tecnologías de Gestión de Datos, de la Universidad de las Ciencias Informáticas en el cual los desarrolladores necesitaban una biblioteca de generación de reportes que fuera libre para adaptarla al negocio, que se pudieran agregar nuevas funcionalidades y que contribuyera, además, a la soberanía tecnológica.

PhpReport ha sido modificado por este equipo de trabajo agregándole nuevas funcionalidades, para mejorar las potencialidades de generación de reportes en los proyectos del centro.

Funcionalidades agregadas al PhpReport

- Nuevos formatos de salida excel, pdf.
- Modificaciones a la salida de HTML, permitiendo mayor personalización del usuario.
- Mejora al soporte de orígenes de datos en *sqlite*.
- Paginación de reportes.
- Agregaciones de funciones de cálculo.

Desarrolladores que ofrecen soporte a PhpReport

Es desarrollado por el Centro de Tecnologías de Gestión de Datos, de la Universidad de las Ciencias Informáticas; que tiene como misiones fundamentales:

- Proveer soluciones integrales, soporte y consultorías relacionadas con tecnologías de bases de datos y análisis de información.
- Desarrollar nuevas tecnologías de bases de datos, de procesamiento y representación de la información a partir del desarrollo de proyectos de I+D [1]con un claro enfoque a la soberanía tecnológica.
- Contribuir con la formación profesional.

Algunas de las ventajas que provee PhpReport son las siguientes:

- Poder contar con la biblioteca PhpReport es de gran importancia pues como la misma es libre se le pueden hacer cambios al código fuente según el negocio de cada proyecto de generación de reportes.
- Provee una serie de funcionalidades nuevas que son de gran importancia y aportan a la generación de reportes dinámicamente enfocadas a los proyectos del centro.

[1] I+D: Sus siglas significan investigación y desarrollo, principal énfasis en la investigación en ciencias aplicadas o en ciencias básicas, o bien el desarrollo de la ingeniería que persigue con la unión de ambas, un incremento en la innovación que conlleva a un aumento de las ventas de las empresas (29).

Sistema Operativo: Puede ser utilizado en cualquier entorno o sistema pues es multiplataforma.

Licencia: Se distribuye a nivel mundial bajo los términos de la Licencia Pública para Librerías GNU (*GNU Library Public License*), por lo que es software libre.

1.1.2 JasperReport

Es el motor de reportes más usado actualmente en el mundo del Código Abierto *(Opensource)*, puede ser embebido en todo tipo de aplicaciones, desde las que generan reportes a partir de una plantilla o modelo predeterminado hasta las que brindan más libertad al usuario para diseñar sus propios reportes y ejecutar otras operaciones complejas. Es una librería implementada completamente en Java brindando el máximo nivel de portabilidad, con un amplio y expandible grupo de posibles fuentes de datos. Posee además una abundante variedad de formatos de salida, importación así como una gran comunidad mundial que mantiene y desarrolla la librería. Genera informes para formatos de impresión predeterminados existentes o para reportes continuos a ser visualizados en la web, los reportes pueden ser exportados a formatos como: PDF, XML, HTML, CSV, XLS, RTF, TXT. A través de los sub-reportes es posible manipular y diseñar reportes con un diseño altamente complejo. Soporta a la misma vez varios orígenes de datos incluso aunque sean de tipos distintos. (2)

Es posible pasar parámetros desde una aplicación a JasperReport, esto es muy simple de implementar y brinda una herramienta muy poderosa que permite clasificar, restringir o mejorar los datos que son enviados al usuario basándose en las condiciones de tiempo de ejecución.

La utilización de esta herramienta permite mejorar la gestión de la empresa mediante la creación y gestión de informes. Con una solución de este tipo, se dispone en el tiempo deseado de los datos indispensables para la gestión eficaz de un departamento. También pueden seguirse fácilmente los resultados obtenidos y la manera en que la actividad progresa en función de los objetivos fijados. Pueden descubrirse las tendencias y los factores claves que afectan a la actividad y a los resultados de la empresa.

Algunas de las ventajas que provee JasperReport son las siguientes:

- Permite una diagramación flexible de los reportes: Los reportes se pueden dividir en secciones opcionales que son: título del reporte, el encabezado de página, una sección para los detalles del reporte, el pie de página y una sección de resumen que aparece al final del reporte.

- JasperReport permite la creación de reportes dentro de reportes (sub-reportes) facilitando su diseño.

- Los reportes son capaces de presentar los datos de manera textual o a través de gráficos, además de mostrar, generar o calcular otros datos de forma dinámica y mostrarlos.

- Permite generar textos o imágenes de fondo para utilizarlo como marcas de agua con el propósito de identificar el reporte o simplemente por motivos de seguridad.

Sistema Operativo: Puede ser utilizado en cualquier entorno o sistema operativo siempre que exista una implementación de la máquina virtual de Java para dicho entorno, las únicas dependencias que deben satisfacerse son: JDK[2] *[Java Development Kit]* 1.3 o superior y JDBC[3] 2.0 en caso que se utilicen fuentes de datos relacionales.

Licencia: Se distribuye a nivel mundial bajo los términos de la Licencia Pública para Librerías GNU *(GNU Library Public License)*, por lo que es software libre y está respaldado por una gran comunidad internacional de desarrollo, el proyecto JasperForge.org y la empresa JasperSoft Corporation.

1.1.3 Comparación entre ambas herramientas de generación de reportes

A continuación se establece una breve comparación entre los motores de reporte estudiados. Para ello se tuvieron en cuenta algunos aspectos que se consideran importantes para seleccionar el motor de reportes más completo e idóneo para ser usado.

Criterios	PhpReport	JasperReport
Arquitectura	Librería	Librería
Licencia	Libre(GPL)	Libre(GPL)
Plataforma	Multiplataforma	Multiplataforma
Fuente de datos	Postgresql, mysql, oracle, mssql, odbc, interbase, informix.	Postgresql, mysql, oracle, mssql, odbc, interbase, informix, xml, csv.
Formato de salida	Html, pdf, excel, csv.	Html, pdf, excels, csv, odt, rtf.
Propiedades de	Paginación, Formato, Hipervínculos	Paginación, Formato, Hipervínculos,

[2] JDK *[Java Development Kit]*: Software que provee herramientas de desarrollo para la creación de programas en Java (30).

[3] JDBC: Biblioteca de clases que permite la conexión con la base de datos usando Java (31).

reporte		Sub-reportes.
Equipo de Soporte	Equipo de GDR.	Gran comunidad mundial.

Tabla 1: Comparación de las Herramientas de generación de reportes.

Ambas opciones cumplen con el principio de soberanía tecnológica que sigue la UCI y no requieren costo de adquisición. Para la selección del motor también se tuvo en cuenta que JasperReport ofrece muchos más componentes que PhpReport como tablas de contingencia dinámica, etiquetas, grid, dial, 2D, 3D entre otros. Tiene un mayor rendimiento pues cuenta con muchas características que no tiene PhpReport como las *Watermarks* ó Marcas de agua, esto se utiliza en los informes generados para los temas de seguridad, pues con esta funcionalidad todos los informes tienen la misma marca de agua, mantienen un aspecto coherente y se hace más difícil de falsificar informes. Otra particularidad del motor seleccionado es que permite el manejo de informes de gran tamaño pues almacena los segmentos de un informe sobre el disco con el fin de liberar algo de memoria. Presenta informes de localización, pues aprovechándose de las características de internacionalización del lenguaje Java es capaz de generar informes en varios idiomas. Cuenta con variables de reportes que permiten asignar las expresiones de un reporte a una variable, eliminado la necesidad de escribir la expresión en múltiples ocasiones. Teniendo en cuenta la superioridad de JasperReport sobre PhpReport en cuanto a fuentes de datos soportadas, formatos de salida, propiedades de reportes y las características mencionadas con anterioridad se determina utilizarlo en la versión 2.0 de GDR. (14)

1.2 Lenguajes de Programación

Un lenguaje de programación es aquel elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; permitiendo al desarrollador comunicarse con los dispositivos de hardware y software existentes. A continuación se describen algunos de estos lenguajes utilizados en el desarrollo de la aplicación. (16)

1.2.1 Lenguaje de Programación del lado del servidor PHP5

PHP es un lenguaje de programación interpretado que significa Procesador Hipertexto *[Hypertext Pre-processor]* (PHP), diseñado originalmente para la creación de páginas web dinámicas. La primera versión de PHP fue creada por Rasmus Lerdorf en 1994. Es usado principalmente en interpretación del lado del servidor *[server-side scripting]* para la generación de páginas Web dinámicas. No es un lenguaje de marcas como podría ser *[HyperText Markup Language]* (HTML), XML o lenguaje de marcas sin hilos *[Wireless Markup Language]* (WML*)*. Está más cercano a JavaScript o a C. Cuenta

con un tipo de datos dinámicos, es multiplataforma y ofrece un soporte para bases de datos MySQL, PostgreSQL, Oracle, MS SQL Server, Sybase mSQL, Informix, entre otras. (3)

Esta tecnología dispone de varias versiones, la más actual que es la usada actualmente por el GDR es PHP5. El 13 de julio de 2004, fue lanzado PHP5, utilizando el motor Zend Engine 2.0. La versión más reciente de PHP es la 5.3.6 que fue lanzada el 23 de agosto de 2011.

Las principales características de PHP5 son: su rapidez, facilidad de aprendizaje; su soporte multiplataforma tanto de diversos sistemas operativos, como servidores HTTP y de bases de datos. Este lenguaje es distribuido de forma gratuita bajo una licencia abierta.

Ventajas:

- El código fuente escrito en PHP5 es invisible tanto para el navegador como para el cliente porque es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP5 sea segura y confiable.
- Permite aplicar técnicas de programación orientada a objetos.
- Biblioteca nativa de funciones bastante amplia e incluida.
- Es completamente expandible. Está compuesto de un sistema principal, un conjunto de módulos y una variedad de extensiones de código.
- Posee muchas interfaces distintas para cada tipo de servidor. Este lenguaje actualmente se puede ejecutar bajo Apache, IIS, AOLServer, Roxen y THTTPD[4].

Teniendo en cuenta la versión anterior del GDR se decide utilizar para la implementación del sistema a desarrollar el lenguaje de programación del lado del servidor PHP5.

1.2.2 Lenguaje de Programación del lado del cliente JavaScript

Al igual que Visual Basic y Perl, JavaScript es un lenguaje interpretado, característica que lo hace especialmente idóneo para trabajar en la Web. Permite el envío dinámico de documentos a través de la web, dejando de ser simples fuentes de información estática. JavaScript comparte muchos elementos con otros lenguajes de alto nivel.

[4] THTTPD: Es un servidor web de código libre disponible para la mayoría de las variantes de Unix.

Es un lenguaje de tipo script compacto, basado en objetos y guiado por eventos, diseñado específicamente para el desarrollo de aplicaciones cliente-servidor. Los programas JavaScript van incrustados en los documentos XHMTL o en un fichero js y se encargan de realizar acciones en el cliente.

Ventajas:

- Los programas escritos en este lenguaje no requieren de mucha memoria ni tiempo adicional de transmisión, por ser pequeños y compactos.
- No requiere un tiempo de compilación, pues los scripts se pueden desarrollar en un período de tiempo relativamente corto.
- Es independiente de la plataforma hardware o sistema operativo este funciona correctamente siempre y cuando exista un navegador que lo soporte.
- Asegura la permanencia de una operación realizada y aunque falle el sistema esta no podrá deshacerse. (4)

Teniendo en cuenta la versión anterior del GDR se decide utilizar para la implementación del sistema a desarrollar el lenguaje de programación del lado del cliente JavaScript.

1.3 Metodología de Desarrollo del Software

Una metodología de desarrollo de software es un marco de trabajo que muestra las tareas que se requieren para construir un software de alta calidad, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben tener. Guía a los desarrolladores en la realización de las actividades, durante todo el proceso de creación de un producto. Las metodologías se pueden clasificar en robustas que son usadas en grandes proyectos y las ágiles se utilizan para proyectos de corto plazo de menor tamaño. En las metodologías ágiles se intenta ser lo más flexible posible, que el cliente pueda cambiar los requisitos cuando quiera y que el código funcione bien, por esta razón se utilizará la metodología ágil en el proceso de desarrollo del sistema. (15)

1.3.1 Proceso Unificado Abierto (OpenUP)

La metodología de Proceso Unificado Abierto (OpenUP) presenta las mismas características de Proceso Unificado de Rational (RUP). Contiene el desarrollo iterativo, casos de uso y escenarios de

conducción de desarrollo, gestión de riesgos y el enfoque centrado en la arquitectura. Es un proceso mínimo y suficiente porque solo incluye el contenido necesario y fundamental. Cuenta con los componentes básicos que pueden servir de base a procesos específicos y la mayoría de los elementos están declarados para fomentar el intercambio de información entre los equipos de desarrollo y mantener un entendimiento compartido del proyecto, sus objetivos, alcance y avances. Fue liberado por el Marco de proceso Eclipse *[Eclipse Process Framework]* (EPF), se desarrolló gracias a una donación realizada por la IBM del Proceso Básico Unificado *[Basic Unified Process]*. Fue entregada a Eclipse a fines de 2005 y renombrado como OpenUp en 2006. OpenUP estructura el ciclo de vida de un proyecto en cuatro fases: concepción, elaboración, construcción y transición. El ciclo de vida del proyecto brinda a los interesados un mecanismo de supervisión y dirección para controlar los fundamentos del proyecto, su ámbito, la exposición a los riesgos, el aumento de valor y otros aspectos. (5)

OpenUp presenta las siguientes características:

- Proceso de desarrollo del software. Es completo en el sentido que puede ser manifestado como todo el proceso para construir un sistema.
- Es extensible pues en el proceso se puede agregar o adaptar según la necesidad de los sistemas. OpenUp es un proceso ágil.
- Es ligero y proporciona una comprensión detallada del proyecto, beneficiando a clientes y desarrolladores sobre productos a entregar y su formalidad.
- Se centra en una arquitectura temprana para reducir al mínimo los riesgos y organizar el desarrollo.
- Es la metodología utilizada por desarrolladores de alto nivel en casi todo el mundo por sus altas cualidades administrativas.

Partiendo de que OpenUP permite dirigir cualquier tipo de proyecto, el tiempo de desarrollo es corto, el equipo de trabajo es pequeño y cuenta con menos de veinte artefactos, bajo la decisión del proyecto se decide utilizar como metodología para guiar el proceso de desarrollo de software del sistema a desarrollar.

1.4 Herramientas y Tecnologías

Las herramientas y tecnologías de información que apoyan al desarrollo de una aplicación Web, desempeñan un papel fundamental en su evolución. Constituyen el soporte en la administración del proyecto, adecuándose a las características y objetivos propios de la organización. Con el fin de realizar un software con la calidad requerida se realizó un análisis de las tecnologías y herramientas necesarias para el desarrollo de la aplicación.

1.4.1 Marco de trabajos de desarrollo del lado del cliente ExtJS

El marco de trabajo ExtJS fue creado inicialmente por Jack Slocum. Empezó siendo un conjunto de librerías y extensiones para Intefaz de usuario Yahoo! *[Yahoo! User Inteface]* (YUI), librería desarrollada en JavaScript que explota las potencialidades del JavaScript asincrónico y Xml [*Asynchronous JavaScript And XML]* (AJAX) para el desarrollo de Aplicaciones Ricas de Internet *[Rich Internet Applications]* (RIA). El mismo está escrito en JavaScript con la finalidad de asistir el desarrollo de aplicaciones enriquecidas para Internet. Con el tiempo se convirtió en un Marco de trabajo independiente y a principios de 2007 se creó una compañía para comercializar y dar soporte del Marco de trabajo Ext. (6)

Por esta razón ExtJS tiene un modelo de licencia dual. Para aplicaciones web comerciales hay que adquirir una licencia y para aplicaciones web opensource, que sean compatibles con la licencia GPL de GNU, la cual es gratuita. ExtJS es uno de los Marcos de trabajo que, además de flexibilizar el manejo de componentes de la página como el Modelo en Objetos para la Representación de Documentos *[Document Object Model]* (DOM), peticiones AJAX, tiene la gran funcionalidad de crear interfaces de usuario bastante funcionales. Es una librería JavaScript para la creación de aplicaciones enriquecidas del lado del cliente.

Principales características de ExtJS:

- Alto rendimiento en ejecución debido a la optimización de código Java Script.
- Controles de usuario personalizables.
- Modelo orientado a componentes, bien diseñado y extensible.
- Posee una Interfaz de Programación de *[AplicacionesApplication Programming] Interface* (API) intuitiva y fácil de utilizar.
- Es distribuido bajo licencias Opensource y comerciales.

La versión 3.0 fue liberada el 3 de junio de 2009 entre sus mejoras incluye:

- Soporte para peticiones directas, CRUD y REST.
- Nuevos ejemplos y componentes (incluye un componente para gráficas).
- Más de 1000 mejoras y correcciones.
- Administración de memoria mejorada para Internet Explorer 6.
- API documentada.
- Compatibilidad con versiones anteriores.

Luego del estudio de las características del marco de trabajo, y siguiendo la estructura propuesta por el equipo de desarrollo de GDR, se decide utilizar como marco de trabajo de desarrollo del lado del cliente ExtJS.

1.4.2 Marco de trabajos de desarrollo del lado del servidor Symfony

Symfony es uno de los marcos de trabajo PHP más populares entre los usuarios y las empresas, pues permite que los programadores sean mucho más productivos a la vez que crean código de más calidad y más fácil de mantener. Symfony es maduro, estable, profesional y está muy bien documentado. (7)

Symfony fue diseñado para ajustarse a los siguientes requisitos:

- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos. Su capa de abstracción y el uso de Propel, permiten cambiar con facilidad de Sistema Gestor de Base de Datos (SGBD) en cualquier fase del proyecto.
- Doctrine es uno de los ORM que tiene Symfony que facilita la elaboración de consultas muy complejas y mejora el rendimiento.
- Utiliza programación orientada a objetos, de ahí que sea imprescindible PHP 5.
- Sencillo de usar en la mayoría de casos, aunque es preferible para el desarrollo de grandes aplicaciones Web que para pequeños proyectos.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Fácil de extender, lo que permite su integración con las bibliotecas de otros fabricantes.

- Una potente línea de comandos que facilitan generación de código, lo cual contribuye a ahorrar tiempo de trabajo.

Luego del estudio de las características del marco de trabajo, y siguiendo la arquitectura propuesta por el equipo de desarrollo de GDR, se seleccionó como marco de trabajo a Symfony en su versión 1.4 del lado del servidor

1.4.3 Herramienta Case: Visual Paradigm

La herramienta de modelado Visual Paradigm soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Utiliza como software de modelado el Lenguaje Unificado de Modelado (UML) que ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. También es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el Grupo de Gestión de Objetos *[Object Management Group]* (OMG). Además Visual Paradigm es una potente herramienta CASE la cual permite dibujar todos los tipos de diagramas de clases y generar código desde diagramas. El mismo es multiplataforma, por lo que brinda facilidades para el diseño de los diagramas necesarios, su documentación. (8)

Algunas características Visual Paradigm:

- Generación de bases de datos - Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Proporciona a los desarrolladores una plataforma que les permite diseñar un producto rápidamente y con la calidad requerida.
- Facilita la interoperabilidad con otras herramientas CASE y se integra con múltiples herramientas de desarrollo, como Eclipse/IBMWebSphere, Jbuilder, NetBeans IDE, Oracle Jdeveloper.
- Genera código y realiza ingeniería inversa para diez lenguajes de programación, Java, C++, PHP y XML Schema. Código a modelo, código a diagrama.
- Se integra con el Visio para importar imágenes del mismo para realizar los diagramas de despliegue.
- Genera documentación para el proyecto en HTML, MS Word y PDF.

- Licencia: gratuita y Comercial.
- Fácil de instalar y actualizar.

Partiendo de la arquitectura definida por el proyecto GDR y el estudio realizado, el cual arrojó características importantes que presenta la herramienta se decide utilizar la herramienta CASE en su versión 6.4.

1.4.4 Lenguaje de marcas XML

El lenguaje de marcas XML representa el conjunto de reglas para definir etiquetas semánticas que organizan un documento en diferentes partes. XML es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados. En teoría HTML es un subconjunto de XML especializado en presentación de documentos para la Web, mientras que XML es un subconjunto del Estándar de Lenguaje de Marcado Generalizado *[Standard Generalized Markup Language]* (SGML) especializado en la gestión de información para la Web. En la práctica, XML contiene a HTML, aunque no en su totalidad. XML fue desarrollado por un grupo de trabajo bajo los auspicios del consorcio red informática mundial *[World Wide Web]*, organismo que vela por el desarrollo de WWW partiendo de las amplias especificaciones de SGML[5]. Permite al usuario definir sus propios lenguajes de anotación adaptados a sus necesidades y contiene tres características muy importantes que son: extensibilidad, estructura y validación. Este fue constituido en 1994 con el objetivo de desarrollar protocolos comunes para la evolución de Internet. (9)

Características que ofrece XML:

- Permite proporcionar diferentes vistas sobre los datos (HTML, PDF, voz, etc.), dependiendo de quién sea el cliente.
- Facilita la integración desde fuentes de datos heterogéneas, por ejemplo, páginas Web, distintas bases de datos.
- Los documentos tienen una estructura que los hace legibles e inteligibles, no solo para los ordenadores, sino, también para los humanos.
- Las aplicaciones de XML son fácilmente extensibles mediante Definiciones de nuevos Tipos de

[5] SGML: Sistema para organización y etiquetado de documentos (32).

Documentos (DTD[6]).

- La herramienta XML proporciona una representación estructural de los datos que ha probado ser ampliamente implementable y fácil de distribuir.
- El poder y la belleza del XML es que mantiene la separación entre la interface de usuario y los datos estructurados.
- Los datos codificados en XML pueden ser transmitidos sobre la Web hasta el escritorio.

Ventajas de XML:

- Las aplicaciones se pueden generar rápidamente y su mantenimiento es sencillo.
- Separa los datos de la presentación y del proceso. Esto permite mostrar y procesar los datos al gusto deseado con sólo aplicar distintas hojas de estilo ó aplicaciones.
- La información es más accesible y reutilizable, por la flexibilidad de las etiquetas de XML que permiten su utilización sin tener que adaptarse a reglas específicas de un fabricante.
- Ofrece un formato para la descripción de datos estructurados, facilitando declaraciones de contenido más precisas y resultados de búsquedas más significativos en varias plataformas.

Este lenguaje es utilizado para ofrecer los formatos de salida, de ambas versiones del GDR, por lo que es de gran utilidad en el desarrollo de la aplicación.

1.4.5 Gestor de Base de Datos PostgreSQL

PostgreSQL es un Sistema Gestor de Base de Datos (SGBD) basado en el proyecto POSTGRES, de la universidad de Berkeley. Es una derivación libre de este proyecto y utiliza el lenguaje SQL. Fue uno de los primeros en tratar los conceptos del sistema objeto-relacional actual. Este proyecto lleva más de una década de desarrollo, hoy día es el sistema libre más avanzado, soportando la gran mayoría de las transacciones SQL y control concurrente. (10)

Entre sus principales ventajas se destacan las siguientes:

[6] DTD: Descripción de estructura y sintaxis de un documento XML (33).

- Permite la gestión de diferentes usuarios, como también los permisos asignados a cada uno de ellos.
- Posee una gran escalabilidad, haciéndolo idóneo para su uso en sitios Web que atienden un gran número de solicitudes.
- Puede ser instalado un número ilimitado de veces sin temor de sobrepasar la licencia.
- Posee estabilidad y confiabilidad legendarias.
- Es extensible a través del código fuente disponible sin costos adicionales.
- Es multiplataforma, disponible en la mayoría de los sistemas operativos.
- Permite implementar reglas, vistas, disparadores, sub-consultas y funciones.

Partiendo de la arquitectura que propone el proyecto GDR para su nueva versión y el estudio de las ventajas antes mencionadas se decide utilizar como gestor de base de datos PostgreSQL en su versión 8.4.

1.4.6 PgAdmin III

PgAdmin III es una aplicación gráfica para gestionar el gestor de bases de datos PostgreSQL, siendo la más completa y popular con licencia OpenSource. Está escrita en C++ usando la librería gráfica multiplataforma wxWidgets, [7]lo que permite que se pueda usar en Linux, FreeBSD, Solaris, Mac OS X y Windows. (11)

PgAdmin III está diseñado para responder a las necesidades de todos los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. El interfaz gráfico soporta todas las características de PostgreSQL y facilita enormemente la administración. La aplicación también incluye un editor SQL con resaltado de sintaxis, un editor de código de la parte del servidor, un agente para lanzar scripts programados. La conexión al servidor puede hacerse mediante conexión TCP/IP. La versión actual de PgAdmin III es la 1.8.2.

Por todas las facilidades que ofrece esta herramienta, fue seleccionada por el proyecto GDR para el desarrollo de sus aplicaciones y es la elegida para el desarrollo de la aplicación.

[7]wxWidgets: Las **wxWidgets** son unas bibliotecas multiplataforma y libres, para el desarrollo de interfaces gráficas programadas en lenguaje C++. (11)

1.4.7 Entorno de desarrollo NetBeans7.0

La plataforma de desarrollo NetBeans provee de una estructura para los proyectos que se puede crear junto a este entorno de desarrollo integrado (IDE), a partir de un conjunto de componentes de software llamados módulos. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que estos pueden ser desarrollados de forma independiente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software. El IDE de NetBeans es gratuito, y de código abierto para desarrolladores de software. El mismo tiene gran éxito con una gran base de usuarios y una comunidad en constante crecimiento. Sun MicroSystems fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos. Provee una estructura para los proyectos, propone un esqueleto para organizar código fuente, el editor conjuntamente integra los lenguajes como HTML, JavaScript y CSS. (12)

1.4.8 Lenguaje de Transformación XSLT

Extensible Stylesheet Language Transformations (XSLT) es un lenguaje de transformación que busca patrones dentro de un documento e indica cómo reestructurarlos. Este no sólo permite hacer transformaciones entre documentos XML, sino que también permite generar cualquier tipo de documento desde un documento XML (HTML, texto plano, programa en algún lenguaje de programación como Java o C++). En particular, hoy es usado por buscadores tales como FireFox y Explorer para poder desplegar documentos XML. Está diseñado para su uso como parte de XSL, que es un lenguaje de hojas de estilo para XML. Además de XSLT, XSL incluye un vocabulario XML para especificar el formato. De esta forma es como XSL específica el estilo de un documento XML mediante XSLT para describir cómo el documento se transforma en otro documento XML que utiliza el vocabulario de formato. (13)

Es importante conocer que XSLT también está diseñado para ser utilizado independientemente de XSL. Sin embargo, XSLT no pretende ser un lenguaje de transformación por completo de propósito general XML; sino que está diseñado principalmente para los tipos de transformaciones que se necesitan cuando XSLT se utiliza como parte de XSL.

1.5 Conclusiones Parciales del Capítulo

En el presente capítulo se realizó un análisis de los elementos fundamentales de las diferentes herramientas propuestas, mostrando como resultado las características claves que servirán de base

para el diseño de la propuesta de trabajo. Se profundizó en el estudio de las características de los motores de reportes PhpReport utilizado en la versión 1.7 de GDR y JasperReport que es el motor a utilizar en GDR 2.0. Se decidió utilizar como metodología de desarrollo a OpenUp, como herramienta de modelado Visual Paradigm. Para el desarrollo de la aplicación se utilizan como lenguajes de programación del lado del servidor PHP en su versión 5 y JavaScript del lado del cliente. Como marcos de trabajo para el desarrollo de aplicaciones Web se seleccionaron a ExtJS del lado del cliente y Symfony, que implementa el estilo arquitectónico Modelo - Vista - Controlador. Este marco de trabajo además, tiene integrado el ORM Propel, el cual garantiza una auditoría de seguridad en el código. El gestor de base de datos elegido es PostgreSQL y como administrador de PostgreSQL se utiliza PGAdminIII. En aras de lograr la organización del código se seleccionó el entorno integrado de desarrollo Netbeans 7.0 con soporte para los lenguajes de programación a utilizar, cumpliendo con el estándar de codificación definido por DATEC.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DEL SISTEMA.

1. Introducción

En este capítulo se enfatiza en la propuesta de solución para el problema de la investigación. Con la ayuda de definiciones y conceptos, se explicará el modelo de dominio para representar la estrucutura que se desea representar en el sistema a desarrollar. Luego de profundizar los conceptos fundamentales que describen el dominio de la aplicación, se establecen los requisitos funcionales y no funcionales. Además se diseñarán los diagramas de clases y de interacción para los casos de uso arquitectónicamente significativos, así como también se definirá con precisión la estructura física que presentará el sistema de importación de reportes mediante el modelo de despliegue.

2.1 Modelo Dominio.

El modelo dominio es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Se representa en UML con un diagrama de clases en el que se muestra conceptos u objetos del dominio del problema, asociaciones entre las clases conceptuales y atributos de estas. Una clase conceptual puede ser una idea o un objeto físico (símbolo, definición y extensión). En la fase de inicio se determinó que los procesos del negocio no están claramente definidos, por esta razón se decide representar los conceptos que definen la situación real del sistema mediante un Modelo de Dominio (ver Figura 1). Facilitando a través de un vocabulario común que ayude a usuarios, clientes, desarrolladores e interesados a comprender el argumento principal del sistema. (22)

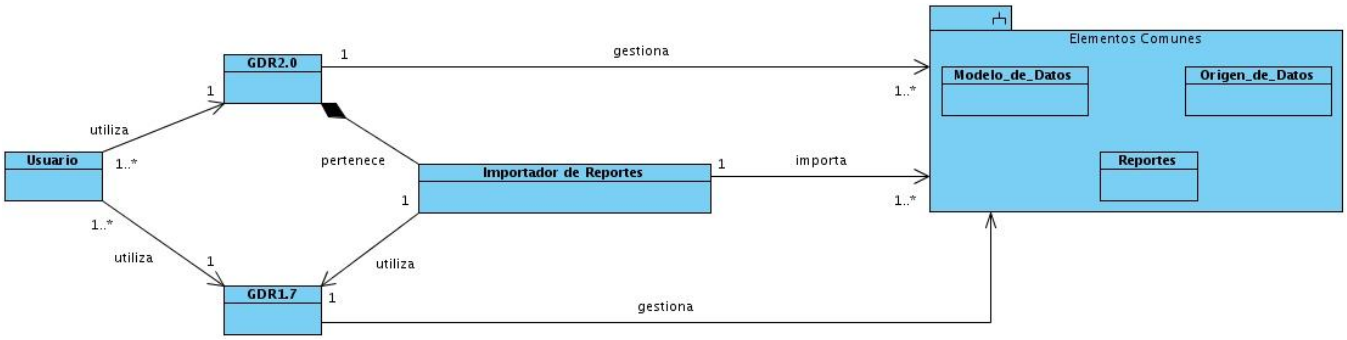


Figura 1 : Modelo de Dominio.

Conceptos del Modelo de Dominio:

Usuario: Persona responsable y capacitada para utilizar el sistema informático Generador Dinámico de Reportes en todas sus versiones.

Origen de datos: Contiene los datos que permiten conectar al Generador Dinámico de Reportes con los Sistemas Gestores de Bases de Datos. Las variables que maneja son: Tipo de Gestor de Base de Datos, Dirección IP del Servidor, Puerto de conexión al Servidor, Usuario, Clave y Base de Datos. Luego de obtenidos los datos permite generar el modelo de datos para obtener los reportes.

Modelo de datos: Se compone por un conjunto de entidades asociadas a un Origen de datos registrado en el sistema. El mismo contiene reportes generados por el sistema para su transformación de PhpReport a JasperReport.

Reporte: Es un archivo, generado por un motor de reportes, con diferentes estructuras que contiene los datos de acuerdo a un interés específico.

GDR1.7: Sistema Informático que garantiza la generación dinámica de reportes partiendo de datos persistentes en algún origen de datos soportado por el sistema. Ofrece la posibilidad de controlar el funcionamiento periódico de una o varias entidades, a través de la creación de reportes en distintos formatos.

GDR2.0: Sistema Informático que garantizará la generación dinámica de reportes partiendo de datos persistentes en algún origen de datos soportado por el sistema. Ofrecerá la posibilidad de controlar el funcionamiento periódico de una o varias entidades, a través de la creación de reportes en distintos formatos. El mismo contará con un Sistema de Importación de reportes para realizar la importación de reportes de una versión a otra.

Importador de Reportes**:** Sistema Informático encargado de mantener la continuidad de los reportes desde la versión 1.7 del Sistema Generador Dinámico de Reportes basada en PhpReport, a la versión 2.0 que utilizará JasperReport como motor de reportes.

2.2 Requerimientos

Un requerimiento no es más que esa condición que debe ser alcanzada o poseída por un sistema o componente de un sistema para satisfacer un contrato, estándar u otro documento impuesto formalmente. Se denomina como: "condición o capacidad que necesita un usuario para resolver un problema o lograr un objetivo, define qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen " (17). Con la representación de las clases más importantes en el Modelo de Dominio, se determinaron los requisitos funcionales y los no funcionales que debe cumplir el sistema, como a continuación se describen.

Requerimientos funcionales

Se identificaron los siguientes requisitos funcionales :

RF1. Conectar el Importador de Reportes a la base de datos.
RF2. Importar los de datos de la versión1.7 de GDR a la versión2.0 de GDR.
RF3. Listar reportes almacenados en la base de datos.
RF4. Buscar los reportes almacenados en la base de datos.
RF5. Transformar reportes de la versión1.7 de GDR a la versión2.0 de GDR.

Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Estas propiedades se describen como las características que hacen al producto atractivo, usable, rápido, confiable o eficiente con el hardware y software, con vista a las funcionalidades que el sistema debe brindar. (17) Se identificaron a raíz de estos elementos los siguientes requisitos no funcionales:

Usabilidad.

RNF 1 Buscar de manera rápida y sencilla un reporte que se desea visualizar

El usuario podrá localizar un reporte de manera rápida, si es posible tenerlo ubicado en alguna categoría que permita tener la posibilidad de organizarlos por áreas temáticas comunes.

RNF 1.1 Importar un reporte de manera sencilla y ágil

El usuario podrá importar un reporte de una versión a otra de manera ágil y sencilla sin necesidad de ser un experto en el uso de la herramienta.

Interfaz

RNF 2 El usuario debe acceder a la aplicación a través del protocolo HTTP usando el navegador Firefox versión 2.0 o superior.

Requerimientos de Software

RNF 3 El servidor donde se instalará la aplicación debe cumplir con los siguientes requisitos de software:

- Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debian 4 GNU/Linux o superior.
- Paquetes: apache2, php5, libapache2-mod-php5, php5-cli, php5-mysql, php5-pgsql, php5-sqlite, php5-sybase, php5-xsl, php5-gd, php-apc.
- Usuario con privilegios de administración del Sistema Operativo.

RNF 3.1 El servidor donde se instalará la Base de Datos del sistema debe cumplir con los siguientes requisitos de software (puede ser el mismo donde estará la aplicación):

- Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debian 4 GNU/Linux o superior.
- PostgreSQL versión 8.3 o superior.
- PGAdmin III o algún administrador para postgreSQL.
- Usuario con privilegios para instalar la base de datos.
- PostgreSQL debe estar correctamente configurado para aceptar conexiones vía TCP/IP utilizando el método de autenticación por md5.

Requerimientos de hardware

RNF 4 Las PC clientes deben cumplir con los siguientes requisitos de hardware:

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 256MB.

RNF 4 .1 Las PC servidor deben cumplir con los siguientes requisitos del hardware:

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 1Gb.
- Disco Duro con 10Gb de capacidad para instalar el sistema.

Requisitos de Licencia.

No se posee ningún requisito de licencia o restricción en el uso del software puesto que se desarrollará con la utilización de herramientas libres como: Symfony 1.4, Postgres 8.4, PgAdminIII, Netbeans 7.0.1 y ExtJS 3.4.

2.3 Diagrama de Caso de Uso del Sistema

A partir de los requisitos funcionales identificados anteriormente se modeló el Diagrama de Casos de Uso, donde se pone de manifiesto el patrón de caso de uso CRUD Parcial. Este patrón alternativo modela una de las vías de los casos de uso como un caso de uso separado. Es preferiblemente utilizado cuando una de las alternativas de los casos de uso es más significativa, larga o más compleja que las otras. Se representan las relaciones entre los actores y casos de uso del sistema evidenciándose los principales procesos del sistema (ver Figura 2***: Diagrama de Casos de Uso del Sistema.***).

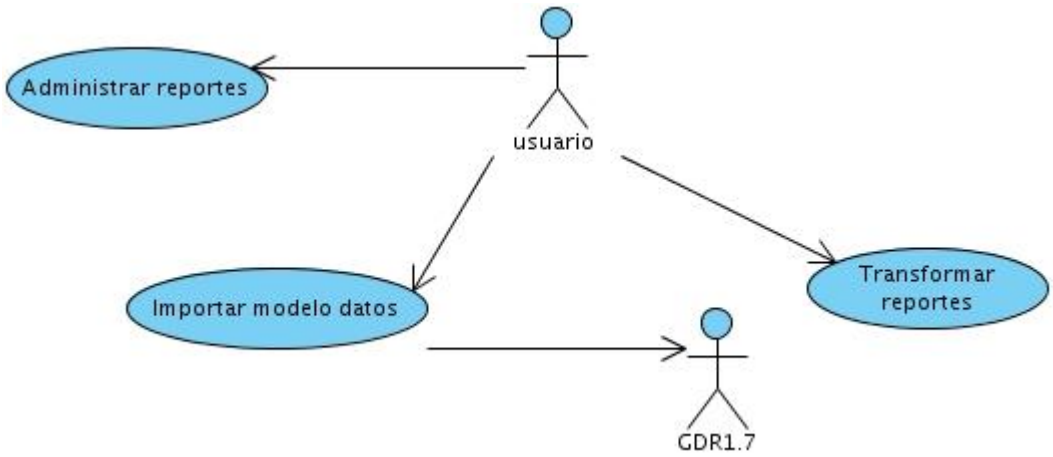


Figura 2: Diagrama de Casos de Uso del Sistema.

Actor	Descripción
Usuario	Ente capacitado en el uso del sistema GDR encargado de importar los reportes de una versión a otra.

GDR1.7	Ente encargado de facilitar algunas funcionalidades con las que cuenta GDR1.7 necesarias en la aplicación.

Tabla 2: Descripción de los actores del sistema.

CU3-Transfomar Reportes.

Objetivo	Transformar reportes almacenados en GDR1.7 a su nueva versión.	
Actores	Usuario.	
Resumen	El caso de uso se inicia cuando el actor selecciona la opción importador de reportes. Permite transformar los reportes almacenados en GDR1.7 para ser reutilizados en su nueva versión. Culmina al terminar de realizar cualquiera de estas operaciones.	
Complejidad	Alta.	
Prioridad	Crítico.	
Precondiciones	Debe existir al menos un reporte creado. Seleccionar al menos un reporte almacenado en la base de datos.	
Postcondiciones	Se transforman los reportes.	
Flujo de eventos		
Flujo básico Importar Reportes		
	Actor	**Sistema**
1.	Selecciona la opción “Importar”.	
2.		Muestra una interfaz brindando un listado de los reportes que han sido previamente importados.
3.	Selecciona un reporte de la lista de reportes que existen en el sistema.	
4.		Muestra una interfaz con la opción de Transformar.
5.	Selecciona la opción “Transformar”.	
6.		Muestra un mensaje indicando el éxito de la acción. Termina el caso de uso.

Flujos alternos		
Nº Evento En caso de que la transformación no se realice con éxito.		
	Actor	**Sistema**
1. 2.		Muestra un mensaje indicando el fallo en la transformación del reporte.

Tabla 3: Descripción del caso de uso Transformar Reportes.

2.4 Objetivos del diseño

El diseño tiene como principales objetivos comprender detalladamente los requisitos funcionales y no funcionales, sistemas operativos, tecnologías de distribución, restricciones relacionadas con el lenguaje o los de programación, tecnologías de interfaz de usuario. Además, crea un punto de partida para las actividades de implementación que siguen. Es el punto de mayor importancia al final de la fase de elaboración y el comienzo de las iteraciones de construcción. Permite facilitar una arquitectura estable y sólida, crear un plano muy cercano del modelo de implementación. Es necesario mantener el modelo de diseño a través de todo el ciclo de vida del software.

Propósitos del diseño:

- Adquirir una comprensión de los aspectos relacionados con los requisitos no funcionales y restricciones relacionadas con los lenguajes de programación, tecnologías de distribución y concurrencia, sistemas operativos, componentes reutilizables y tecnologías de interfaz de usuario.
- Crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.
- Descomponer los trabajos de implementación en partes más manejables que puedan ser llevadas a cabo por diferentes equipos de desarrollo.
- Capturar las interfaces entre los subsistemas antes en el ciclo de vida del software, lo cual es muy útil cuando utilizamos interfaces como elementos de sincronización entre diferentes equipos de desarrollo. (18)

2.5 Diagrama de Clases del Diseño

El diagrama de clases del diseño se encarga de representar los métodos y atributos de cada una de las clases del sistema, para mostrar de forma simple la colaboración y las tareas de cada una de ellas en relación al sistema que conforman. Partiendo de la descripción detallada de los casos de uso del sistema, se modelaron los diagramas de clases del diseño.

A continuación se muestra el diagrama de clases diseño del caso de uno de los casos de usos más significativos del sistema: Transformar Reportes (ver Figura 3**: Diagrama de Clases del Diseño del Caso de Uso Transformar Reportes.**):

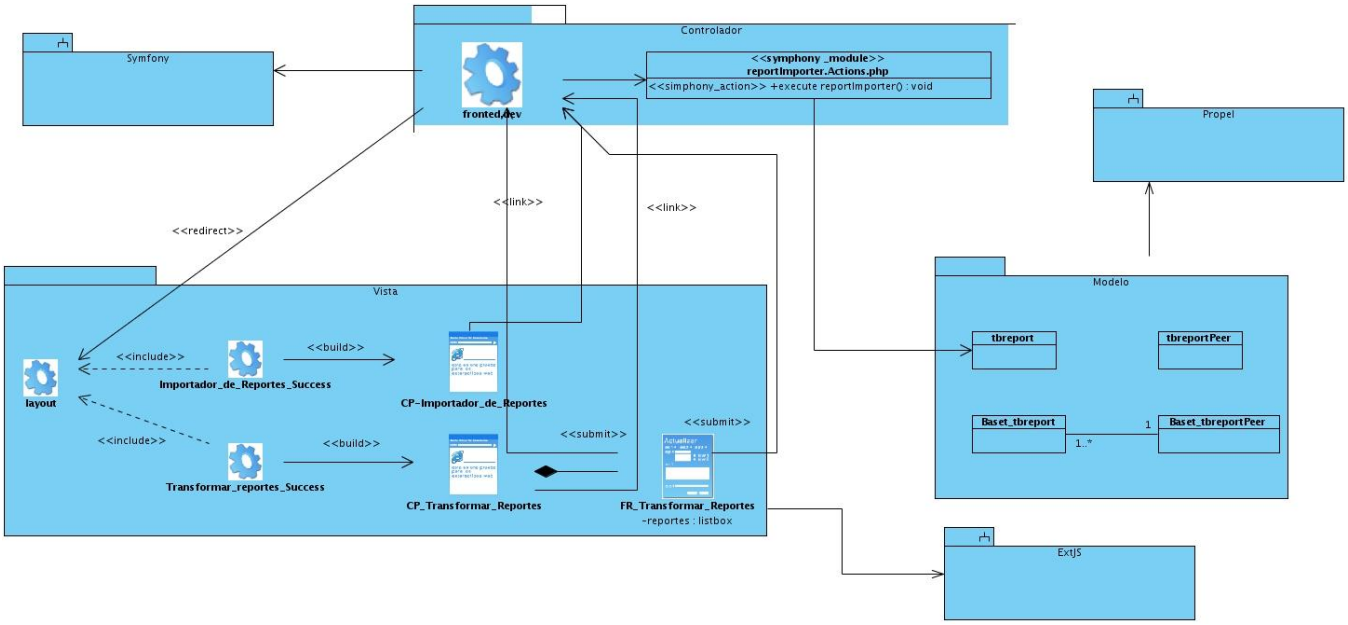


Figura 3: Diagrama de Clases del Diseño del Caso de Uso Transformar Reportes.

Nombre: Transformar_Reportes.Actions.php	
Tipo de Clase: Controladora.	
Para cada responsabilidad:	
Nombre	**Descripción**
+ execute_report_importer()	Brinda la posibilidad de transformar los reportes de la de versión actual a la nueva. Obtiene los datos de la tabla tbreport, que es la encargada de realizar las operaciones relacionadas con los

	reportes, pues contiene todas las propiedades de los mismos.

Tabla 4: Descripción de la capa Controlador del diagrama de clases del diseño Transformar Reportes.

Patrones arquitectónicos

Los patrones son de gran utilidad para describir las mejores prácticas, buenos diseños, y encapsulan la experiencia permitiendo reutilizar dicha experiencia. Constituyen mecanismos cuyo objetivo es la solución de problemas que ocurren repetidamente dentro de un contexto muy bien definido. Definen la estructura de un sistema software, los cuales a su vez se componen de subsistemas con sus responsabilidades, también tienen una serie de directivas para organizar los componentes del mismo sistema, con el objetivo de facilitar la tarea del diseño de tal sistema. Para crear un diseño eficiente se decidió utilizar uno de los patrones arquitectónicos más conocidos para el diseño Web, el Modelo-Vista-Controlador (MVC) el cual se aplica en la presente solución. El MVC permite separar las capas: Modelo (representa la información con la que trabaja la aplicación, es decir, su lógica de negocio), Vista (transforma el modelo en una página Web que permite al usuario interactuar con ella) y Controlador (se encarga de procesar las interacciones del usuario y realiza los cambios apropiados en el modelo o en la vista).

La calidad de diseño de la interacción de los objetos y la asignación de responsabilidades presentan gran variación. Las decisiones poco acertadas dan origen a sistemas y componentes pocos robustos y difíciles de mantener, entender, reutilizar o extender. Una implementación hábil se funda en los principios cardinales que rigen un buen diseño orientado a objetos.

Patrones de diseño GRASP

En los patrones generales de software para asignación de responsabilidades (GRASP) se codifican algunos de los principios, que se aplican al diseñar los diagramas de interacción. Una de las principales ventajas a señalar en el uso del framework Symfony es que está basado en patrones de diseño. Los patrones de diseño empleados en la solución pertenecen al conjunto de patrones GRASP y GOF, su utilización ayudó a refinar el diseño y a asignar las responsabilidades de las distintas clases de diseño, haciéndolas más sencillas, reutilizables y encapsuladas. A continuación se explican los patrones utilizados:

Experto: El patrón experto en información es el principio básico de asignación de responsabilidades. Indica que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento). Al incluir Propel como ORM, se generan las clases para la gestión de las entidades con las responsabilidades debidamente asignadas. Esta es precisamente la solución que propone el patrón Experto, pues cada una de estas clases cuenta con un conjunto de funcionalidades relacionadas directamente con la entidad que representan. (20)

Creador: En las clases Actions se encuentran las acciones definidas para el sistema de información a desarrollar divididas de forma organizada por módulos. En las acciones se crean los objetos de las clases que representan las entidades, evidenciando de este modo que las clases Actions representan el patrón "creador" de dichas entidades. (20).

Alta Cohesión: Symfony permite asignar responsabilidades con una alta cohesión, por ejemplo la clase Actions tiene la responsabilidad de definir las acciones para las plantillas y colabora con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades, es decir, está formada por diferentes funcionalidades que se encuentran estrechamente relacionadas proporcionando que el software sea flexible frente a grandes cambios. (20)

Controlador: Todas las peticiones Web son manejadas por un solo controlador frontal (sfAction), que es el único punto de entrada de toda la aplicación en un entorno determinado. Cuando el controlador frontal recibe una petición, utiliza el sistema de enrutamiento para asociar el nombre de una acción y el nombre de un módulo con la URL entrada por el usuario. (20)

Patrones de diseño GOF

Front Controller (Controlador frontal): El patrón Front Controller maneja todos los request en una aplicación. (21) Se evidencia en el report_importer.php, pues es el punto de entrada a la aplicación que recibe todas las peticiones y decide el flujo a seguir, carga la configuración y determina la acción a ejecutarse. Es único para cada aplicación y las acciones, que incluyen el código específico del contenido de cada página.

Decorador: Añade funcionalidad a una clase dinámicamente. Esto permite no tener que crear sucesivas clases que hereden de la primera incorporando la nueva funcionalidad, sino otras que la implementan y se asocian a la primera. (21)

El patrón Decorador se evidencia cuando la clase layout, definida en el diagrama de clases del diseño, contiene todo el código HTML y a su vez decora todas las plantillas a usar, en dependencia de la petición del usuario.

Command (Orden): La estrategia del patrón Command sugiere proporcionar una interfaz genérica para los componentes en los que el controlador puede delegar responsabilidades, minimizando el acoplamiento entre estos componentes. (21) En el framework Symfony se utiliza este patrón, el cual está representado por las acciones, por tanto cada acción, encapsula una petición en un objeto.

2.6 Vista Lógica

La Vista lógica proporciona una base para comprender la estructura y la organización del diseño del sistema, pues contiene las clases del diseño más importantes, organizadas por paquetes y subsistemas en capas de trabajo. La solución es representada en términos de paquetes y clases de diseño, como también la realización de los casos de uso, subsistemas de los casos de uso arquitectónicamente más significativos. Consiste en mostrar una propuesta de subsistemas en los que se dividirá la aplicación. La misma ofrece los principales componentes de diseño y sus relaciones de forma independiente, para así ofrecer un principio de implementación para la aplicación.

A continuación se muestra la vista lógica de las clases que abarcan el comportamiento arquitectónicamente significativo (ver Figura 4**: Vista lógica del sistema.**):

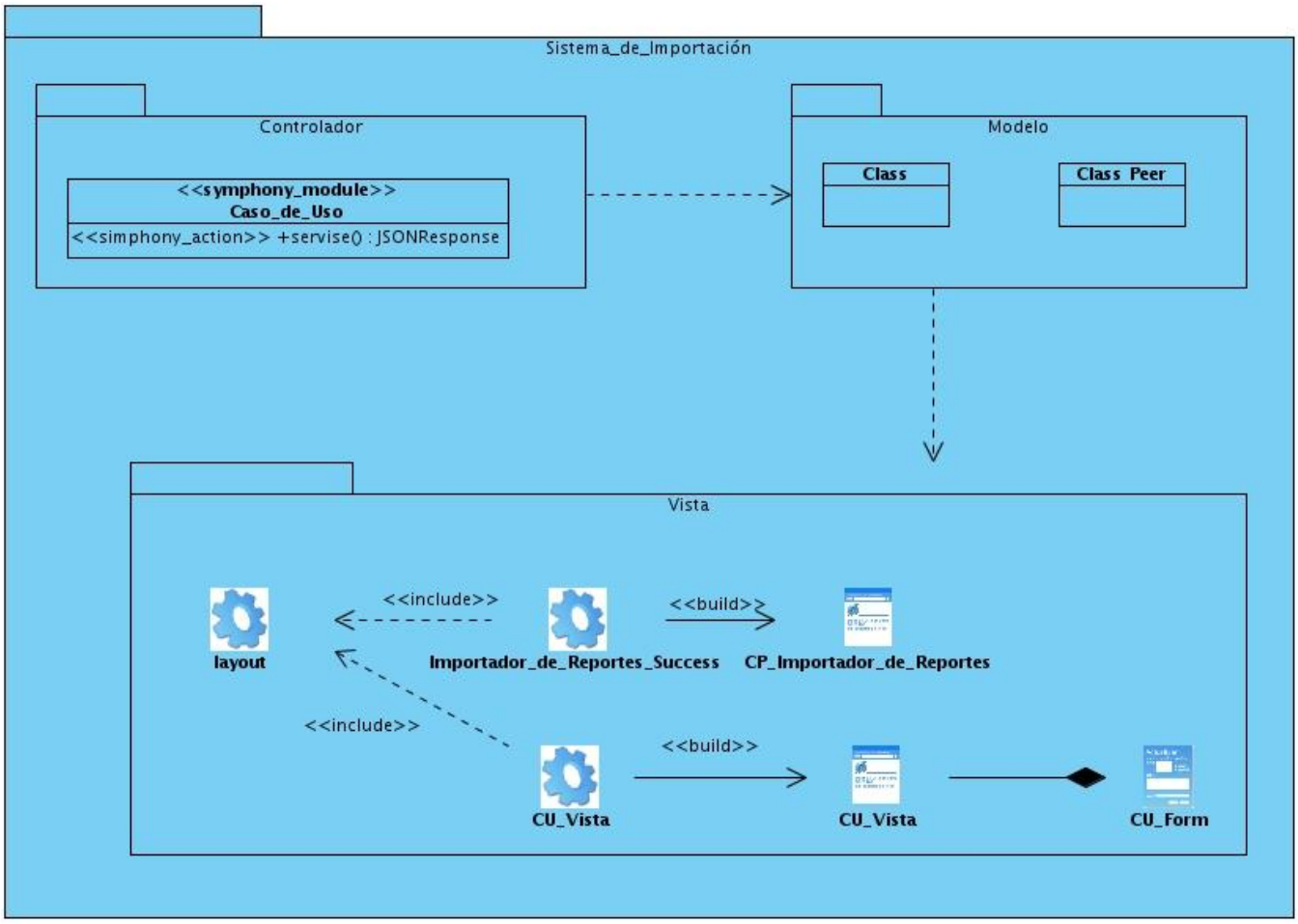


Figura 4: Vista lógica del sistema.

Con el fin de facilitar la implementación de la aplicación se utilizó el framework Symfony que tiene entre sus patrones más utilizados: el patrón Modelo-Vista-Controlador (MVC), logrando la organización del código, la reutilización y la flexibilidad. El MVC se representa a través de tres elementos fundamentales: el modelo, la vista y el controlador. (23)

La capa de la vista aprovecha la separación de código, por lo que se encuentra dividida en dos partes: la plantilla y el layout. La plantilla se encarga de visualizar las variables definidas en las acciones (estereotipado como <<symfony action>>) agrupadas en el controlador correspondiente (estereotipado como <<symfony module>>), mientras que el layout contiene el código HTML común a todas las páginas.

La capa del controlador, contiene el código que enlaza la lógica de negocio con la vista, está dividida en varios componentes que se utilizan con diversos propósitos:

- El controlador frontal es el único punto de entrada a la aplicación. Carga la configuración y

determina la acción a ejecutarse.

- Las acciones contienen la lógica de la aplicación. Verifican la integridad de las peticiones y preparan los datos requeridos por la capa de presentación.
- Los objetos request, response dan acceso a los parámetros de la petición, las cabeceras de las respuestas y a los datos persistentes del usuario.

Una vez ejecutada la petición la interfaz de red se encarga de realizar el sistema de enrutamiento a través de la comunicación HTTP, asociando el nombre de la acción y el nombre del módulo con la URL escrita por el usuario.

El modelo se encarga de la abstracción de la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes de, por ejemplo, el tipo de gestor de bases de datos utilizado por la aplicación. Las clases de la capa del modelo se generan automáticamente, en función de la estructura de datos de la aplicación. La librería Propel se encarga de esta generación automática, creando el esqueleto o estructura básica de las clases y genera automáticamente el código necesario. Cada tabla del modelo de datos genera 4 clases pero las verdaderas clases presentan la estructura de Clase.php y ClasePeer.php.

Las relaciones fundamentales entre estos elementos son de dependencia y las de flujo de datos. La relación de flujo de datos entre la interface y el modelo representa la correspondencia de las invocaciones del cliente con las acciones que del lado del servidor darán respuesta a las mismas, se transfiere tanto en la solicitud como en la respuesta el objeto con todos los atributos que se envían.

2.7 Diagramas de Secuencia

Los diagramas de secuencia ofrecen las interacciones entre objetos, ordenadas en secuencia temporal durante un escenario concreto. Describen el curso particular de los eventos de un caso de uso, proporcionando una realización física de comportamiento de los casos de uso en términos de clases del diseño. A continuación se muestra el diagrama de secuencias del caso de uso más significativo Transformar Reportes.

En el presente diagrama se muestran el flujo de acciones que ocurren en la aplicación al ejecutar la acción “Transformar” para Transformar un reporte dentro del caso de uso Transformar Reportes (ver Figura 5**: Diagrama de Secuencias Transformar Reportes.**).

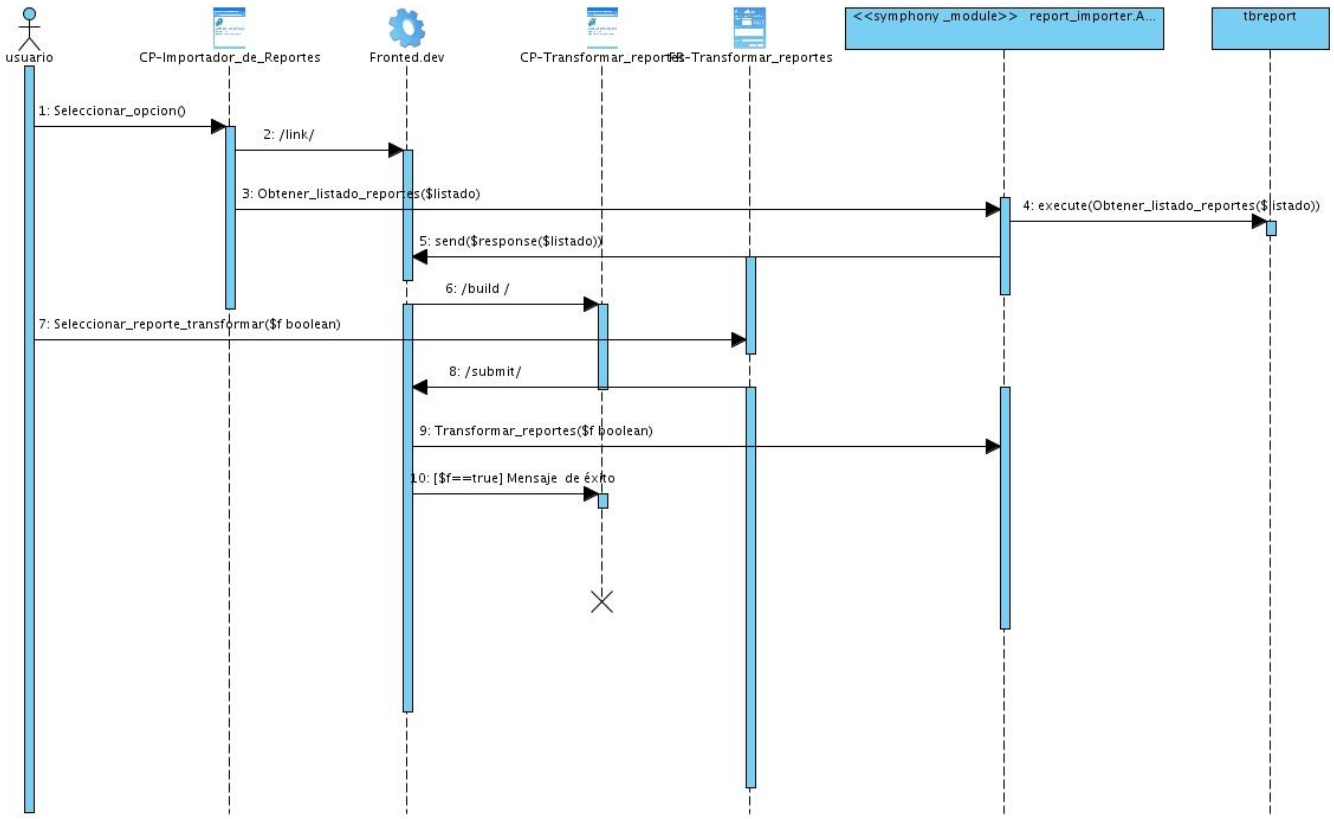


Figura 5: Diagrama de Secuencias Transformar Reportes.

2.8 Vista de Despliegue

El Sistema Generador Dinámico de Reportes está diseñado para que cada organización instale una instancia del sistema en sus servidores puesto que el mismo puede manejar información delicada y que lo utilice de acuerdo a sus necesidades teniendo en cuenta los niveles de integración previstos. El modelo de despliegue muestra la configuración de nodos de procesamiento en tiempo de ejecución, los enlaces de comunicación entre ellos, y el componente de los casos y los objetos que residen en ellos. Además proporciona un modelo minucioso de la estructura en la que los componentes se desplegarán a lo largo de la infraestructura del sistema. Permite detallar las capacidades de red, las especificaciones del servidor, los requisitos de hardware y otra información relacionada al despliegue del sistema propuesto. El mismo constituye el mapeo de componentes de software ejecutables con los nodos de procesamiento. Además, tiene en cuenta los siguientes requerimientos: disponibilidad del sistema, rendimiento y escalabilidad. A continuación se muestra la estructura de los componentes que se desplegarán a lo largo de la infraestructura del sistema (ver Figura 6**: Diagrama de Despliegue del sistema.**):

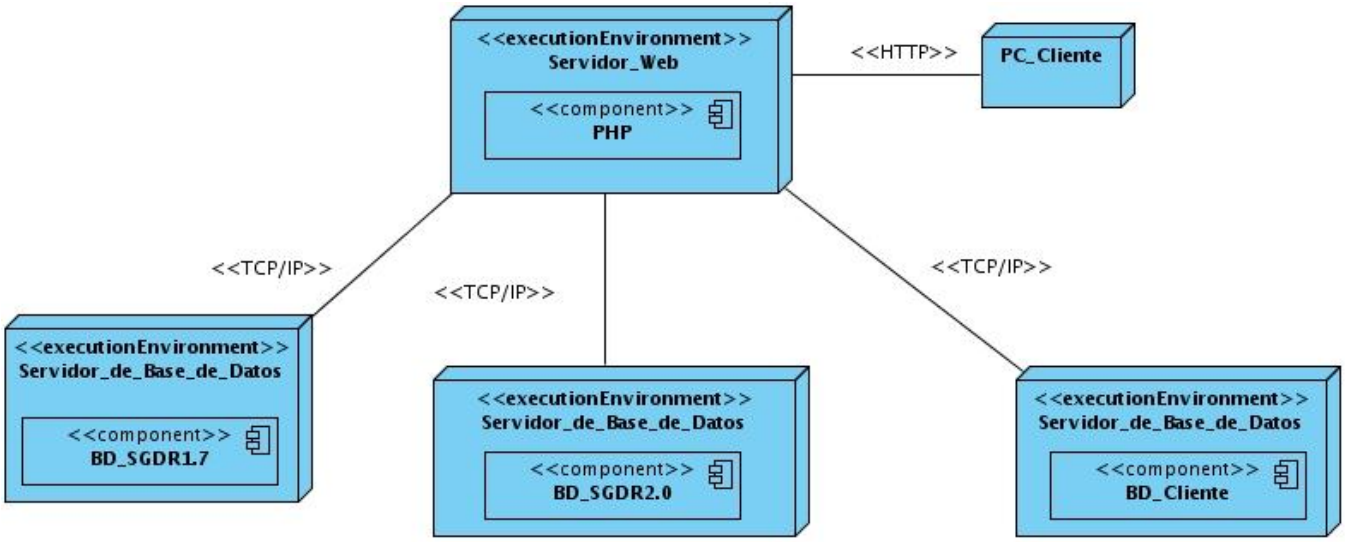


Figura 6: Diagrama de Despliegue del sistema.

Estaciones de trabajo con la aplicación cliente: Se refiere a las estaciones de trabajo que el usuario utilizará para acceder a la aplicación Web y transcribir sus datos.

Servidor de Aplicación: Servidor de aplicación utilizado para la publicación de la aplicación; y para lograr la conexión del sistema con la PC Cliente se utiliza protocolo de transferencia de hipertexto *[Hypertext Transfer Protocol]* (HTTP) como protocolo de comunicación. Es la herramienta principal para ejecutar la lógica de negocio en el lado del servidor. Es el responsable de ejecutar el código de las páginas servidor. Se utiliza el servidor de aplicación Apache Tomcat.

Protocolos de comunicación: Un protocolo de comunicación es un conjunto de reglas establecidas entre dos dispositivos para permitir la comunicación entre ambos.

- Conexión HTTP

  Es el protocolo utilizado entre los browser de los clientes y el servidor Web. Este elemento de la arquitectura representa un tipo de comunicación no orientado a la conexión entre clientes y servidor.

- TCP/IP

  Es la base del Internet que sirve para enlazar computadoras. El protocolo TCP/IP es utilizado para establecer la conexión entre el servidor de aplicación y el servidor de base de datos.

- SOAP

Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

Servidor de Base de Datos: Se refiere a un servidor que radica en cada nodo regional en el cual el cliente define que sean guardados los datos. En el servidor central estarán almacenados todos los datos recopilados por todos los nodos regionales.

2.9 Conclusiones parciales

En el presente capítulo fueron descritos el análisis y el diseño de la aplicación comenzando por el modelo de dominio, los requisitos funcionales y no funcionales, quedando confeccionado el diagrama de casos de uso del sistema. Se elaboró la vista lógica apoyada en las particularidades que tiene la arquitectura de Symfony por el patrón Modelo-Vista-Controlador. Partiendo de la realización de los casos de uso, se realizaron los diagramas de secuencia, reflejando un proceso lógico de acciones para resolver las peticiones del cliente. A través del diagrama de despliegue se describió como se realizará el despliegue de los componentes a lo largo de la infraestructura del sistema, para así dar comienzo a la implementación de la aplicación.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA.

3.1 Introducción

En el capítulo correspondiente a las disciplinas de implementación y pruebas, se analizarán los artefactos principales como el Modelo de Implementación y el Modelo de datos generado a partir de las clases persistentes. Se realizará una descripción de como los elementos del modelo de diseño se implementan en términos de componentes y se realizarán las pruebas para validar el mismo arrojando los principales resultados obtenidos a través de los casos de pruebas realizados a cada caso de usos.

3.2 Diagrama de clases persistentes

La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo. Las clases persistentes constituyen información de larga duración, o sea, son los datos que necesitan ser almacenados para ser gestionados en cualquier momento. Una vez identificadas las clases cuyos estados son los más trascendentes en el tiempo de vida de la aplicación a desarrollar, se crea el diagrama de clases y se seleccionan las que constituyen una necesidad conservar su estado; quedando modelado el diagrama de clases persistentes. Una de las principales características que posee este diagrama es describir la estructura del sistema mostrando sus clases, atributos y las relaciones entre ellas. Generalmente, estas clases tienen como origen las clases clasificadas como entidad, porque modelan la información del sistema y el comportamiento asociado de algún fenómeno o concepto. (26) A continuación se muestra el diagrama para el sistema a desarrollar:

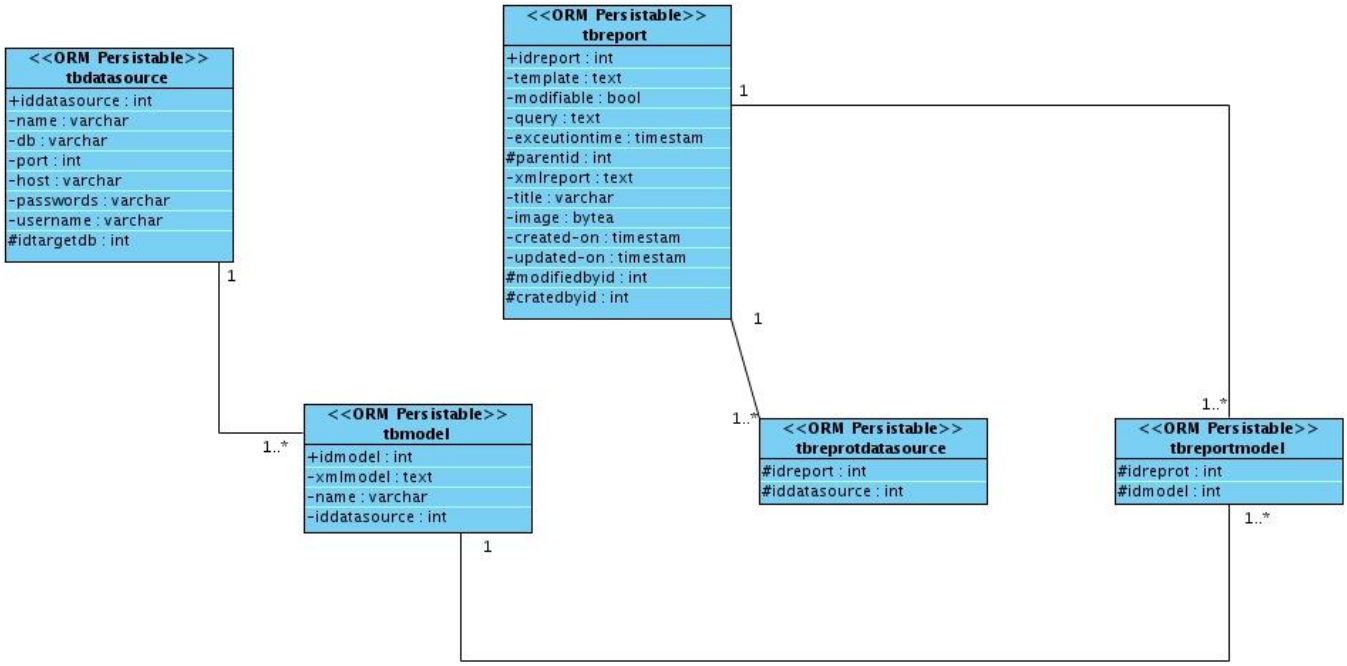


Figura 7: Diagrama de Clases Persistentes del Sistema.

3.3 Modelo de datos

Un modelo de datos es la estructura o representación física de las tablas de la base de datos obtenido a partir del diagrama de clases persistentes. Aporta la base conceptual para diseñar aplicaciones que hacen un uso intensivo de datos. De igual forma, permiten describir las estructuras de datos de la base, las restricciones de integridad y las operaciones de manipulación de los datos. Además brindan la base formal para las herramientas y técnicas empleadas en el desarrollo y uso de sistemas de información. (27). A continuación se muestra el modelo de datos del Importador de Reportes:

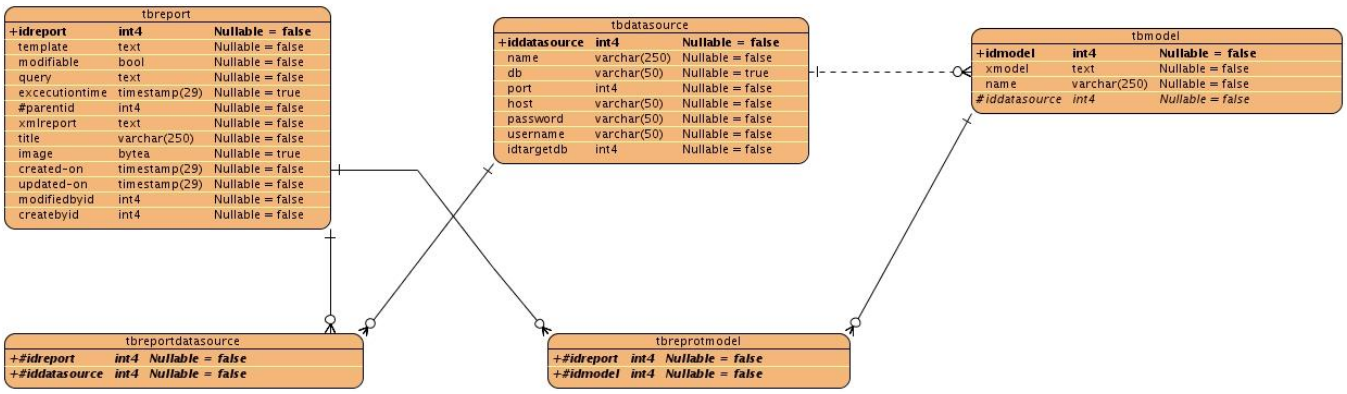


Figura 8: Modelo de Datos del Sistema.

Descripción de las tablas

Nombre: tbreport.		
Descripción: Almacena las categorías y características de los reportes.		
Atributo	**Tipo**	**Descripción**
Idreport	int4	Identificador auto-incremental.
template	Text	Contiene las plantillas de los reportes.
modifiable	Bool	Contiene las propiedades de modificación de los reportes.
query	Text	Contiene las consultas de los reportes.
excecutiontime	Timestamp	Contiene el tiempo de ejecución de un reporte.
parentid	Int4	Contiene el id padre o raíz de los Idreport.
xmlreport	Text	Contiene el XML del cuerpo del reporte.
title	varchar(250)	Contiene el titulo del reporte.
image	Bytea	Contiene las imágenes que tiene el reporte.
created-on	Timestamp	Tiempo de creación del reporte.
updated-on	Timestamp	Tiempo de modificación del reporte.
modifiedbyid	int4	Contiene los datos de modificación.
createbyid	int4	Contiene los datos de creación.

Tabla 5: Descripción de la tabla tbreport de la base de datos.

Nombre: tbdatasource.		
Descripción: Almacena los datos del nodo origen para el cual se almacenarán las informaciones de los reportes.		
Atributo	**Tipo**	**Descripción**
Iddatasource	int4	Identificador auto-incremental.
Name	varchar(250)	Contiene el nombre del reporte donde estará ubicado el nodo origen.
Db	varchar(50)	Contiene el nombre de la base de datos.
Port	int4	Puerto por la cual se conectará el servidor central.
Host	varchar(50)	Número de IP del nodo origen.
Username	varchar(50)	Usuario por el cual se conectara al nodo

		origen.
Password	varchar(50)	Contraseña del usuario.
Idtargetdb	int4	Identificador auto-incremental.

Tabla 6: Descripción de la tabla tbdatasource de la base de datos.

Nombre: tbmodel.		
Descripción: Almacena las categorías y características de los modelos de datos.		
Atributo	**Tipo**	**Descripción**
Idmodel	integer4	Identificador auto-incremental.
Xmlmodel	Text	Contiene el xml del cuerpo del modelo de datos.
Name	varchar(250)	Contiene el nombre del modelo de datos almacenado en la bd.
Iddatasource	int4	Identificador auto-incremental.

Tabla 7: Descripción de la tabla tbmodel de la base de datos.

Nombre: tbreportdatasource.		
Descripción: Almacena los reportes y los orígenes de datos.		
Atributo	**Tipo**	**Descripción**
Idreport	int4	Identificador auto-incremental.
Iddatasource	int4	Identificador auto-incremental.

Tabla 8: Descripción de la tabla tbreportdatasource de la base de datos.

Nombre: tbreportmodel.		
Descripción: Almacena los reportes y los modelos de datos.		
Atributo	**Tipo**	**Descripción**
Idreport	int4	Identificador auto-incremental.
Idmodel.	int4	Identificador auto-incremental.

Tabla 9: Descripción de la tabla tbreportmodel de la base de datos.

3.4 Transformación de PhpReport a JasperReport.

En el GDR1.7 se utiliza actualmente como motor de reportes a PhpReport, un reporte generado por esta herramienta, cuenta con un XML dividido en capas que contienen unas a otras. Las principales

etiquetas que conforman el XML, son <DOCUMENT> la cual almacena las principales propiedades que presentará el documento del reporte tales como: Encabezado <HEADER> y Pie de Página <FOOTER>. Estas a su vez cuentan con atributos para darle formato al reporte, agrupados en columnas <COL> que están contenidas en bandas <BAND>. Otros de los nodos básicos en el XML de un reporte con formato PhpReport es la etiqueta <PAGE> encargada de los elementos que almacenan las páginas. La banda <GROUPS> muestra los grupos anidados <GROUP> los cuales presentan la estructura anterior en cuanto a encabezado y pie de página, además cuentan con elementos <FIELDS> para introducir datos y mostrar informaciones. PhpReport establece una estructura de árbol donde cada uno de los nodos depende de la raíz, por lo que todas las bandas anteriormente explicadas se encuentran embebidas dentro de la banda <REPORT> y se pueden encontrar de forma recursiva como se encuentra representado en la siguiente figura:

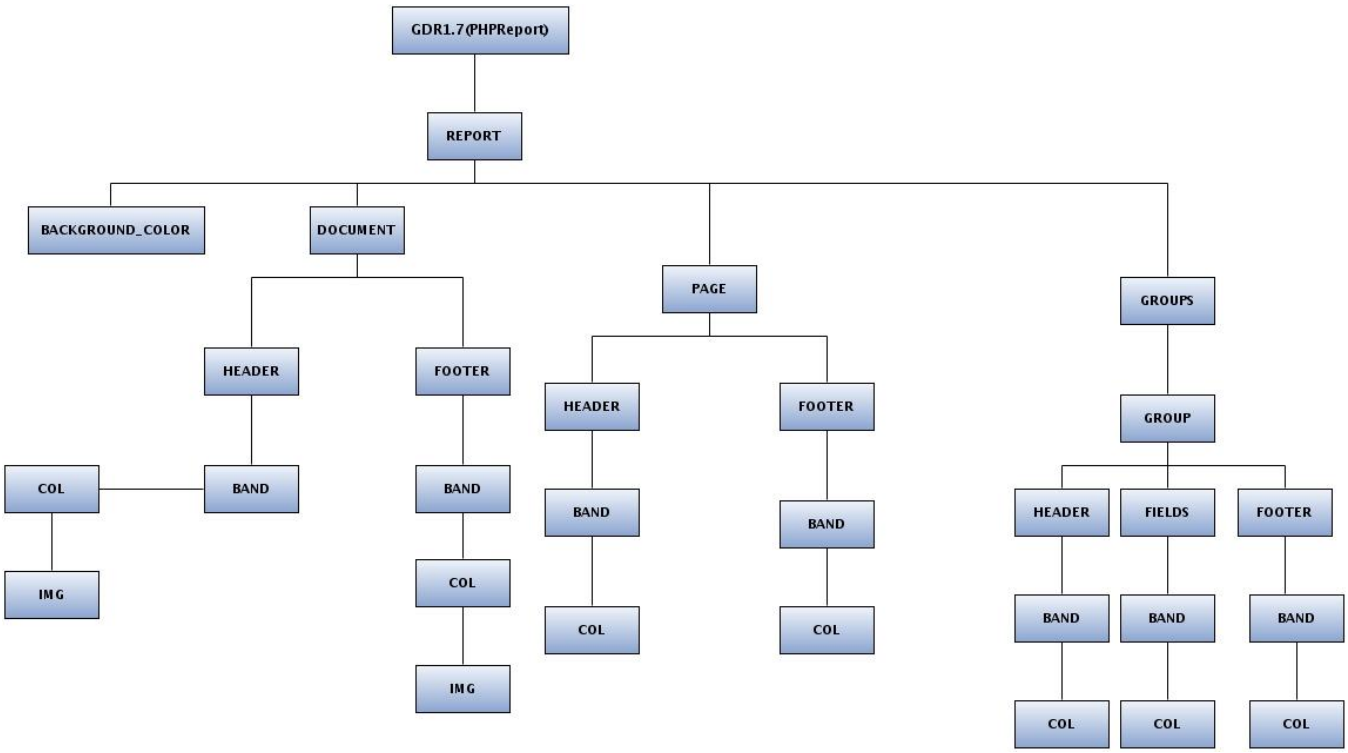
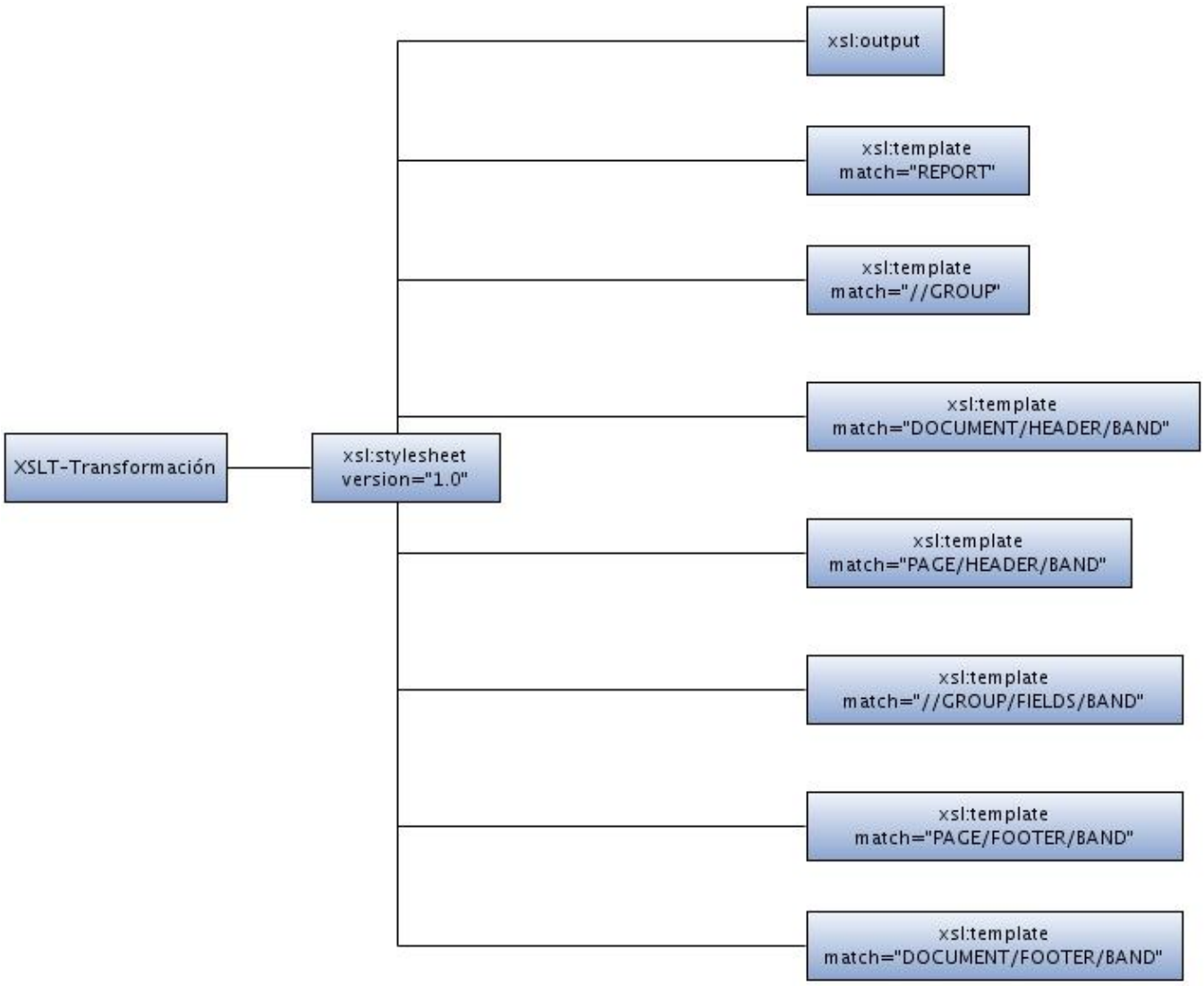


Figura 9: Estructura del XML en GDR1.7.

La tecnología XSLT se utiliza en la transformación de los XML de reportes con formatos PhpReport a los XML de reportes con formato JasperReport. Mediante la búsqueda de las etiquetas especificadas anteriormente, a través de los templates. El atributo match (<xsl:template match=" ">) indica la ruta a la cual se le aplicará la transformación, indicando los cambios, modificaciones y nuevas propiedades que

presentará el XML transformado. Para identificar la estructura y las reglas que deben presentar los archivos XML con formato JasperReport se utilizó el Gestor de Reportes iReport.



JasperReport genera los XML con una estructura divida en niveles, cada uno de los nodos cuenta con sus propiedades y se encuentran ubicados según el orden correspondiente, de esta forma las etiquetas se generan de forma independiente una de la otra. Las secciones principales con las que cuenta el documento XML, son el titulo del Reporte <title>, encabezado del documento <pageHeader>, encabezado de las columnas <columnHeader>, cuerpo del documento <detail>, pie de columna <columnFooter >, pie del documento <pageFooter>, cierre del documento <summary>, este es el formato y el orden en el que este motor genera sus etiquetas, entre otros elementos que se muestran en la figura (ver Figura 10**: Estructura del XML en GDR2.0.**). Dentro de cada una de estas

etiquetas podemos encontrar delimitadores (tags), similares al lenguaje HTML, para indicar campos de texto, como el tamaño de letra, el tipo de letra, ancho y altura entre otras propiedades. Incluso, se pueden encontrar consultas realizadas a la base de datos, referencias a parámetros y variables.

Luego de aplicarle la transformación mediante la tecnología de XSLT a los XML de los reportes almacenados en GDR1.7. En asociación con la estructura presentada por el XML que genera el motor de reportes JasperReport, guiado por la herramienta iReport, queda con el formato que se muestra en la siguiente figura:

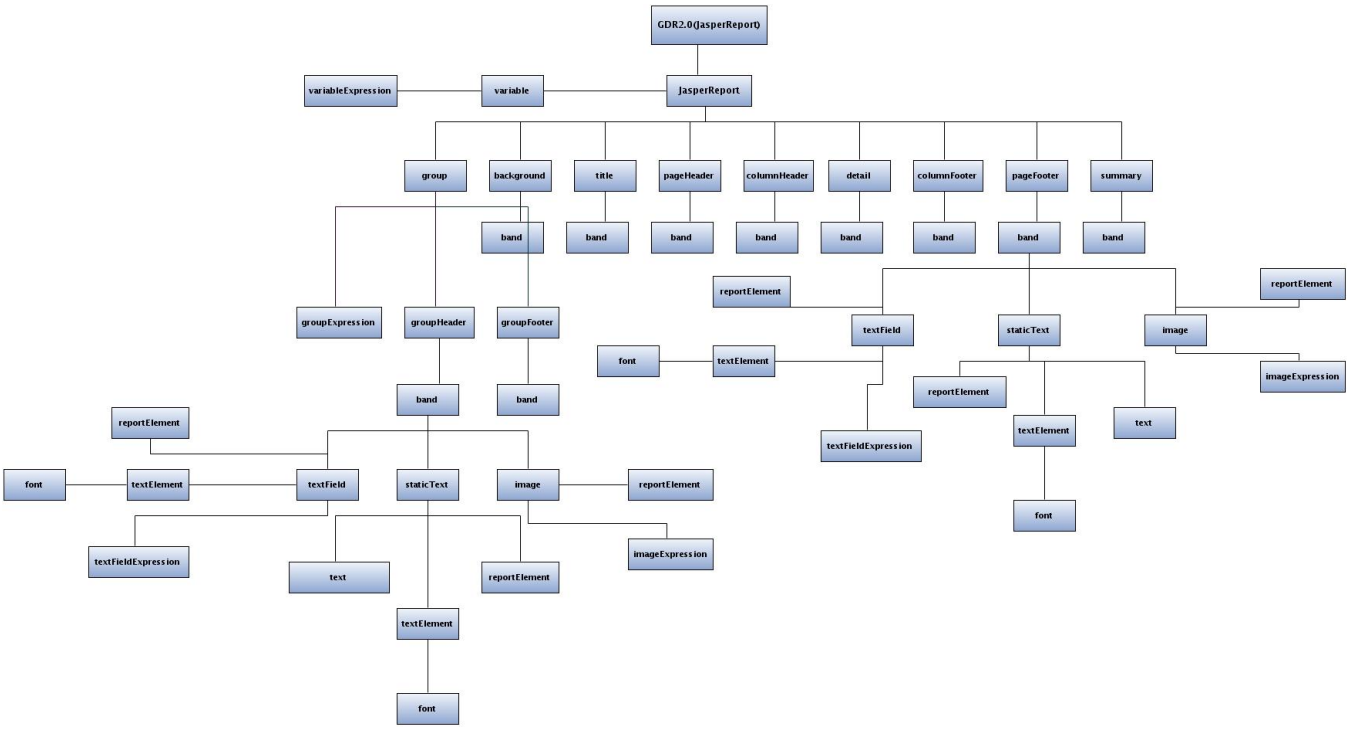


Figura 10: Estructura del XML en GDR2.0.

3.5 Modelo de Implementación

El Modelo de Implementación describe cómo los elementos del modelo de diseño, se implementan en términos de componentes, ficheros de código fuente, ejecutables, entre otras. Describe también cómo se organizan los componentes de acuerdo a los mecanismos de estructuración disponibles en el entorno de implementación y lenguaje o lenguajes de implementación empleados, y cómo dependen los componentes unos de otros (28). Es de gran utilidad a la hora de implementar el sistema, pues facilita la organización del producto y lo hace más entendible a los desarrolladores.

3.5.1 Diagrama de Componentes

Dentro del Modelo de Implementación se encuentran los diagramas de componentes. Un diagrama de componentes representa cómo un sistema de software es dividido en componentes, mostrando las dependencias que existen entre ellos. Los componentes físicos incluyen: archivos, cabeceras, bibliotecas compartidas, módulos, ejecutables y paquetes.
Los diagramas de componentes prevalecen en el campo de la arquitectura de software, pero pueden ser usados para modelar y documentar cualquier arquitectura de sistema; estos son utilizados para modelar la vista estática y dinámica de un sistema. Muestran la organización y las dependencias entre un conjunto de componentes (25).

Estructura general del sistema.

La aplicación report_importer está compuesta por 4 paquetes de componentes: el Web, actions, Model y XSL. La carpeta Web presenta la librería ExtJS, y las clases relacionadas con la vista del sistema a desarrollar. En la carpeta actions se encuentran los ficheros php representando las funcionalidades o acciones con los objetos del negocio que le dan funcionalidad al sistema. El componente Model está compuesto por las clases relacionadas con el acceso a datos de la aplicación. En la carpeta XSL se encuentran los ficheros relacionados con la transformación de los reportes de un formato a otro (ver Figura 11**: Diagrama de componentes del sistema.**)

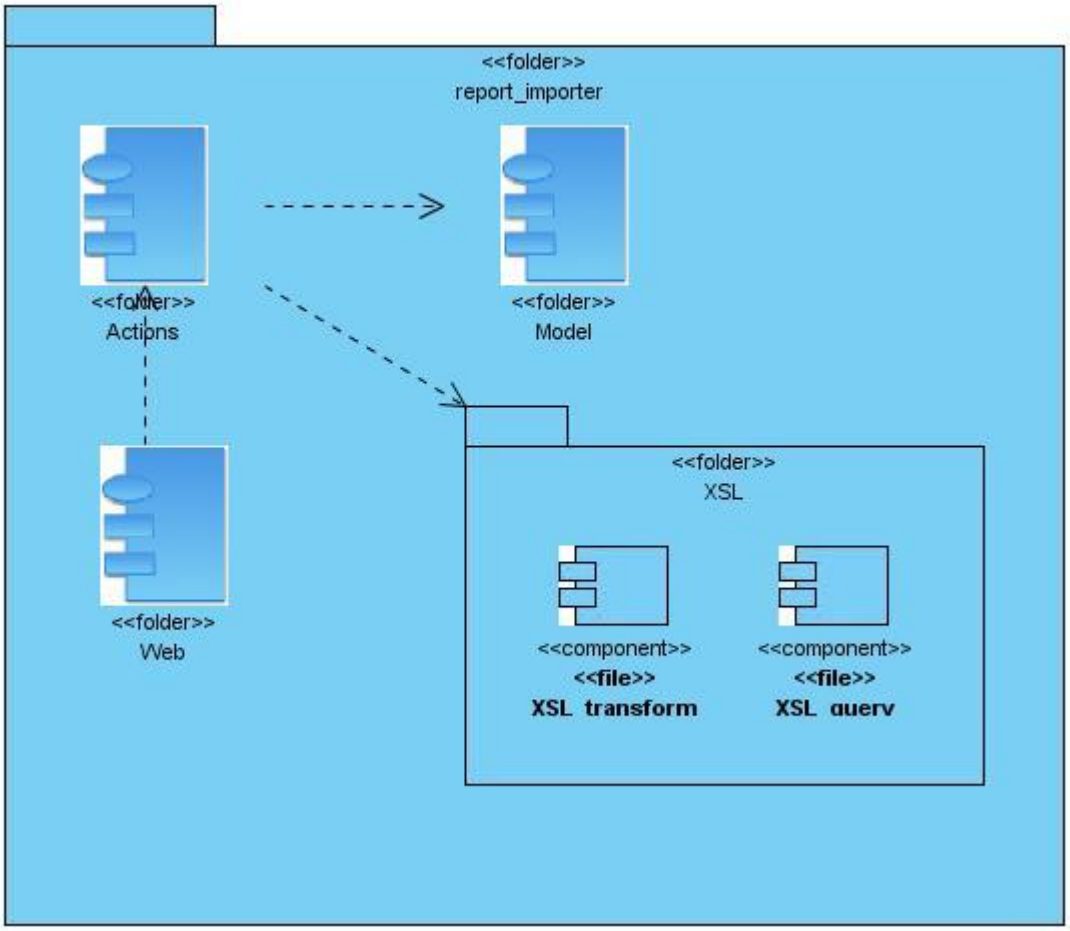


Figura 11: Diagrama de componentes del sistema.

A continuación se muestra el diagrama de componentes correspondiente al caso de uso Transformar reportes (ver Figura 12**: Diagrama de componentes del caso de uso Transformar Reportes.**).

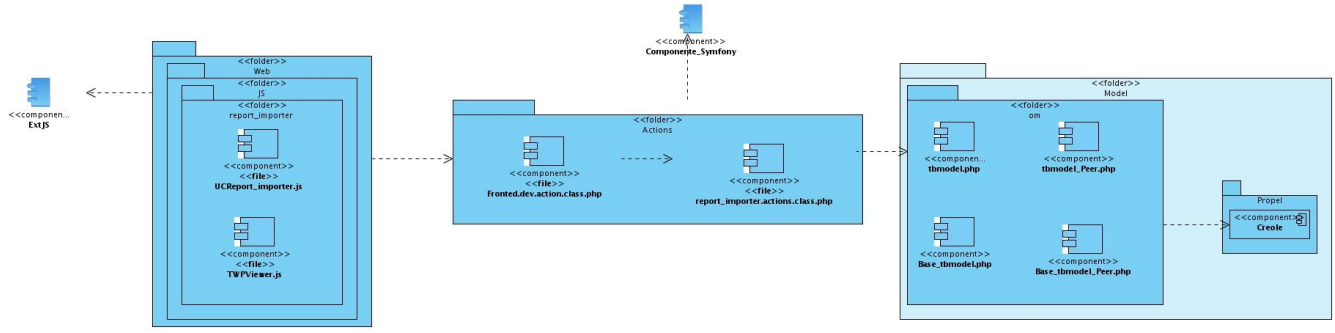


Figura 12: Diagrama de componentes del caso de uso Transformar Reportes.

En el diagrama representado se observa la capa de la vista la cual está compuesta por los componentes comunes del framework de ExtJS que genera automáticamente las clases ext_all.js y ext_base.js, que son utilizadas en el diseño del sistema. También en esta carpeta se almacenan todos los archivos de tipos javascript, css y las imágenes que la componen relacionados con la vista de la aplicación. La capa de acceso a datos está compuesta por las clases que son generadas por el framework de acceso a datos Propel y por las librerías resultado del mapeo de la respectiva base de datos de GDR. Propel genera automáticamente las clases de la capa del modelo, creando el esqueleto o estructura básica de las clases y genera automáticamente el código necesario. La abstracción de la base de datos es completamente invisible al programador, pues la realiza otro componente específico llamado Creole. De esta manera, si se cambia el sistema gestor de bases de datos en cualquier momento, no se debe reescribir ni una línea de código, tan sólo es necesario modificar un parámetro en un archivo de configuración. Estas clases generadas se archivan en las carpetas de Modelo de Objetos (OM) y Mapeo a la Base de Datos (MAP). La carpeta OM contiene las clases base del modelo, generadas una vez que se analiza el esquema de la base de datos. También se muestra el framework Symfony representado como paquete de componentes, pues el mismo es necesario en la ejecución de las diferentes aplicaciones que representan los archivos .php con las acciones y objetos del negocio que le dan funcionalidad al sistema. Además se observa como los componentes relacionados con las acciones utilizan el intérprete de PHP integrado al servidor Apache con la extensión php5_pgsql para acceder al servidor de base de datos PostgreSQL.

3.6 Ejemplos de Códigos

El servicio Importar modelos de datos mostrado con el nombre execute, recibe cómo parámetro un objeto request en formato JSON con el valor de los campos necesarios para establecer la conexión a la base de datos. Luego de lograr la conexión se realiza un backup a las tablas especificadas y son restauradas en la base de datos de GDR2.0. De esta forma se guardan los modelos de datos con sus

orígenes de datos y reportes asociados. Finaliza la ejecución con el retorno de un objeto response que contiene el formato JSON con el tipo de mensaje en dependencia si se efectuó o no satisfactoriamente el servicio.

```php
<?php

class ConnectionAction extends sfAction {

    public function execute($request) {
        try {
            $puerto = $request->getParameter("puerto");
            $dirIP = $request->getParameter("dirIP");
            $nomBD = $request->getParameter("nombreBD");
            $user = $request->getParameter("usuario");
            $pass = $request->getParameter("pswField");

            $connection = pg_connect("host=$dirIP  port=$puerto dbname=$nomBD user=$user password=$pass");
            $command = "PGUSER=$user PGPASSWORD=$pass pg_dump --host=$dirIP --port=$puerto --table=tbreport --
table=tbmodel --table=tbdatasource --table=tbcategory --table=tbreportmodel --table=tbreportdatasource --
table=tbcategoryreport -a --disable-triggers --format=custom  --file=/var/www/gdr2/test.dump $nomBD ";
            exec($command);
            $command = "PGUSER=$user PGPASSWORD=$pass pg_restore --host=$dirIP --port=$puerto -a --disable-
triggers --superuser=$user --dbname=gdr1.8.1 /var/www/gdr2/test.dump";
            exec($command);
            return $this->renderText("{success:true, data:{msg:'se pudo realizar la conexión.'}}");
        } catch (Exception $e) {
            return $this->renderText("{success:false, errors:{msg:'No se pudo realizar la conexión.'}}");
        }
    }

}


?>
```

El servicio Transformar reportes mostrado con el nombre execute, recibe cómo parámetro un objeto request en formato JSON con el valor del id del reporte seleccionado. Permitiendo el acceso a las propiedades del reporte que son captados por el identificador, entre los que se encuentra el XML del reporte, la consulta y el tipo de gestor de base de datos. Luego de haber establecido los parámetros se obtiene la ruta del archivo de transformación de la consulta y del XML del reporte aplicándole la transformación correspondiente. Almacenando este resultado en la base de datos mediante la modificación del campo que contiene el XML del reporte como resultado del servidor. Finaliza la ejecución con el retorno de un objeto response que contiene el formato JSON con el tipo de mensaje en dependencia si se efectuó o no satisfactoriamente el servicio.

```
class reportImporterAction extends sfAction {

    public function execute($request)
    {
        try {
            $reportID = $request->getParameter('id');//Recogida de datos//
            $report = Report::getById($reportID);
            $queryXML = $report->getQuery();
            $reportXML =  $report->getXmlreport();
            $interface = $report->getDataSource()->getGestorType();

            $xml = new DOMDocument ( );
            $xml->loadXML($queryXML);

            $proc = new XSLTProcessor ( );
            $xsl = sfConfig::get('sf_root_dir') . DIRECTORY_SEPARATOR . 'XSLtrans' . DIRECTORY_SEPARATOR .
$interface . '_sql.xsl';
            $proc->importStylesheet(DOMDocument::load($xsl));
            $sql = $proc->transformToXml($xml);


            $xml = new DOMDocument ( );
            $xml->loadXML($reportXML);
            $proc = new XSLTProcessor ( );
            $xsl = sfConfig::get('sf_root_dir') . DIRECTORY_SEPARATOR . 'XSLtrans' . DIRECTORY_SEPARATOR  .
'first.xsl';
            $proc->importStylesheet(DOMDocument::load($xsl));
            $setParameter = $proc->setParameter("", 'tipo', $sql);
            $jasper = $proc->transformToXml($xml);
            $report->setXmlreport($jasper);
            $report->save();
```

Figura 13: Ejemplo de código del método reportImporter.

3.7 Modelo de Prueba

Las pruebas de software constituyen un pilar indispensable para evaluar la calidad de un producto a través de la búsqueda y documentación de errores, certificando el cumplimiento de requerimientos. Además se puede definir como una actividad en el cual un sistema o componente es ejecutado bajo unas condiciones específicas, los resultados son observados y registrados. Las pruebas son especificadas mediante un documento que establece las condiciones de ejecución, las entradas de la prueba y los resultados esperados. Se aplican durante todo el ciclo de desarrollo del software para diferentes objetivos y en distintos niveles de trabajo.

El propósito de esta disciplina es:

- Proporcionar retroalimentación temprana y frecuente sobre si el sistema cumple los requisitos.
- Objetivamente medir los progresos realizados en pequeños incrementos.
- Identificar los problemas con la solución, aumentando la probabilidad de que el sistema se comportará correctamente.
- Asegurar que los cambios en el sistema no introducen nuevos defectos.

- Mejorar la velocidad, facilitando el descubrimiento de los temas con los requisitos, diseños e implementaciones lo más pronto posible.

En la evaluación dinámica del sistema se aplicó el nivel de trabajo Pruebas de Desarrollador, el cual está diseñado e implementado por el equipo de desarrollo. Cada nivel de prueba engloba una técnica de prueba específica según los atributos de calidad que se deseen verificar con las pruebas al software. La técnica de prueba seleccionada para evaluar los servicios que ofrece el sistema es la de Prueba funcional, la cual tiene como principal objetivo asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. A través del método de Caja Negra, también conocido como Pruebas de Comportamiento, se ejecutará cada caso de uso usando datos válidos e inválidos, para verificar que los resultados esperados ocurran cuando se usen datos válidos y se desplieguen los mensajes apropiados de error.

Para confeccionar los casos de prueba de Caja Negra se utilizó el criterio de la técnica de Partición de Equivalencia, donde se divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. En esencia, esta técnica intenta dividir el dominio de entrada de un programa en un número finito de variables de equivalencia. De tal modo que se pueda asumir razonablemente que una prueba realizada con un valor representativo de cada variable es equivalente a una prueba realizada con cualquier otro valor de dicha variable. Las variables de equivalencia representan un conjunto de estados válidos y no válidos para las condiciones de entrada de un programa. Se definen dos tipos de variables de equivalencia: las válidas, que representan entradas válidas al programa, y las no válidas, que representan valores de entrada erróneos, aunque pueden existir valores no relevantes a los que no sea necesario proporcionar un valor real.

Casos de prueba

Un caso de prueba se diseña según las funcionalidades descritas en los casos de uso. El propósito que se persigue con este artefacto es lograr una comprensión común de las condiciones específicas que la solución debe cumplir. Se parte de la descripción de los casos de uso del sistema, como apoyo para las revisiones. Cada planilla de caso de prueba recoge la especificación de un caso de uso, dividido en secciones y escenarios, detallando las funcionalidades descritas en él y describiendo cada variable. Además quedan plasmados las revisiones realizadas al caso de prueba; así como un registro de todo aquello que no corresponde a la calidad del software. Se efectuaron los casos de pruebas a los 4 casos de uso del sistema, plasmándose en el expediente de proyecto de la fase de pruebas en la planilla de caso de pruebas del proyecto. (Ver planilla Diseño de Casos de Prueba)

A continuación se presenta la tabla de secciones a probar en el caso de uso Transformar Categoría:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Transformar reportes	EC 1.1: Transformar reportes	El usuario decide transformar el reporte seleccionado, el sistema envía un mensaje indicando que la petición se realizó exitosamente, terminando así el CU.

Tabla 10: Secciones a probar en el Caso de Uso Transformar Reportes.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Lista de reportes.	Listbox	No	Será un reporte que se encuentre almacenado en la base de datos.

Tabla 11: Descripción de las variables.

Esta descripción posibilitó que se realizara una matriz de datos, donde se evaluó y probó la validez de cada uno de los datos introducidos en el sistema, específicamente en la sección que se estuvo probando. Utilizando un juego de datos válidos e inválidos se identificó el empleo de la técnica de partición de equivalencia.

Matriz de Datos

Escenario	Descripción	1	Respuesta del sistema.	Flujo central
EC 1.1 Transformar reportes de forma correcta.	Se transforman correctamente los reportes.	V (Se selecciona el reporte)	Se establece la transformación del reporte seleccionado de forma correcta.	1-Selecciona la opción “Importar”. 2-Se selecciona el reporte. 3-Selecciona la opción transformar.
EC 1.2 Transformar reportes de forma	Se transforman los reportes sin ser seleccionados. Se	( )	Se muestra un mensaje que la transformación no se	1-Selecciona la opción “Importar”. 2-No se selecciona

correcta.	muestra un mensaje de error.		realizó.	ningún reporte. 3-Selecciona la opción transformar. 4-Se muestra un mensaje de error en la transformación.

Tabla 12: SC Transformar Reportes.

Luego de aplicar las pruebas funcionales se arrojan diferentes resultados, pero en esencia, se busca mayor eficacia y eficiencia a la hora de encontrar defectos, verificando la calidad, de forma tal que se minimicen tiempos, costos, y se obtenga un resultado satisfactorio. El sistema desarrollado después de aplicarle las pruebas arrojó 6 no conformidades significativas, las cuales fueron resueltas, culminando el presente trabajo con resultados satisfactorios.

3.8 Conclusiones Parciales

Como resultado de este capítulo se obtuvieron los diagramas de componentes a través de la estructura que presenta los diagramas de clases del diseño. De esta forma se realizó la implementación del sistema en términos de componentes, ofreciendo solución a los requisitos especificados en el capítulo anterior. Se representaron las clases persistentes y la descripción de la estructura de las tablas sobre las que se establece el modelo de datos del sistema, así como sus atributos y relaciones. Finalmente se realizaron los casos de pruebas, luego de la implementación del sistema, para validar la completitud de los requerimientos, obteniéndose 6 no conformidades significativas, las cuales fueron resueltas, culminando el presente trabajo con resultados satisfactorios.

CONCLUSIONES

Para el desarrollo del componente para la importación de reportes en el Generador Dinámico de Reportes en su versión2.0:

- Se analizaron los preceptos teóricos y técnicos que sustentaron la investigación, para realizar la herramienta de importación que se integrará a GDR en su versión 2.0, lo que permitió la selección de la metodología, las herramientas y tecnologías que se aplicaron durante la implementación del sistema.
- A partir de las funcionalidades descritas se realizó el análisis y el modelo de diseño de la herramienta para importar reportes, proporcionando el punto de partida para las actividades de implementación.
- Se implementó el componente para importar reportes posibilitando la transformación de reportes desde la versión 1.7 del Generador Dinámico de Reportes a su versión 2.0.
- Las pruebas realizadas para validar las funcionalidades que fueron implementadas, para desarrollar la herramienta de importación de reportes, demostraron que los indicadores de calidad cumplieron satisfactoriamente con los requisitos, garantizando su correcto funcionamiento.

RECOMENDACIONES

Después de haber cumplido con los objetivos trazados en el presente trabajo de diploma se proponen las siguientes recomendaciones:

- Integrar la herramienta de Importador de Reportes a la nueva versión (2.0) del Generador Dinámico de Reportes.
- Integrar la herramienta Importador de Reportes en GDR2.0 con la aplicación Módulo Visor de Reportes.

REFERENCIAS BIBLIOGRÁFICAS

1. **Ramírez, JGE and Morales, EM.** Oncena Semana Tecnológica. [Online] 2010. [Cited: 10 2, 2011.] semanatecnologica.fordes.co.cu.Bibliografía

2. **Danciu, Teodor.** *The Jasper Reports Ultimade Guide.* 2002.

3.**Welling, Luke.** Desarrollo Web con php5. [En línea] 2005. [Citado el: 2 de 9 de 2011.] orton.catie.ac.cr.

4.**Pacheco Escribano, Lorenzo Fernandez.** Ejercicios prácticos tutorizados. [En línea] books.google.com.

5. **Karen.** openup-como-alternativa-metodolgica. [En línea] 2008. kasyles.blogspot.com. 09.

6. **Villa, Crysfel.** Ext JS: La herramienta que revolucionó el front-end. [En línea] 2009. sg.com.mx.

7. **I Arzuza, M Hernández, F Ortiz, C Rodríguez.** Symfony. [En línea] unisimonbolivar.edu.co.

8. **S Pérez Lovelle, F Orejas Valdés.** Revista Ingeniería. [En línea] 2010 . rii.cujae.edu.cu.

9. **SL Mora, SL Mora, MZ Fornieles, SL Mora.** El futuro de Internet:'XM. *Los secretos xml.* [En línea] dlsi.ua.es.

10. **GB Hall, JP Alperin, SK LEÓN.** GeoFocus. [En línea] 2008 . geofocus.rediris.es.

11. PgAdmin_III. [En línea] guia-ubuntu.org.

12. **Boudreau, Tim.** NetBeans: the definitive guide. [En línea] 2002. books.google.com.

13. **Navarro, Gonzalo.** Cómo funciona la Web. [En línea] ciw.cl.

14. **Heffelfinger, David R.** JasperReports 3.5 for Java Developers. [En línea] Agosto de 2009. www.packtpub.com. 1050809.

15. **Jacobson, Ivar, Grady, Booch y Rumbaugh, James.** El proceso unificado de desarrollo de software. [En línea] 2000. en.scientificcommons.org.

16. **Kernighan, Brian W, Ritchie, Dennis M. y Gómez Muñoz, Néstor.** Lenguajes de programación . [En línea] 1991. books.google.com.

17.**Tecnología y Synergix**. [En línea] 2009. [Citado el: 2012 de Febrero de 16.] http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/.

18. **Pressman, Roger S**.; 2007. *"Ingeniería de software. Un enfoque práctico"*. 6ta Edición. Parte I. Prólogo y Capítulos 1, 2 y 21. Páginas 1-18, 22-45 y 641-657.

20. **Lobo, Armando Robert.** *Arquitectura de Software para el Sistema Integrado de Gestión Estadística 2.0 Nuragas.* Ciudad de la Habana: s.n., 2009.

21. **Hernández, Pedro Veloso.** *Uso de patrones de arquitectura.*

22. **Díaz Anton, Maria Gabriela, y otros.** academia-interactiva.com. *IV Congreso .* [En línea] 2003. [Citado el: 5 de 9 de 2011.] academia-interactiva.com.

23. **Bascón Pantoja, Ernesto.** Revista Acta Nova. [En línea] 2011. ucbconocimiento.ucbcba.edu.bo.

24. **JACOBSON, I.; G. BOOCH**, et al. El proceso Unificado de Desarrollo de Software. p.xviii Addison Wesley Object Technology.

25. **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* España : Adison-Wesley.

26. **JACOBSON, G. B. J. R. I.** *El Proceso Unificado del Desarrollo de Software.* 2000.

27. modelo-de-datos. [En línea] 26 de 08 de 2004. [Citado el: 18 de 3 de 2012.] mundogeek.net.

28. **JACOBSON, I.; G. BOOCH**, et al. El proceso Unificado de Desarrollo de Software. p.xviii Addison Wesley Object Technology.

29. **Martín, Mariana.** fec.uh.cu. [En línea] 2008. [Citado el: 1 de 03 de 2012.]

30. **Gong, Li, Mueller, Marianne y Prafullchandra, Hemma.** Proceedings of the. [En línea] 1997. [Citado el: 23 de 04 de 2012.] static.usenix.org.

31. **Reese, George.** Database Programming with JDBC and Java. [En línea] 2000. [Citado el: 13 de 02 de 2012.] dl.acm.org.se Programming with JDBC and Java. [En línea] 2000. dl.acm.org.

32. **Herwijnen, E Van.** [En línea] 1994. [Citado el: 5 de 10 de 2012.] books.google.com.

BIBLIOGRAFÍA

- **JACOBSON, I.; G. BOOCH**, et al. *El proceso Unificado de Desarrollo de Software*. p.xviii Addison Wesley Object Technology.
- **LOBO, Armando Robert.** *Arquitectura de Software para el Sistema Integrado de Gestión Estadística 2.0 Nuragas*. Ciudad Habana: Trabajo de Diploma, 2009.
- **PRESMAN, Roger S**.; 2007. *Ingeniería de software. Un enfoque práctico*. 6ta Edición. Parte I. Prólogo y Capítulos 1, 2 y 21. Páginas 1-18, 22-45 y 641-657.
- **WILLIAMS, Ian.** *Beginning xsl and xpath.* Indianapolis,Indiana : Wiley Publishing, Inc, 2009. 978-0-470-47725-0.
- **DUCHARME, Bob.** *XSLT Quickly.* s.l. : Manning Publications Co, 2001. 1-930110-11-1.
- **MANGANO, Sal.** *XSLT Cookbook, 2nd Edition.* s.l. : O'Reilly, 2005. 0-596-00974-7.
- **ORTIZ, Kadir Hector.** http://www.eumed.net/libros/2009c/583/Representacion%20del%20Modelo%20de%20Objetos%20de%20Dominio.html. [En línea] [Citado el: 15 de enero de 2011.]
- **POTENCIER, Fabien y François Zaninotto.** *Symfony La Guía Definitiva.* 30 de diciembre de 2008.
- **Sanchez, Linet Lores y Roque, Diana Monné.** *Aplicación de las pruebas de liberación al Sistema Informático de Genética Médica.* Ciudad de la Habana, Junio 2009. Trabajo de Diploma para optar por el título de Ingeniero Informático.
- **TELLO, Jesús Cáceres.** *Diagramas de Casos de Uso*. Universidad de Alcalá Departamento de Ciencias de Computación: s.n.
- **DANCIU**, Teodor. 2002. The Jasper Reports Ultimade Guide. 2002.
- Comunity, OpenUp. Open UP. [En línea] EPF Copyrigth. http://epf.eclipse.org/wikis/opneup.
- Diseño, Patrones de. 2001. Patrones de Diseño. [En línea] 2001. http://mit.ocw.universia.net/6.170/6.170/f01/pdf/lecture-12.pdf.
- ExtJS. 2009. API Documentation. [En línea] 2009. http://extjs.com/deploy/dev/docs.
- **HEFFELFINGER**, David R. Agosto 2009. Jasper Reports for Java Developers. Agosto 2009.
- **LARMAN**, Craig. 1999. UML y Patrones. México : Dawn Speth White, 1999. Potencier, Fabien y **ZANINOTTO**, Francois. 2009. Symfony. La guia definitiva. [En línea] 2009. http://www.symfony-project.org.
- Propel. 2009. Propel project. [En línea] 2009. http://propel.phpdb.org/trac.
- Sitio oficial del Visual Paradigm. Disponible en: http://www.visual-paradigm.com/product/vpuml. (21/01/2008).

- **Kernighan, Brian W, Ritchie, Dennis M. y Gómez Muñoz, Néstor.** Lenguajes de programación . [En línea] 1991. books.google.com.
- **Tecnología y Synergix**. [En línea] 2009.http://synergix.wordpress.com/2008/07/07/requisito-funcional-y-no-funcional/.
- **Hernández, Pedro Veloso.** *Uso de patrones de arquitectura.*
- **Díaz Anton, Maria Gabriela, y otros.** academia-interactiva.com. *IV Congreso .* [En línea] 2003. academia-interactiva.com.
- **Bascón Pantoja, Ernesto.** Revista Acta Nova. [En línea] 2011. ucbconocimiento.ucbcba.edu.bo.
- **JACOBSON, I.; G. BOOCH**, et al. El proceso Unificado de Desarrollo de Software. p.xviii Addison Wesley Object Technology.
- **Jacobson, Ivar, Booch, Grady y Rumbaugh, James.** *El Proceso Unificado de Desarrollo de Software.* España : Adison-Wesley.
- **JACOBSON, G. B. J. R. I.** *El Proceso Unificado del Desarrollo de Software.* 2000.
- modelo-de-datos. [En línea] 26 de 08 de 2004.mundogeek.net.
- 28. **JACOBSON, I.; G. BOOCH**, et al. El proceso Unificado de Desarrollo de Software. p.xviii Addison Wesley Object Technology.
- **Martín, Mariana.** fec.uh.cu. [En línea] 2008.
- **Gong, Li, Mueller, Marianne y Prafullchandra, Hemma.** Proceedings of the. [En línea] 1997. [Citado el: 23 de 04 de 2012.] static.usenix.org.
- **Reese, George.** Database Programming with JDBC and Java. [En línea] 2000. [Citado el: 13 de 02 de 2012.] dl.acm.org.se Programming with JDBC and Java. [En línea] 2000. dl.acm.org.
- **Herwijnen, E Van.** [En línea] 1994. [Citado el: 5 de 10 de 2012.] books.google.com.
- **Kernighan, Brian W, Ritchie, Dennis M. y Gómez Muñoz, Néstor.** Lenguajes de programación . [En línea] 1991. books.google.com.
- **Tecnología y Synergix**. [En línea] 2009. [Citado el: 2012 de Febrero de 16.]
- **GB Hall, JP Alperin, SK LEÓN.** GeoFocus. [En línea] 2008 . geofocus.rediris.es.
- PgAdmin_III. [En línea] guia-ubuntu.org.
- **I Arzuza, M Hernández, F Ortiz, C Rodríguez.** Symfony. [En línea] unisimonbolivar.edu.co.
- **S Pérez Lovelle, F Orejas Valdés.** Revista Ingeniería. [En línea] 2010 . rii.cujae.edu.cu.
- **SL Mora, SL Mora, MZ Fornieles, SL Mora.** El futuro de Internet:'XM. *Los secretos xml.* [En línea] dlsi.ua.es.

ANEXOS

Figura 14 : Diagrama de Clases del Diseño del Caso de Uso Administrar Reportes.

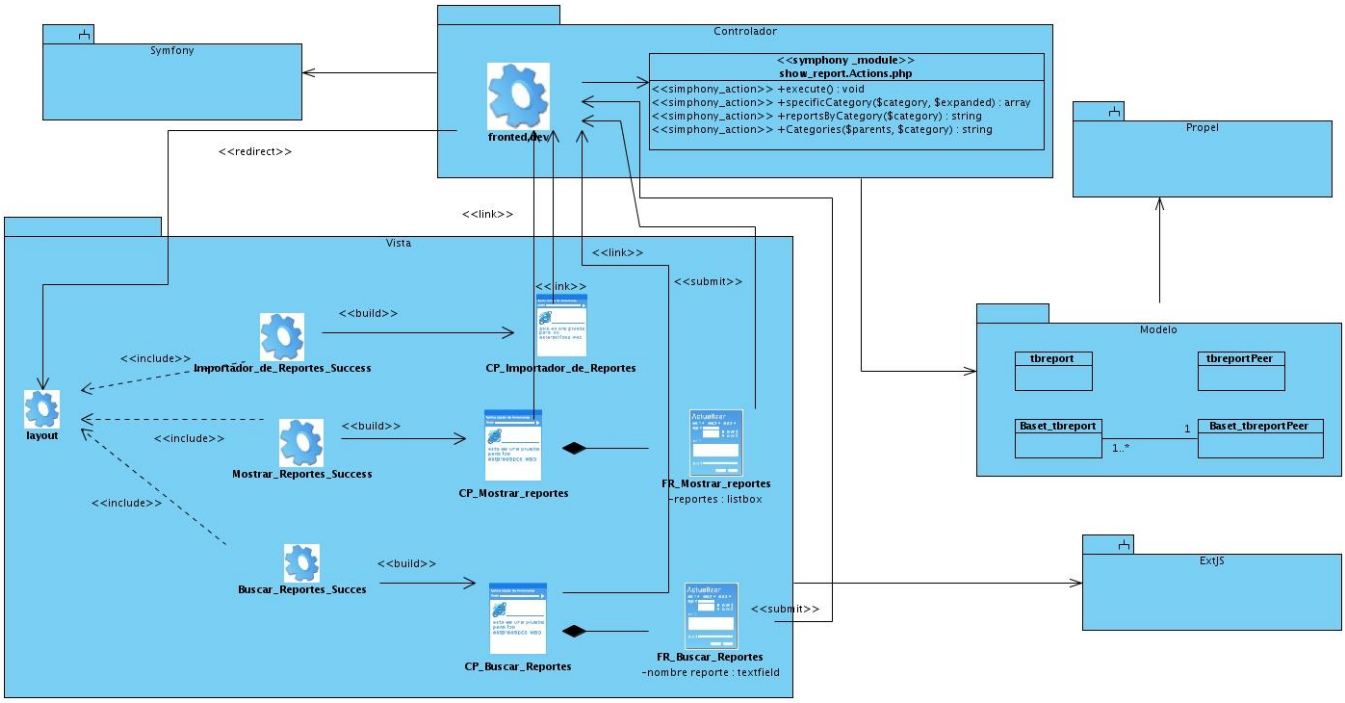


Figura 15 : Diagrama de Clases del Diseño del Caso de Uso Importar Modelos de Datos.

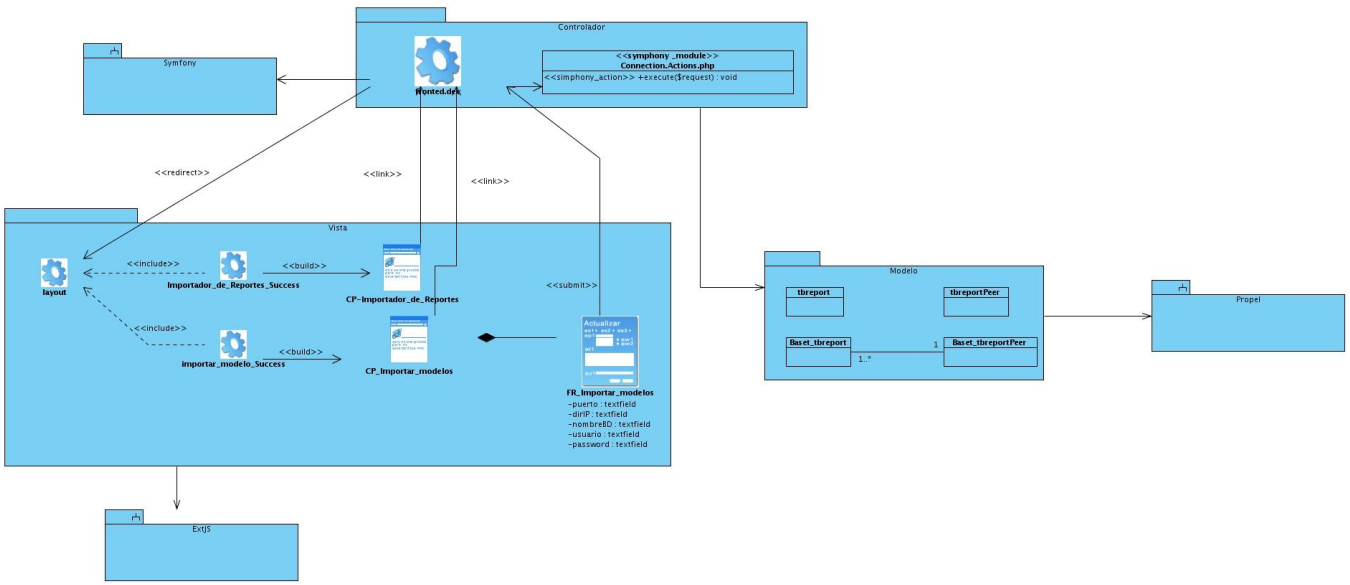


Figura 16: Diagrama de Secuencias del Caso de Uso Administrar Reportes Escenario-Mostrar Reportes.

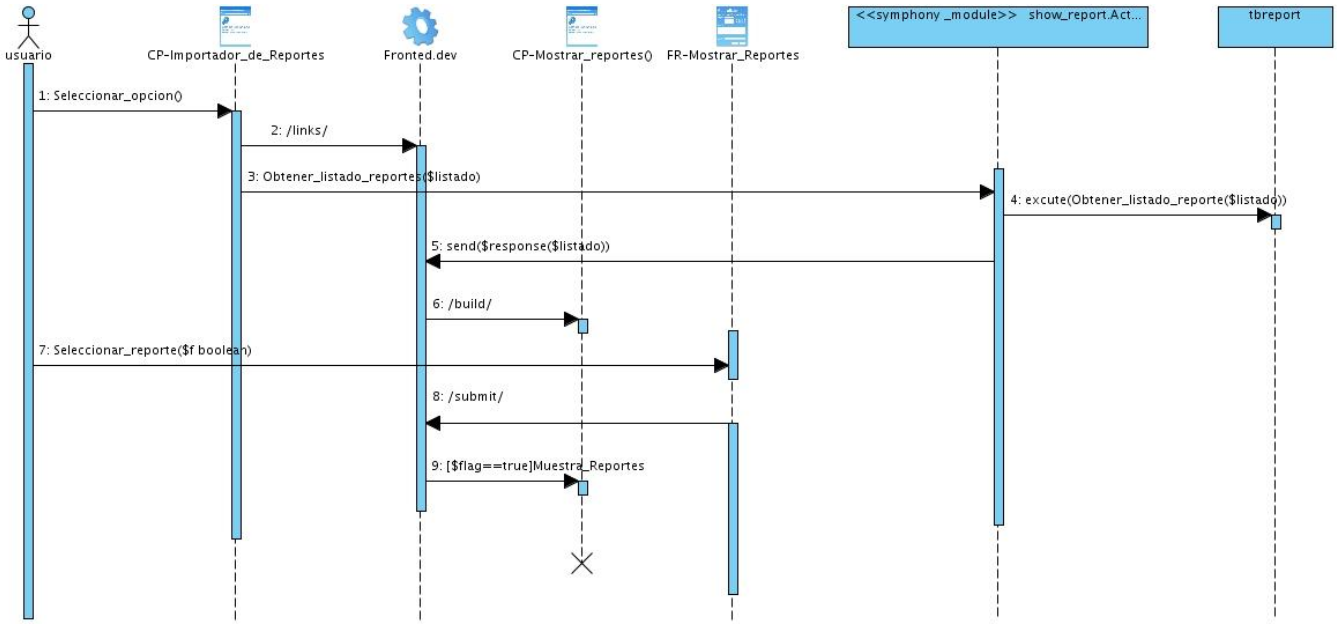


Figura 17: Diagrama de Secuencias del Caso de Uso Administrar Reportes Escenario Buscar Reportes.

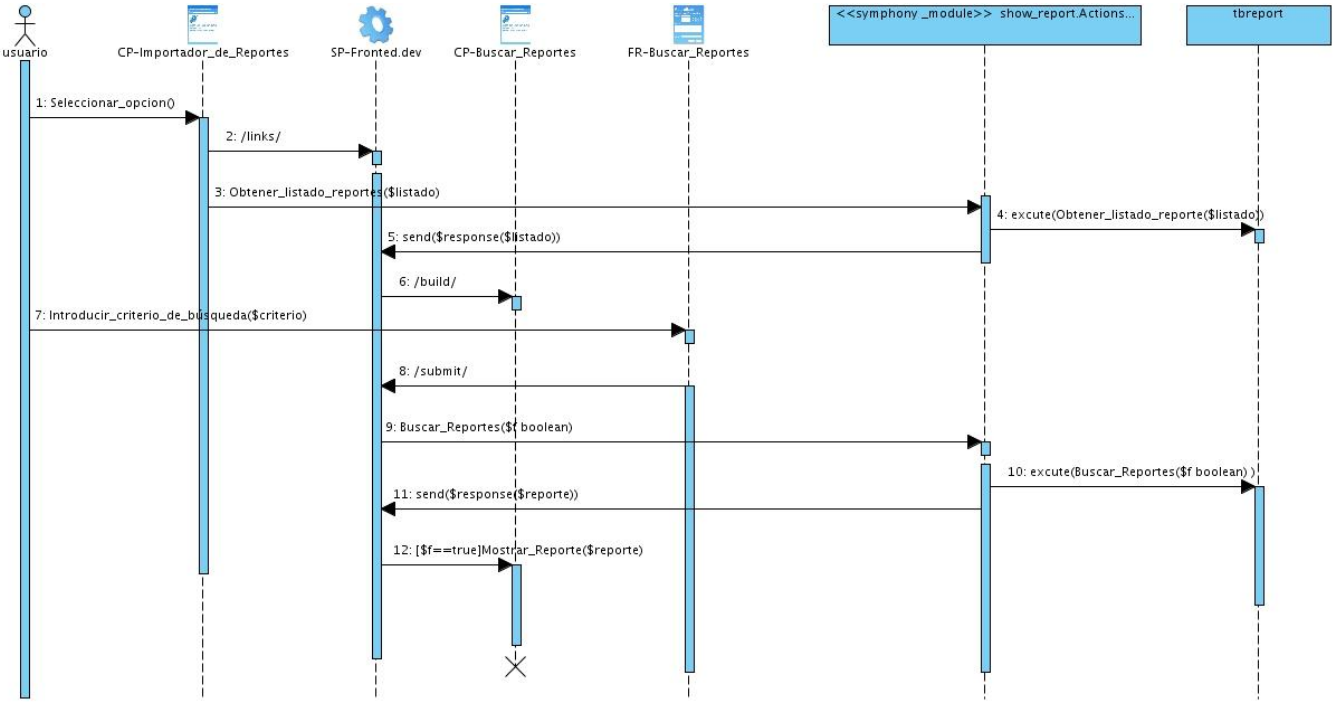


Figura 18: Diagrama de Secuencias del Caso de Uso Importar Modelos de Datos.

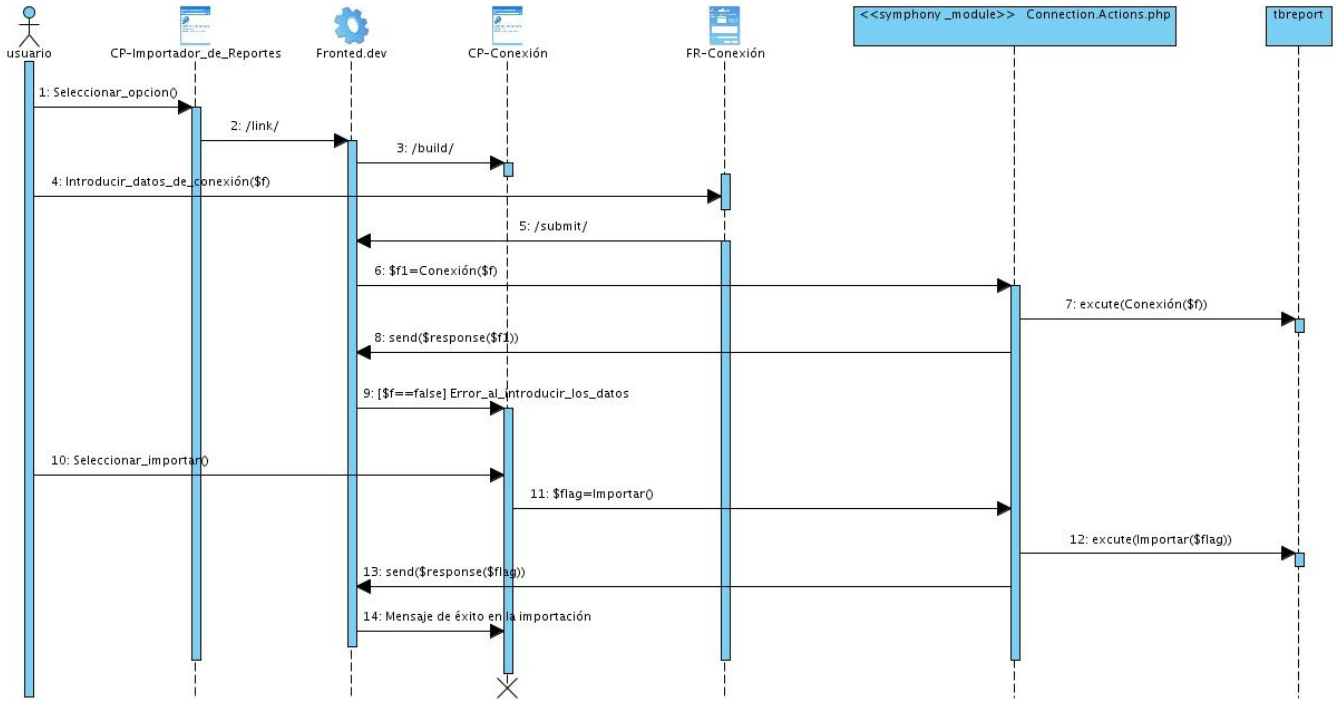


Figura 19: Diagrama de Componentes del la Capa Vista.

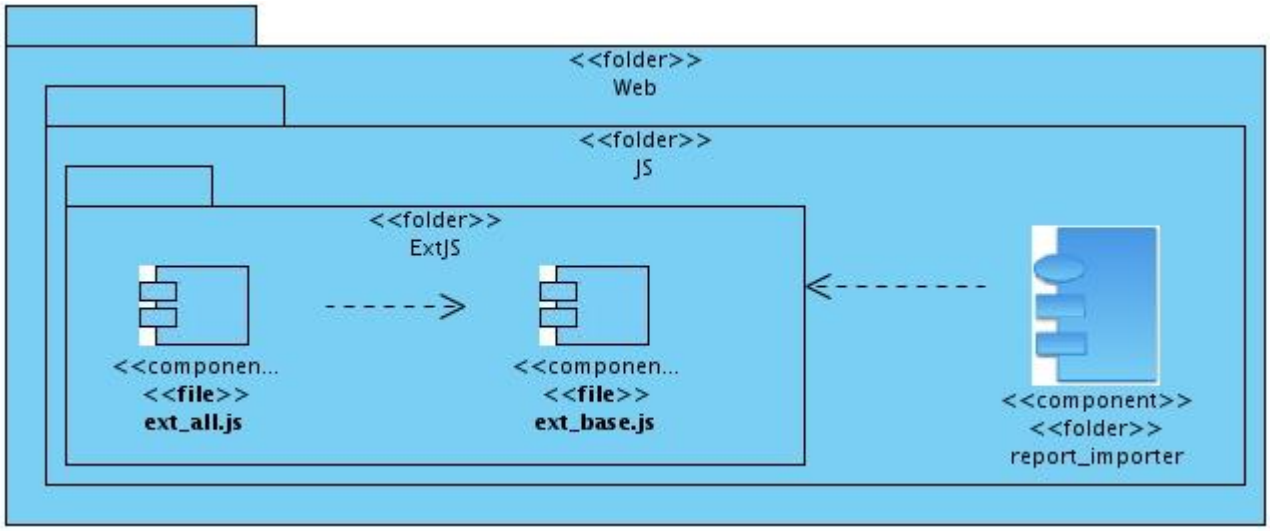


Figura 20: Diagrama de Componentes del Caso de Uso Administrar Reportes.

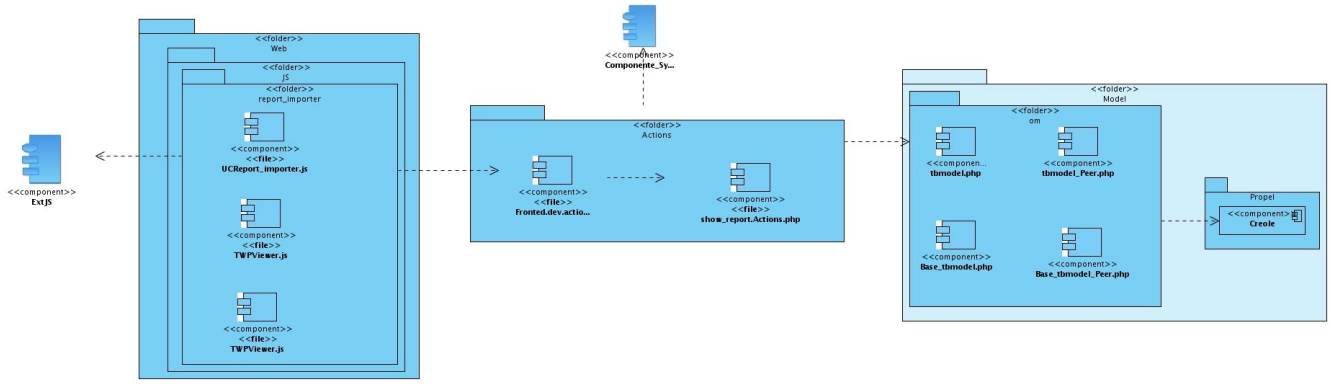
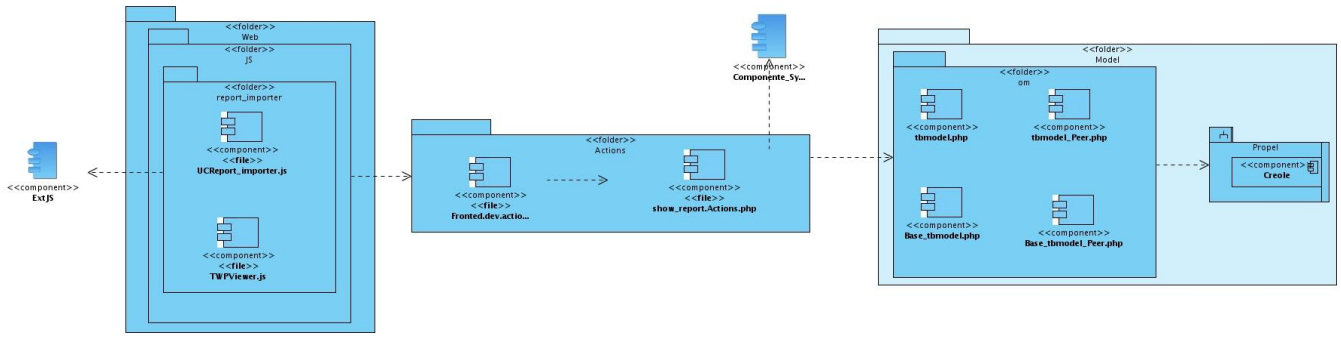


Figura 21 : Diagrama de Componentes del Caso de Uso Importar Modelos de Datos.



GLOSARIO DE TÉRMINOS

BD: Base de Datos.

CU: Caso de Uso.

Framework: Marco de trabajo.

GDR: Generador dinámico de reportes.

IDE: Entorno desarrollo integrado**.**

OpenUp: Metodología para el proceso del desarrollo del software.

UML: Unified modeling language(Lenguaje Unificado de Modelado).

ORM: Mapeo de Objetos a Bases de Datos.

CASE: Ingeniería de Software Asistida por Computación.

GOF: Gang of Four.

GRASP: General Responsability Assignment Software Patterns (Patrones de Software para la asignación General de Responsabilidades)

SOAP:. Simple Object Access Protocol

MVC: Modelo – Vista – Controlador.

GPL: Licencias Open Source.

UI: Interfaces de usuario.