Universidad de las Ciencias Informáticas

Facultad 6

Título: Módulo Visor de Reportes para el Generador Dinámico de Reportes v2.0

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

Autora:

Lianet Cruz González

Tutores:

Ing. Vania Elena Yanes León

Ing. Aurelio Rodríguez Durán

Ing. Miguel Lezcano Ramos

La Habana, Junio 2012
"Año 54 de la Revolución"

"El genio es uno por ciento de inspiración y un noventa y nueve por ciento de dedicación."

Thomas Alva Edison

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 _____.

Lianet Cruz González	Ing. Vania Elena Yanes León
_____	_____
Firma del Autor	Firma del Tutor
Ing. Aurelio Rodríguez Durán	Ing. Miguel Lezcano Ramos
_____	_____
Firma del Autor	Firma del Autor

DATOS DE CONTACTO

Tutor: Ing. Vania Elena Yanes León

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: veyanes@uci.cu

Tutor: Ing. Aurelio Rodríguez Durán

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: arduran@uci.cu

Tutor: Ing. Miguel Lezcano Ramos

Universidad de las Ciencias Informáticas, La Habana, Cuba.

Email: mlezcano@uci.cu

AGRADECIMIENTOS

A mis padres Osmara y Lázaro*, quienes sin escatimar esfuerzo alguno han sacrificado gran parte de su vida para formarme y educarme, a ellos les debo todo lo que soy. Gracias por tanto amor, por sus consejos, su apoyo incondicional, por los valores que me enseñaron desde niña y por ser un ejemplo para mí. Mami te agradezco por ser única, comprensiva y darme lo mejor de ti. Papi gracias por tu paciencia, por ser el mejor papá del mundo y por estar siempre para mí. Los quiero mucho.*

A mi hermano *por ser tan especial para mí. Gracias, se que siempre puedo contar contigo y puedes tener la seguridad de que siempre estaré presente para ti. Te quiero mucho y estoy muy orgullosa de ti aunque nunca te lo diga.*

A mis abuelas, abuelos, tías, tíos, primas, primos*, (aunque algunos no estén hoy físicamente), en fin a toda mi linda familia por su amor, por creer siempre en mí y estar seguros de que este momento llegaría. Siempre los llevo en mi corazón.*

A mis amigas*, Aryanna, Rosaida, Martha, Wendy, la nena, Dany por los buenos momentos que pasamos juntas y los recuerdos que quedarán grabados para siempre en mi memoria, en especial a mi amiga Adrialis que más que una amiga es una hermana para mí. Gracias chicas por todo su apoyo, por aceptarme como soy y por estar conmigo en las buenas y en las malas.*

A mis tutores Vania, Aurelio y Miguel, *por su paciencia, comprensión y por estar siempre dispuestos a ayudarme. Al tribunal y a la oponente por sus reconstructivas críticas.*

A mis amistades de la UCI *que han sido una familia para mi y a todas aquellas personas que no dudaron ni un minuto en brindarme su ayuda incondicional, en especial a Garnache, Adnan, Victor, Lianet Salazar, Wilber, Claudia, Julio, Marcos, Yoandy, Yanier, Marla, Miguel y Galiano. Gracias y éxitos para todos.*

RESUMEN

Una manera de gestionar la información es a través de los reportes, estos son de gran utilidad para el análisis de resultados y brindan una mayor eficacia en el proceso de toma de decisiones. Por tal motivo en la actualidad existen herramientas que facilitan su proceso de generación las cuales son conocidas por el nombre de generadores de reportes.

En la Universidad de las Ciencias Informáticas se ha desarrollado el Generador Dinámico de Reportes (GDR), el cual garantiza la creación de reportes a partir de información almacenada en bases de datos. El presente trabajo de diploma está orientado al análisis, diseño e implementación del módulo Visor de Reportes para la versión 2.0 del GDR con el fin de mejorar las potencialidades de visualización de los reportes. Para ello se realizó un estudio de las metodologías, tecnologías y herramientas quedando seleccionada como metodología de desarrollo de software OpenUp, PHP como lenguaje de programación del lado del servidor y JavaScript del lado del cliente, utilizando como apoyo los frameworks Symfony y ExtJs. Para el modelado se utilizó Visual Paradigm y como entorno de integrado desarrollo NetBeans. Se realizaron pruebas que validan los resultados de la solución y verifican que el sistema posee la calidad requerida.

Palabras claves: reportes, generadores de reportes, Visor de reportes, Generador Dinámico de Reportes.

TABLA DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

La información es un elemento fundamental para el desarrollo de la sociedad. Se puede decir que la vida gira en torno a ella, pues toda actividad humana involucra de alguna manera su uso. La misma es considerada por las empresas como uno de sus principales recursos y como un factor crítico para la determinación del éxito o fracaso del negocio, por eso se hace cada vez más necesaria su correcta gestión. La gestión de la información es el proceso que se encarga de suministrar los recursos necesarios para la toma de decisiones, así como para mejorar los procesos, productos y servicios de una organización. (1)

Actualmente, debido al notable crecimiento del flujo de la información, constituye un desafío para las empresas e instituciones el proveer a las personas los datos necesarios en el momento adecuado. En este sentido juegan un importante papel los reportes, que son objetos que aplican un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y fácil de interpretar por los usuarios y que permiten realizar ciertas operaciones.

Es necesario medir todos los aspectos de las empresas ya que lo que no se mide no se controla y por lo tanto no se puede mejorar. Atendiendo a esto, los reportes también son utilizados por la mayoría de las organizaciones para visualizar y analizar los resultados, de esta forma es más cómodo para los responsables observar la marcha del negocio con respecto a las metas propuestas. Generar reportes ante el gran volumen de datos y la considerable variedad de fuentes de las que proviene, muestra gran complejidad cuando se realiza de forma manual, por esta razón se han desarrollado en el mundo diferentes herramientas, tanto libres como propietarias, que facilitan su proceso de generación. Dichas herramientas son llamadas generadores de reportes, los cuales tienen definido un mecanismo para ejecutar consultas a las bases de datos y adquirir información de ellas en forma de reporte. Con ellas es posible tener un mayor control sobre el diseño y la manera en que la información será visualizada.

Los generadores de reportes sirven de apoyo a los sistemas de información posibilitando la materialización de la gestión de la información de una manera relativamente sencilla. Algunos de los más utilizados mundialmente son: Active Reports, Exchange Reporter Plus, Jasper Reports, Agata Reports y Crystal Reports, estos sólo cubren parte del ciclo de vida de los reportes. Existen otros que son herramientas privativas y que poseen un costo muy elevado, como es el caso de Microsoft SQL Server 2005 Reporting Services.

Cuba no se encuentra ajena al desarrollo de la industria del software, como parte de uno de los programas de la Revolución cubana surge en el año 2002 la Universidad de las Ciencias Informáticas

(UCI). La misma tiene como uno de sus objetivos, crear productos, servicios informáticos y soluciones tecnológicas integrales, tanto para la informatización nacional como para la exportación. Para ello cuenta con varios proyectos productivos agrupados de acuerdo a temáticas afines en diferentes centros de desarrollo. Uno de estos centros es el Centro de Tecnología de Gestión de Datos (DATEC), que tiene como misión proveer soluciones integrales, productos y servicios relacionados con las tecnologías de Gestión de Datos y Bioinformática, permitiendo la formación de profesionales integrales con un alto nivel científico- productivo y el desarrollo de investigaciones afines. (2) Este está formado por cuatro departamentos: Almacén de Datos, PostgreSQL, Bioinformática e Integración de Soluciones. En este último se ha desarrollado el sistema Generador Dinámico de Reportes (GDR) con el fin de formular reportes a partir de información contenida en las bases de datos para brindar así una mayor eficacia en el proceso de toma de decisiones.

El sistema está compuesto por un conjunto de cinco módulos que brindan funcionalidades para el trabajo con los reportes ofreciéndoles soporte durante todo el ciclo de vida. Dentro de los módulos que contiene se encuentra el Visor de Reportes el cual es utilizado frecuentemente por los usuarios finales.

La versión actual del sistema GDR (1.7), basada en el motor de reportes PHPReports, carece de potencialidades que están presentes en otras herramientas de su tipo. Específicamente el módulo Visor de Reportes presenta las siguientes deficiencias:

- Limitados formatos de salida, lo cual implica pocas posibilidades de reutilización del informe.
- No permite rango de visualización por páginas, dificultando la navegación en el reporte.
- Ausencia de opciones para la visualización de los diferentes formatos soportados, lo cual no permite enriquecer la visualización del informe.
- No permite visualización de informes que contengan un conjunto de reportes.

Teniendo en cuenta lo antes expresado se define el siguiente **problema de la investigación:**
¿Cómo visualizar los reportes para el Generador Dinámico de Reportes v2.0?

La investigación tiene como **objeto de estudio:** Proceso de desarrollo de software de los sistemas generadores de reportes.

Enmarcado en el **campo de acción:** Visor de Reportes para el Generador Dinámico de Reportes.

Para darle solución al problema planteado, se define como **objetivo general:** Desarrollar el módulo Visor de Reportes del Generador Dinámico de Reportes v2.0 para mejorar las potencialidades de visualización de los reportes.

En correspondencia con ello, se plantean como **objetivos específicos:**

- Realizar el análisis y diseño del Módulo Visor de Reportes para el Generador Dinámico de Reportes v2.0.
- Implementar el Módulo Visor de Reportes para el Generador Dinámico de Reportes v2.0.
- Validar el Módulo Visor de Reportes para el Generador Dinámico de Reportes v2.0.

Con vista a dar cumplimiento a los objetivos planteados, se definen las siguientes **tareas de la investigación**:

- Análisis de los conceptos básicos y técnicos implicados en el desarrollo del módulo Visor de Reportes.
- Selección de las herramientas y metodología a utilizar.
- Definición de los requisitos funcionales y no funcionales para el correcto funcionamiento del módulo.
- Selección de patrones de diseño a emplear en el desarrollo del módulo Visor de Reportes.
- Diseño del módulo Visor de Reportes.
- Implementar la interfaz gráfica del módulo Visor de Reportes.
- Implementar los algoritmos internos y las funciones de procesamiento del módulo Visor de Reportes.
- Diseño de los casos de prueba.
- Ejecución de las pruebas.

El presente trabajo de diploma ha sido estructurado de la siguiente forma: resumen, introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía, anexos.

Capítulo 1.Fundamento Teórico: Se presentan los conceptos fundamentales relacionados con el objeto de estudio. Además se abordan las características de la metodología de desarrollo de software, herramientas y tecnologías que se utilizarán para dar solución al problema planteado.

Capítulo 2.Análisis y Diseño de la Solución: En este capítulo se aborda acerca de cómo debe funcionar el sistema, haciendo una descripción general del funcionamiento del mismo. Se elabora una lista de requerimientos funcionales y no funcionales que se deben tener en cuenta para la realización del sistema. Se presenta una vista abstracta del diseño del sistema.

Capítulo 3. Implementación y Prueba: Se describen implementaciones relevantes y se visualizan algunas pantallas del sistema informático. Además se realiza la validación del sistema a través de

pruebas funcionales.

CAPÍTULO 1: FUNDAMENTO TEÓRICO

1. Introducción

En el presente capítulo se plantean los principales conceptos que se relacionan con el objeto de estudio y que son necesarios para entender posteriormente la propuesta de solución. Se estudian las características de las tecnologías, metodologías de desarrollo, lenguajes de programación y herramientas existentes, haciendo una selección de aquellas que serán utilizadas para dar solución al problema científico.

1.1. Conceptos fundamentales relacionados con el dominio del problema

La generación de reportes es el proceso de obtención de reportes de manera personalizada, basado en tablas o consultas. Su objetivo principal es sin duda proporcionar información útil y de una manera más rápida al personal al cual esté dirigido. En este proceso intervienen los generadores de reportes.

1.1.1. Reporte

Un reporte es un informe o una noticia. Este tipo de documento (que puede ser impreso, digital o audiovisual) pretende transmitir una información, aunque puede tener diversos objetivos. Generalmente agrupan información de acuerdo a un interés específico, muestran el resultado de una búsqueda determinada. En la esfera de la informática, los reportes son informes que organizan y exhiben la información contenida en una base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. (3)

El reporte es aquel documento que se utilizará cuando se quiera informar o dar noticia acerca de una determinada cuestión**.** También podrá incluir algunos elementos persuasivos, como hacer recomendaciones o sugerencias y también algunas conclusiones a través de las cuales se le indique al lector del mismo alguna acción o conducta a adoptar en el futuro**.** (4)

1.1.2. Generadores de reportes

Generadores de reportes: son programas para crear informes sobre diseño en una amplia variedad de formatos que no son rutinariamente producidos por un sistema de información. Extraen datos de los archivos o de las bases de datos y crean reportes de acuerdo con muchos formatos, proporcionan más control, pueden manejar datos de cálculos y lógica compleja antes de darles la salida. (5)

Los sistemas generadores de reportes complementan a los sistemas de información, estos permiten

obtener la información en forma de reporte, ofreciendo a los trabajadores calificados un mayor nivel de detalle y flexibilidad, además de la capacidad de interactuar con los resultados obtenidos basándose en los datos para tomar sus propias decisiones. (6)

Los generadores de reportes son programas diseñados para crear y administrar informes en una amplia variedad de formatos. Estos proporcionan más control, pueden manejar datos de cálculos y lógica compleja antes de darles la salida. Se caracterizan por dos elementos básicos: un diseñador o editor de informes y un motor de generación. (7)

A continuación se mencionan y caracterizan algunos de los motores de reportes reconocidos internacionalmente.

1.1.3. PHPReports

PHPReports es un motor de generación de reportes para PHP, es software libre y está bajo licencia GPL. A pesar de ser multiplataforma, este software funciona mejor con el uso de GNU / Linux como sistema operativo y Apache como servidor web. Su funcionamiento se basa en transformaciones XSL (siglas de Extensible Stylesheet Language) conocidas como XSLT a partir de la clase PHPReportsMaker. Los reportes en PHPReports están divididos en capas que contienen unas a otras (ver Figura 1), entre ellas se encuentran la capa Documento y la capa Página, cada una tiene su propio encabezado y pie de página. Por último está la capa Grupo en la que se cargarán los campos obtenidos de la consulta a la base de datos. (8)

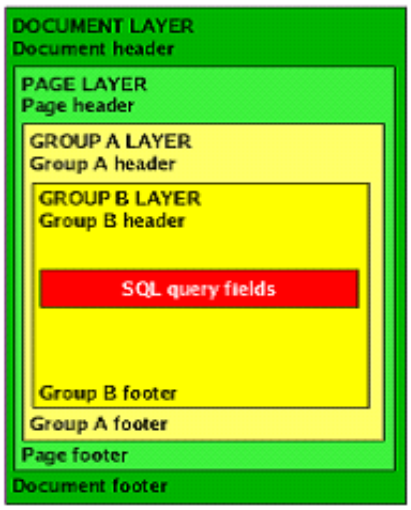


Figura 1: Capas de un reporte en PHPReports.

1.1.4. JasperReports

Es una librería de clases de Java de código abierto desarrollada por Teodor Danciu en el 2001. Su propósito principal es crear reportes de una manera simple, flexible y dinámica. Este potente motor para la generación de reportes tiene la habilidad de crear informes para ser visualizados en la web o enviados a la impresora. Los reportes pueden ser exportados a una multitud de formatos como PDF, XLS, RTF, HTML, XML, CVS y texto plano. Además de los datos en texto, JasperReports permite incluir en los reportes imágenes y gráficos para que los mismos tengan un aspecto profesional. Esta herramienta posee un amplio y expandible grupo de posibles fuentes de datos.

Algunas de las características más significativas que provee JasperReports 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.
- Permite que los desarrolladores le surtan datos en varias formas: lo cual significa que los desarrolladores pueden pasar datos a los reportes a través de parámetros.
- Pueden generar sub-reportes: JasperReports permite la creación de reportes dentro de reportes lo que facilita bastante el diseño porque es posible usar estos sub-reportes en otros reportes.
- Los reportes son capaces de presentar los datos de manera textual o a través de gráficos: no sólo son capaces de mostrar los datos que le son pasados sino que pueden generar o calcular nuevos datos de forma dinámica y mostrarlos.
- Pueden generar marcas de agua: JasperReports 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.
- El software es libre pues es distribuido mundialmente bajo los términos de la Licencia Pública para Librerías GNU (GNU Library Public License) y está respaldado por una gran comunidad internacional de desarrollo, el proyecto JasperForge.org y la empresa JasperSoft Corporation.
- Puede ser utilizado en cualquier entorno o sistema operativo siempre que exista una implementación de la máquina virtual de Java para dicho entorno. (9)

1.1.5. Active Reports

Active Report es una herramienta de plataforma propietaria que permite diseñar y crear reportes de forma fácil y rápida. Se caracteriza por dar soporte tanto a aplicaciones web como de escritorio siendo capaz de emplear una amplia variedad de orígenes de datos. Además soporta gráficos, imágenes, sub-reportes y permite la exportación de los reportes a los formatos PDF, Excel, RTF, TIFF, XML y HTML. Active Reports incluye un control de visor de informes que admite zoom y vista previa de informes, múltiples fichas para visualización de hipervínculos, vistas divididas y de múltiples páginas, un panel de índice de contenidos, miniaturas, búsquedas de texto, anotaciones y personalización de barra de herramientas. Se distribuye bajo licencia privativa perteneciente a GrapeCity, inc.

Sistema operativo para despliegue: Windows 7, Windows Server 2008, Windows Vista, Windows XP, Windows Server 2003. (10)

1.1.6. Microsoft SQL Server 2005 Reporting Services

Microsoft SQL Server 2005 Reporting Services es una solución basada en servidor que se utiliza para generar informes empresariales que extraen contenido de una variedad de orígenes de datos relacionales y multidimensionales, que publica informes que se pueden ver en diversos formatos, y que administra la seguridad y las suscripciones de manera centralizada. Los informes creados se pueden visualizar y administrar mediante una conexión basada en Web o como parte de una aplicación de Microsoft Windows o un portal de SharePoint. Puede crear informes tabulares, matriciales o de formato libre. También puede crear informes adaptados a una circunstancia determinada, que utilicen modelos y orígenes de datos predefinidos. (11)

Reporting Services contiene los componentes principales siguientes:

- Un conjunto completo de herramientas que se pueden utilizar para crear, administrar y ver informes.
- Un componente Servidor de informes que aloja y procesa informes en diversos formatos. Los formatos de salida incluyen HTML, PDF, TIFF, Excel, CSV, etc.
- Una API (Application Programming Interface) que permite a los programadores integrar o extender procesamientos de datos e informes en aplicaciones personalizadas o crear herramientas personalizadas para generar y administrar informes.
- Sistema Operativo: Microsoft Windows, preferiblemente en sus versiones para servidores.
- Licencia: se distribuye bajo licencia privativa perteneciente a Microsoft Corporation, dicha licencia forma parte de la licencia de Microsoft SQL Server 2000 ó 2005 (Microsoft Corporation). (12)

1.1.7. Selección del motor de reportes a utilizar

Para la selección del motor a utilizar en esta nueva versión del generador dinámico de reportes se tuvieron en cuenta las características del mismo en relación con las necesidades de los clientes. A continuación se ofrece una tabla comparativa (ver Tabla 1) resumiendo los aspectos fundamentales de los generadores de reportes estudiados, sin mencionar el PHPReports, ya que el GDR actualmente está basado en dicho motor y se pretende sustituirlo.

Tabla 1: Comparación entre generadores de reportes.

Generador de reportes	Libre	Sistema Operativo	Formatos de salida
JasperReports	Si	Multiplataforma	PDF, HTML, XLS, Excel, CSV, ODT, RTF y XML
Active Reports	No	Windows 7, Windows Server 2008, Windows Vista, Windows XP, Windows Server 2003.	PDF, Excel, RTF, HTML, texto y en formato TIFF.
Microsoft SQL Server 2005 Reporting Services	No	Microsoft Windows, preferiblemente en sus versiones para servidores.	.HTML, PDF, TIFF, Excel, CSV.

Una vez analizadas las características de dichas tecnologías de reportes se decide escoger como motor a utilizar en esta nueva versión del GDR a JasperReports por ser software libre y por sus ventajas en cuanto a la amplia variedad de formatos de salidas que ofrece, propiedades de los reportes y fuentes de datos que soporta. Además, es una herramienta multiplataforma que se distribuye bajo licencia pública y que maneja los sub-reportes. Las funcionalidades que brinda dicho motor responden en mayor medida a las deficiencias actuales del GDR.

1.2. Lenguajes de programación

Se define como lenguaje de programación al elemento dentro de la informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis; que se pone a disposición del programador para que este pueda comunicarse con los dispositivos hardware y software existentes. (13)

Cada lenguaje de programación tiene sus características propias, las cuales lo hacen más potente o más débil para determinada finalidad. Actualmente existe un gran número de lenguajes de programación web. El estudio de los mismos garantizará contar con los recursos indicados para codificar la solución a desarrollar.

1.2.1. PHP5 5.3

PHP, que significa Hypertext Pre-Preprocessor (inicialmente llamado Personal Home Page), es un lenguaje script diseñado para el desarrollo de páginas web dinámicas del lado del servidor. Sus fragmentos de código se intercalan fácilmente en páginas HTML, debido a esto, y a que es de código abierto, es muy popular y extendido en la web.

PHP se integra muy bien junto a otro software, especialmente bajo ambientes Unix. Utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables, conformando sistemas robustos y estables. Dispone de una amplia gama de librerías. (14)

Principales ventajas:

- Es un lenguaje multiplataforma. Corre en diversas plataformas utilizando el mismo código fuente: diferentes versiones de Unix, Windows (95, 98, NT, ME, 2000, XP, Vista, Seven) y Mac.
- Completamente orientado a la web.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- La sintaxis de PHP es similar a la de C, por esto cualquier programador con experiencia en lenguajes del estilo C podrá entender rápidamente PHP.
- El código fuente escrito en PHP es invisible al navegador y al cliente ya que es el servidor el que se encarga de ejecutar el código y enviar su resultado HTML al navegador. Esto hace que la programación en PHP sea segura y confiable.
- Puede interactuar con numerosos motores de bases de datos.
- PHP generalmente es utilizado como módulo de Apache, lo que lo hace extremadamente veloz.
- PHP es de código abierto.

El 13 de julio de 2004, fue lanzado PHP5, utilizando el motor Zend Engine 2.0. Su principal objetivo ha sido mejorar los mecanismos de POO para solucionar las carencias de las anteriores versiones.PHP5 también ha posibilitado:

- Mejoras de rendimiento.
- Mejor soporte para MySQL con extensión completamente reescrita.
- Mejor soporte a XML (XPath, DOM).
- Soporte nativo para SQlite.
- Soporte Integrado para SOAP.
- Iteradores de datos.
- Manejos de excepciones. (14)

1.2.2. JavaScript

JavaScript es un recurso muy potente para lograr mejorar las páginas web así como lograr un óptimo funcionamiento de sus proyectos web. Es el lenguaje más utilizado para programar scripts del lado del cliente, basado en objetos y guiado por eventos, diseñado específicamente para el desarrollo de aplicaciones cliente-servidor dentro del ámbito de Internet. Además es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Los programas JavaScript van incrustados en los documentos XHMTL o en un fichero js, y se encargan de realizar acciones en el cliente, como puede ser: pedir datos, mostrar mensajes y comprobar campos.

Las principales características de Java Script como lenguaje son:

- **Interpretado:** No requiere compilación. El navegador del usuario se encarga de interpretar las sentencias JavaScript contenidas en una página HTML y las ejecuta adecuadamente.
- **Estructurado:** provee todos los recursos de un lenguaje estructurado como C (funciones, ciclo por conteo y por condición, condicionales, condicionales múltiples, y demás estructuras).
- **Tipado Dinámico:** como la mayoría de los lenguajes scripts los tipos de las variables están asociados al valor que contiene en un momento dado.
- **Orientado a Objetos:** implementa una variante del orientado a objetos no basada en clases propiamente sino en objetos prototipos a través de los cuales se pueden crear otros objetos idénticos y extender sus funcionalidades.

Ventajas:

- Los programas escritos en este lenguaje no requieren de mucha memoria ni tiempo adicional de transmisión, por ser pequeños y compactos.
- Es independiente de la plataforma hardware o sistema operativo, y 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. (15)

1.3. Marcos de trabajo y librerías

En el desarrollo de software, un framework o marco de trabajo es una estructura conceptual y tecnológica que contempla herramientas de apoyo para la organización y el desarrollo de un software. Este puede incluir programas de soporte, librerías de código, códigos script, u otras aplicaciones que faciliten el desarrollo. Los objetivos principales que persigue son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones.

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener y facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. (16)

1.3.1. EXTJS 3.3.1

ExtJS es una librería JavaScript para la creación de aplicaciones enriquecidas del lado del cliente. Se caracteriza principalmente por poseer componentes de interfaz de usuario personalizables, cuenta con un buen diseño y documentación. Además flexibiliza el manejo de componentes de la página como el DOM, peticiones AJAX, DHTML y crear interfaces de usuario bastante funcionales. Originalmente construida como una extensión de la biblioteca YUI, y sale al mercado el 1 de abril de 2007. Esta librería incluye:

- Componentes de Interfaz de Usuario (UI) de alto rendimiento y personalizables.
- Modelo de componentes extensibles.
- Un API fácil de usar.
- Licencias Open Source (GPL) y comerciales.

Ventajas:

- Una de las grandes ventajas de utilizar ExtJS es que permite crear aplicaciones complejas utilizando componentes predefinidos.
- Evita el problema de tener que validar el código para que funcione bien en cada uno de los navegadores (Firefox, IE, Safari, Opera).
- El funcionamiento de las ventanas flotantes lo pone por encima de cualquier otro.
- Relación entre Cliente-Servidor balanceado: Se distribuye la carga de procesamiento entre,

permitiendo que el servidor pueda atender más clientes al mismo tiempo.

- Eficiencia de la red: Disminuye el tráfico en la red pues las aplicaciones cuentan con la posibilidad de elegir que datos desea trasmitir al servidor y viceversa (Criterio este que puede variar con el uso de aplicaciones de pre-carga).
- Comunicación asíncrona. En este tipo de aplicación el motor de render puede comunicarse con el servidor sin necesidad de estar sujeta a un clic o una acción del usuario, dándole la libertad de cargar información sin que el cliente se dé cuenta. (17)

1.3.2. Symfony 1.4.8

Symfony es un framework que sigue el patrón arquitectónico MVC[1] diseñado para optimizar el desarrollo de las aplicaciones web. Automatiza las tareas más comunes de los proyectos web, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Symfony es maduro, estable, profesional y está muy bien documentado. Está desarrollado completamente con PHP 5 y está enfocado al desarrollo en el mismo lenguaje de programación. Algunas de sus características más relevantes son:

- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos. Esto se garantiza a través de un ORM (Propel o Doctrine).
- Implementa Front Controller como patrón derivado del MVC.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales y adaptables a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.
- Soporta la instalación de extensiones o “plugins” para añadir nuevas funcionalidades. (16)

[1] Modelo-Vista-Controlador

1.3.3. Propel

Propel es un es un framework para el mapeo objeto-relacional (ORM) escrito en PHP5, que funciona como una capa de abstracción entre el usuario y la base de datos. Básicamente se encarga de abstraer o separar al desarrollador del lenguaje estructurado de consultas (SQL), desarrollando clases a partir de las tablas y realizando relaciones entre ellas para facilitar el trabajo con las mismas y con sus atributos. Es también un proyecto de software libre, fácil de usar y bien documentado. Solo es necesario definir la estructura de su base de datos en un archivo XML y propel solo gestionará las bases de datos Este framework está completamente integrado en Symfony e incluso es su ORM por defecto.

Una de las tantas ventajas del uso de esta capa de abstracción, es que evita utilizar una sintaxis específica de un sistema de bases de datos en particular, esta capa transforma automáticamente las consultas, optimizando a la hora de particularizar para cada motor soportado. (16)

1.3.4. Lime

La automatización de pruebas es uno de los grandes avances en la programación. En el desarrollo de software, las pruebas aseguran la calidad del producto. Symfony incluye su propio Framework para crear pruebas unitarias, este es llamado Lime. Se basa en la librería Test::More de Perl y es compatible con TAP, lo que significa que los resultados de las pruebas se muestran Con el formato definido en el *"Test Anything Protocol"*, creado para facilitar la lectura de los resultados de las pruebas. Lime Proporciona el soporte para las pruebas unitarias, es más eficiente que otros frameworks de pruebas de PHP y tiene las siguientes ventajas:

- Ejecuta los archivos de prueba en un entorno independiente para evitar interferencias entre las diferentes pruebas. No todos los frameworks de pruebas garantizan un entorno de ejecución "limpio" para cada prueba.
- Las pruebas de Lime son fáciles de leer y sus resultados también lo son. En los sistemas operativos que lo soportan, los resultados de Lime utilizan diferentes colores para mostrar de forma clara la información más importante.
- El núcleo de Lime se valida mediante pruebas unitarias.
- Está escrito con PHP, es muy rápido y está bien diseñado internamente. Consta únicamente de un archivo, llamado lime.php, y no tiene ninguna dependencia. (16)

Los frameworks que se utilizaran son Propel, ExtJS y Symfony. El uso de estos simplifica el desarrollo de aplicaciones mediante la automatización de las tareas, proporcionando estructura al código fuente,

creando código más legible y fácil de mantener. Además con la utilización de ExtJs será posible utilizar componentes predefinidos para la interfaz visual del módulo.

1.4. Entorno Integrado de desarrollo

Los desarrolladores que desean utilizar su tiempo de manera más eficiente posible, recurren al uso de un entorno de desarrollo integrado. Un entorno integrado de desarrollo o IDE (de sus siglas en inglés), es un programa que está compuesto por un conjunto de herramientas que utilizan los programadores para desarrollar código. Estos IDE proveen un marco de trabajo amigable y están pensados para su utilización con un único lenguaje de programación o con varios de estos.

Entre las herramientas de las que usualmente dispone un entorno de desarrollo integrado se pueden encontrar las siguientes: un editor de texto, un intérprete, un compilador, herramientas para la automatización, un depurador, un sistema de ayuda para la construcción de interfaces gráficas de usuario y un sistema de control de versiones. A continuación se mencionan y caracterizan dos de los IDEs más conocidos

1.4.1. ZendStudio

ZendStudio es una poderosa herramienta desarrollada por Zend Technologies Ltd. Permite agilizar el desarrollo web y simplificar proyectos complejos usando el lenguaje de programación PHP. Además de servir de editor de texto para páginas PHP, proporciona una serie de ayudas que pasan desde la creación y gestión de proyectos hasta la depuración de código.

El programa entero está escrito en Java, lo que a veces supone que no funcione tan rápido como otras aplicaciones de uso diario. Sin embargo, esto ha permitido a Zend lanzar con relativa facilidad y rapidez versiones del producto para Windows, Linux y MacOS, aunque el desarrollo de las versiones de este último sistema se retrase un poco más. (18)
Entre sus características más significativas están:

- No requiere la instalación previa de PHP ni del entorno de ejecución de Java.
- Soporte para PHP 4 y PHP 5.
- Resaltado de sintaxis, autocompletado de código, ayuda de código y lista de parámetros de funciones y métodos de clase.
- Inserción automática de paréntesis y corchetes de cierre.
- Detección de errores de sintaxis en tiempo real.
- Funciones de depuración.

- Instalación de barras de herramientas para Internet Explorer y Mozilla Firefox (opcional).
- Soporte para gestión de grandes proyectos de desarrollo.
- Manual de PHP integrado.
- Soporte para navegación en bases de datos y ejecución de consultas SQL. (19)

1.4.2. NetBeans 7.0

NetBeans es un entorno de desarrollo integrado (IDE) de código abierto y es un producto libre y gratuito sin restricciones de uso, las aplicaciones basadas en esta plataforma pueden ser extendidas fácilmente por otros desarrolladores de software. Además, ofrece todas las herramientas necesarias para crear aplicaciones de escritorio, empresariales, web y móviles con el lenguaje Java, JavaFX, C/C ++ y lenguajes dinámicos como PHP, JavaScript, Groovy y Ruby. Es fácil de instalar y se puede ejecutar tanto en Windows, Linux, Mac OS X como en Solaris. Otras de las características novedosas que integra este IDE son:

- Introducción de JDK 7 de apoyo, incluidas las mejoras del editor (sintaxis, pistas).
- Asistente simplificado con instalación guiada al controlador JDBC.[2]
- Edición y despliegue de los procedimientos almacenados.
- Soporte de edición de HTML 5.
- Diseñador de GridBagLayout nuevas para mejorar el desarrollo GUI[3] Swing.
- Mejora de la detección de cambios externos. (20)

1.4.3. Selección del entorno integrado de desarrollo a utilizar

Después de un análisis de los IDE de desarrollo para aplicaciones web Netbeans y ZendStudio, se decide utilizar Netbeans 7.0 ya que este proporciona un marco de trabajo fácil y amigable. Es gratuito y de código abierto. Presenta un completamiento de código eficiente. Soporta el lenguajes Java, JavaFX, C/C ++ y otros lenguajes como PHP, JavaScript, HTML, Groovy y Ruby Además la plataforma puede ser extendida fácilmente por otros desarrolladores de software. Por su parte ZendStudio requiere licencia de pago y no incluye editor visual HTML.

1.5. Lenguaje de modelado

Lenguaje Unificado de Modelado (UML por sus siglas en inglés, *Unified Modeling Language*), es el lenguaje de modelado de sistemas de software más conocido y utilizado actualmente. Es un lenguaje gráfico que tiene como objetivo visualizar, especificar, construir y documentar los artefactos de un

[2] Java Database Connectivity
[3] Interfaz Gráfica de Usuario

sistema que se crean a través de las distintas etapas de su ciclo de vida, principalmente durante el análisis y el diseño del mismo. UML estandariza diferentes tipos de diagramas para representar gráficamente un sistema desde distintos puntos de vista. El hecho de que UML sea un lenguaje de propósito general proporciona gran flexibilidad y expresividad a la hora de modelar sistemas. (21)

Mediante UML es posible establecer la serie de requerimientos y estructuras necesarias para plasmar un sistema de software previo al proceso intensivo de escribir código. Este posee más características visuales que programáticas, las mismas facilitan a integrantes de un equipo multidisciplinario (analistas, diseñadores, especialistas de área y programadores) participar e intercomunicarse fácilmente.

1.6. Herramienta de modelado

Visual Paradimg for UML 6.4

Es una herramienta CASE[4], desarrollada por la compañía Visual Paradigm International, que utiliza UML como lenguaje de modelado. Soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Es muy potente, fácil de instalar, utilizar y actualizar. Te permite dibujar todo tipo de diagramas UML. Facilita la interoperabilidad con otras herramientas CASE como el Rational Rose y se integra con las siguientes herramientas Java: Eclipse/IBM WebSphere, Jbuilder, NetBeans, Oracle Jdeveloper, BEA Weblogic. Esta herramienta contribuye a una rápida construcción de aplicaciones de calidad. Entre otras de sus principales características se encuentran:

- Software libre.
- Disponibilidad en múltiples plataformas (Windows, Linux).
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Licencia: gratuita y comercial.
- Soporta aplicaciones Web.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Varios idiomas.
- Fácil de instalar y actualizar.

[4] Ingeniería de Software Asistida por Computación.

- Compatibilidad entre ediciones.
- Soporte ORM - Generación de objetos Java desde la base de datos.
- Generación de bases de datos - Transformación de diagramas de Entidad-Relación en tablas de base de datos.
- Ingeniería inversa de bases de datos - Desde Sistemas Gestores de Bases de Datos (DBMS) existentes a diagramas de Entidad-Relación. (22)

1.7. Metodologías de desarrollo

Cada día se pretende construir software con mayor calidad pero esto no es labor fácil. El éxito del producto dependerá en gran medida de la metodología de desarrollo de software escogida por el equipo para guiar y organizar actividades que conlleven a las metas trazadas. Piattini define a dicha metodología como *"un conjunto de procedimientos, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software".*

No existe una metodología universal. Las características de cada proyecto en cuanto a tiempo de duración, equipo de desarrollo y recursos varían continuamente, por lo que a lo largo de los años se han propuesto numerosas metodologías que se pueden agrupar en: metodologías ágiles o ligeras y metodologías tradicionales o robustas. En dependencia de las particularidades de cada producto de software se escoge la metodología a ser aplicada.

Tabla 2: Comparación entre metodologías. (23)

Metodologías Ágiles	Metodologías Tradicionales
Basadas en heurísticas provenientes de prácticas de producción de código	Basadas en normas provenientes de estándares seguidos por el entorno de desarrollo
Especialmente preparados para cambios durante el proyecto	Cierta resistencia a los cambios
Reglas de trabajo Impuestas internamente (por el equipo)	Reglas de trabajo Impuestas externamente

Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
No existe contrato tradicional o al menos es bastante flexible	Existe un contrato prefijado
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Grupos pequeños y trabajando en el mismo sitio	Grupos grandes y posiblemente distribuidos
Pocos artefactos	Más artefactos
Pocos roles	Más roles
Menos énfasis en la arquitectura del software	La arquitectura del software es esencial y se expresa mediante modelos

1.7.1. RUP

Las metodologías tradicionales o robustas son aquellas que están guiadas por una fuerte planificación durante todo el proceso de desarrollo, en la cual se establecen estrictamente las actividades involucradas, los roles definidos, los artefactos que se deben producir, las herramientas y notaciones que serán utilizadas así como el modelado y documentación detallada. (24)

Una de las metodologías de desarrollo de este tipo más usada actualmente es RUP (*Rational Unified Process*) en español Proceso Unificado de Desarrollo: RUP es un proceso para el desarrollo de un proyecto de un software que define quién, cómo, cuándo y qué debe hacerse en el proyecto. Tiene como objetivo principal asegurar la producción de software de calidad dentro de plazos y presupuestos predecibles.

Esta metodología se destaca por tres características esenciales:

- Dirigido por casos de uso: Estos reflejan las necesidades de los clientes, extraídas cuando se modela del negocio y representadas después a través de los requerimientos. A partir de aquí los casos de uso guían el proceso de desarrollo ya que los modelos que se obtienen, como resultado de los diferentes flujos de trabajo, representan la realización de los casos de uso.

- Centrado en la arquitectura: La arquitectura muestra la visión común del sistema completo en la que el equipo de proyecto y los usuarios deben estar de acuerdo, por lo que describe los elementos del modelo que son más importantes para su construcción, los cimientos del sistema que son necesarios como base para comprenderlo, desarrollarlo y producirlo económicamente.
- Iterativo e Incremental: RUP propone que cada fase se desarrolle en iteraciones. Una iteración involucra actividades de todos los flujos de trabajo, aunque desarrolla fundamentalmente algunos más que otros.

RUP divide el proceso de desarrollo en fases. Además, el ciclo de vida de esta metodología contiene flujos de trabajos, los cuales se dividen en flujos de trabajo de desarrollo y flujos de trabajo de soporte (ver Figura 2).

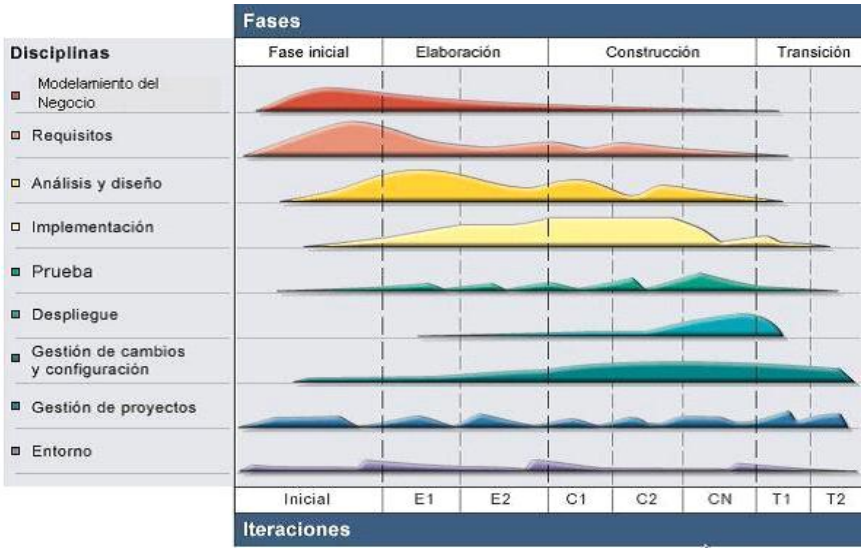


Figura 2: Flujos de trabajo y fases en RUP.

Inicio: se hace un plan de fases, se identifican los principales casos de uso y se identifican los riesgos.

Elaboración: se hace un plan de proyecto, se completan los casos de uso y se eliminan los riesgos.

Construcción: se enfoca en la elaboración de un producto completamente operativo y eficiente.

Transición: se instala el producto y se capacita al usuario. Como consecuencia de esto suelen surgir nuevos requisitos a ser analizados. (25)

1.7.2. OpenUP

Las metodologías ágiles cambian significativamente algunos de los énfasis de los métodos ingenieriles. La diferencia inmediata es que son menos orientados al documento, exigiendo una cantidad más pequeña de documentación para una tarea dada. Están más orientadas a la generación de código con ciclos muy cortos de desarrollo, se dirigen a equipos de desarrollo pequeños, hacen

especial hincapié en aspectos humanos asociados al trabajo en equipo e involucran activamente al cliente en el proceso. (24)

OpenUp es un ejemplo de metodología ágil. Este proceso de desarrollo unificado está basado en *Rational Unified Process* (RUP), fue liberado por el EPF (*Eclipse Process Framework*), se construyó sobre una donación realizada por la IBM del *Basic Unified Process*. Fue entregada a Eclipse a fines de 2005 y renombrado como OpenUp en 2006. Es reconocido mundialmente como uno de los procesos de desarrollo de software de mayor calidad.

Es un proceso interactivo de desarrollo de software mínimo, completo y extensible. El proceso es mínimo en que solamente el contenido fundamental es incluido; es completo en que puede ser manifestado como todo el proceso para construir un sistema; extensible en que puede ser utilizado como fundamento sobre el cual el contenido de proceso se pueda agregar o adaptar según lo necesitado. (26)

Mantiene las mismas características de RUP pues está dirigido por casos de uso, centrado en la arquitectura y además es iterativo e incremental. Permite un abordaje ágil al proceso de desarrollo de software, con sólo proveer un conjunto simplificado de contenidos, fundamentalmente relacionados con orientación, productos de trabajo, roles, y tareas. Desarrolla un ciclo de vida interactivo que mitiga el riesgo a tiempo y ofrece demostrar resultados en curso al cliente del proyecto. Se centra en una arquitectura temprana para reducir al mínimo los riesgos y organizar el desarrollo. Además ofrece la administración de diferentes áreas del proyecto, manejando el ciclo de vida de manera apropiada. Presenta cuatro principios básicos interrelacionados, a saber:

- Colaboración para unificar intereses y compartir conocimientos.
- Equilibrio de prioridades competentes a maximizar el valor de los involucrados con el resultado del proyecto.
- Enfoque en la articulación de la arquitectura.
- Desarrollo continuo para obtener realimentación y realizar las mejoras respectiva.

El ciclo de vida de un proyecto según la metodología OpenUP se divide en 4 fases fundamentales: Concepción, Elaboración, Construcción y Transición. Propone 6 flujos de Trabajo:

- **Requerimientos**: en este flujo de trabajo se realizan entrevistas con el cliente para comprender el problema a resolver y se definen los requerimientos.
- **Análisis y Diseño**: se realiza el diseño de los requisitos que serán después implementados.
- **Implementación**: en esta disciplina se realiza la implementación del sistema basándose en el

diseño realizado.

- **Prueba**: busca los defectos a los largo del ciclo de vida.
- **Gestión del Proyecto**: involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
- **Gestión de Configuración y Cambios**: describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización y actualización, control de versiones.

1.7.3. Selección de la metodología de desarrollo a utilizar

Después de haber realizado un estudio sobre las metodologías de desarrollo de software se determinó, por políticas del departamento Integración de Soluciones, utilizar OpenUP, por ser un proceso completo, práctico y muy eficiente. Esta metodología está diseñada para el desarrollo de proyectos pequeños, característica presente en cada uno de los equipos de trabajo de la línea. Además, mantiene las principales características de RUP, obviando solamente las partes opcionales que esta propone, permitiendo el desarrollo de aplicaciones mediantes la generación de artefactos ligeros apropiados, utilizando el lenguaje UML e incrementando así las probabilidades de éxito en función de costo, tiempo y alcance.

1.8. Conclusiones del capítulo

En el presente capitulo se han abordado los principales conceptos relacionados con el objeto de estudio y algunas tendencias en el desarrollo de aplicaciones web con el fin de proveer una solución factible que permita conformar la versión 2.0 del Sistema Generador Dinámico de Reportes. En aras de agilizar el desarrollo de la solución informática, se adopta Symfony 1.4 como marco de trabajo el cual se destaca por poseer una arquitectura flexible. Teniendo en cuenta la integración existente entre el IDE Netbeans 7.0 y el marco de trabajo Symfony, se decide utilizar dicho IDE para el desarrollo de la solución. Para la programación del lado del servidor uno de los seleccionados fue PHP 5 por poseer características ideales que lo distinguen del resto de los lenguajes de programación. Como metodología de desarrollo, por políticas del departamento Integración de Soluciones, se determinó utilizar OpenUP, por ser un proceso completo, práctico y muy eficiente, según las características del grupo de desarrollo y del proyecto.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA SOLUCIÓN

2. Introducción

En el presente capítulo se describe el proceso desarrollado durante la fase de análisis y diseño. El mismo incluye el modelo de dominio del sistema a partir del cual se definen una serie de requisitos funcionales y no funcionales con que contará la solución. Se generan los diferentes diagramas y artefactos necesarios, para dar cumplimiento a los requisitos trazados. Además se describen los patrones de diseño que se utilizarán.

2.1. Modelo de Dominio Propuesto

El modelo de dominio es una representación visual estática del entorno real objeto del proyecto y se realiza cuando no están claros los procesos o cuando no se identifican claramente los actores y trabajadores del negocio. Este modelo ayuda a comprender los conceptos que utilizan los usuarios, es decir, los conceptos con los que trabajan y con los que deberá trabajar la aplicación. Su objetivo es contribuir a la comprensión del contexto y de los requisitos del sistema. Se describe mediante diagramas de UML, especialmente mediante diagramas de clases. A continuación se presenta el modelo de dominio propuesto (ver Figuara 3) para un mejor entendimiento del sistema:

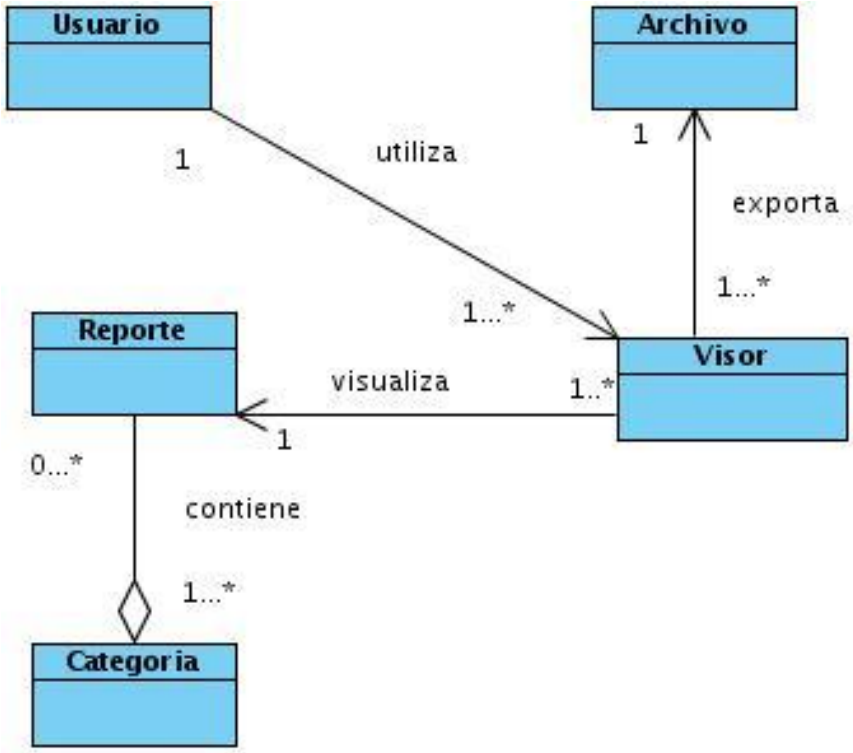


Figura 3: Modelo de Dominio.

Usuario: Persona responsable y capacitada para acceder y utilizar el sistema informático Generador Dinámico de Reportes.

Reporte: Informe generado por el sistema que organizan y exhibe de manera resumida la información contenida en una base de datos de acuerdo a un interés específico. Su función es mostrar los datos por medio de un diseño atractivo y fácil de interpretar por los usuarios.

Visor: Herramienta que permite la visualizar, exportar un reporte creado mediante el Sistema Generador Dinámico de reportes.

Categoría: Grupo en el que pueden clasificarse los reportes que comparten características en común. Estos grupos facilitan la búsqueda de información.

Archivo: Documento que se obtiene como resultado de la exportación de un reporte y que puede estar en diferentes formatos (HTML, PDF, Excel, CSV, RTF, ODT, XML).

2.2. Requerimientos del sistema

Un requerimiento puede definirse como un atributo necesario dentro de un sistema, que puede representar una capacidad, una característica o un factor de calidad del sistema de tal manera que le sea útil a los clientes o a los usuarios finales.

2.2.1. Requerimientos funcionales

Los requisitos funcionales (RF) son capacidades o condiciones que el sistema debe cumplir. Este tipo de requerimiento específica algo que el sistema entregado debe ser capaz de realizar.
Para cumplir los objetivos de este sistema el mismo debe permitir:

RF1 Visualizar lista de reportes existentes en el sistema agrupados por categorías.
RF2 Buscar reporte registrado en el sistema.
RF3 Actualizar lista de reportes existentes en el sistema.
RF4 Mostrar vista previa del reporte seleccionado.
RF5 Exportar reporte a HTML
RF6 Exportar reporte a PDF
RF7 Exportar reporte a Excel
RF8 Exportar reporte a CSV
RF9 Exportar reporte a ODT
RF10 Exportar reporte a RTF
RF11 Exportar a XML

RF12 Configurar Opciones de exportación para HTML
RF13 Configurar Opciones de exportación para PDF
RF14 Configurar Opciones de exportación para Excel
RF15 Configurar Opciones de exportación para CSV

2.2.2. Requerimientos no funcionales

Los requisitos no funcionales (RNF) son propiedades o cualidades que el producto debe tener. Especifican propiedades, que de una forma u otra restringen el entorno del sistema o de la implementación como por ejemplo rendimiento, interfaz de usuario, facilidad de mantenimiento, dependencias de la plataforma, entre otros. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Para el sistema propuesto se definen los siguientes requisitos no funcionales:

Requerimientos de usabilidad.

La aplicación tiene que ser capaz de ofrecer facilidades de uso para un buen entendimiento y aceptación del producto por los usuarios finales. Debe ser sencilla a la vista de los usuarios. La herramienta debe ser WEB, pero con características muy similares a las aplicaciones de escritorio en cuanto al diseño de las interfaces visuales y los tiempos de respuesta de la interacción del usuario con el sistema.

Requerimientos de Software

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

- SO: 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 SO.

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):

- SO: 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

Las PC clientes debe cumplir con los siguientes requisitos de hardware:

- Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 256MB.
- 20 GB de espacio en disco duro.

Las PC Servidor deben cumplir con los siguientes requisitos de 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.

Requerimientos de Fiabilidad

El sistema debe estar disponible las 24 horas del día. En caso de fallo, pudiera estar fuera de servicio por un período de 72 horas máximo. La precisión y exactitud de las salidas del sistema se corresponden con la calidad y exactitud de la información contenida en las base de datos desde donde se extraigan los reportes. El sistema no será responsable por la falta de veracidad de dicha información.

Requerimientos de Eficiencia

La eficiencia del sistema depende en gran medida de la velocidad de conexión a las base de datos donde se encuentre, así como del volumen de información contenido en las mismas. Sin embargo el sistema debe mantener tiempos de respuestas en un marco razonable.

Restricciones de Diseño e Implementación

El sistema deberá hacer uso de los lenguajes PHP (versión 5.2 o superior) y JavaScript para la programación del lado del servidor y del lado del cliente respectivamente. Como frameworks de desarrollo se usarán Ext Js, y Symfony, este último propone una arquitectura modular en tres capas: el modelo, la vista y el controlador. Como herramienta de desarrollo de usará NetBeans 7.0 y como sistema gestor de base de datos se utilizará PostgreSQL 8.4. Otra componente importante es el JasperReports, el cual constituye el núcleo del proceso de generación dinámico de reportes.

2.3. Modelo de Casos de Uso del Sistema

Un modelo de casos de uso es un modelo del sistema que contiene actores, casos de uso y sus relaciones. Este modelo describe la funcionalidad propuesta del nuevo sistema. Sirve como acuerdo entre clientes y desarrolladores, y proporciona la entrada fundamental para el análisis, el diseño y las pruebas.

2.3.1. Diagrama de Casos de Uso del Sistema.

Un diagrama de caso de uso es una representación grafica de parte o el total de los actores y casos de usos del sistema, incluyendo sus interacciones. Los diagramas de casos de uso sirven para especificar la funcionalidad y el comportamiento de un sistema mediante su interacción con los usuarios u otros sistemas. Un diagrama de casos de uso consta de los siguientes elementos:

- **Actor:** Se le llama actor a toda entidad externa al sistema que guarda una relación con éste y que le demanda una funcionalidad. Es un rol que un usuario juega con respecto al sistema. No necesariamente representa a una persona en particular, sino más bien la labor que realiza frente al sistema. Incluye usuarios humanos y otros sistemas computarizados.
- **Casos de Uso:** Los casos de uso son fragmentos de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. De manera más precisa, un caso de uso especifica una secuencia de transacciones que el sistema puede desarrollar respuesta a un evento que inicia un actor sobre el propio sistema.

- **Relaciones:** Una relación es una conexión entre los elementos del modelo**.** En un diagrama de casos de uso, además de las relaciones entre casos de uso y actor, llamadas asociaciones y las relaciones entre casos de uso, de dependencias (<<include>> y <<extends>>), pueden existir relaciones de herencia ya sea entre casos de uso o entre actores.

La siguiente figura representa el diagrama de casos de uso referente al módulo Visor de Reportes del Generador Dinámico de Reportes (ver Figura 4):

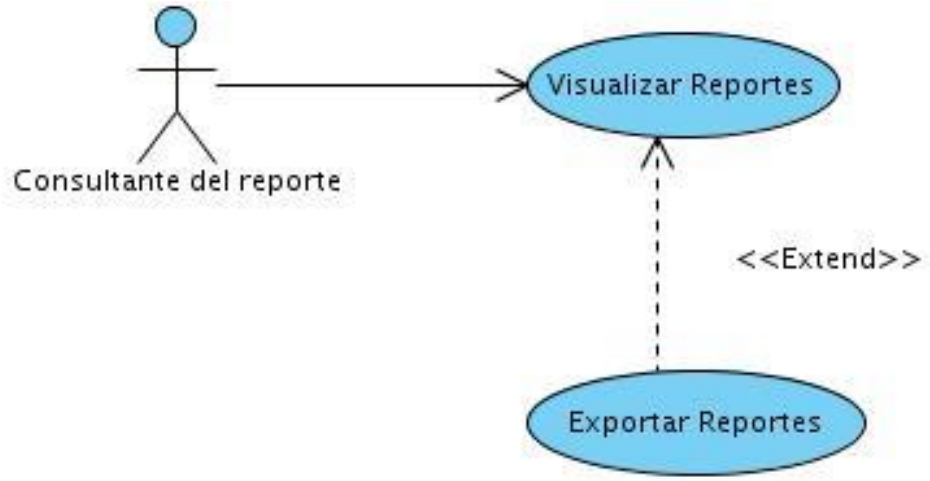


Figura 4: Diagrama de Casos de Uso del Sistema (DCUS).

Visualizar Reportes: Permite mostrar el listado de reportes existentes en el sistema agrupados por categorías y mostrar la vista previa de un reporte seleccionado. Además brinda la posibilidad de buscar un reporte específico en la lista de reportes.

Exportar Reportes: Permite exportar un reporte a diferentes formatos y brinda la posibilidad de configurar los parámetros para la exportación. Esos parámetros pueden ser: salida de datos por cantidad de páginas, algunas opciones de seguridad, entre otros.

Tabla 3: Especificación del Caso de Uso Visualizar Reportes

Objetivo	Visualizar los reportes registrados en el sistema
Actores	Consultante del reporte
Resumen	El caso de uso comienza al seleccionar la opción “Visor de reportes” muestra el listado de reportes existentes en el sistema agrupados por categorías. Además brinda la posibilidad de buscar un reporte específico en la lista de reportes. Finaliza al ser visualizada una vista previa de un reporte o al terminar de realizar cualquiera de estas operaciones.
Complejidad	Media
Prioridad	Alta
Precondiciones	Tiene que existir al menos un reporte. No deben existir cambios en el origen de datos asociado al reporte.

Postcondiciones	Se visualiza un reporte.	
Flujo de eventos		
Flujo básico <Nombre del flujo básico>		
	Actor	**Sistema**
1.	Selecciona la opción “Visor de Reportes”.	
2.		Muestra la lista de reportes existentes en el sistema, organizados por categorías. Permite la acción de: -Buscar un reporte. Ver Sección 1: “Buscar reporte”.
3.	Selecciona un reporte de la lista de reportes existentes en el sistema.	
4.		Resalta el reporte seleccionado y muestra una vista previa del mismo.
5.		Termina el caso de uso.
Flujos alternos		
Nº Evento <No se estableció la conexión con el servidor>		
	Actor	**Sistema**
1		2.1 Muestra un mensaje de notificación al actor informando que no se pueden obtener datos de la Base de Datos.
Sección 1: “Buscar Reporte”		
Flujo básico <Nombre del flujo básico>		
	Actor	**Sistema**
1.	Introduce el criterio de búsqueda (parte del nombre del reporte o el nombre completo).	
2.		Reduce la lista de reportes en concordancia con el criterio de búsqueda introducido.

		En caso de no coincidir el criterio de búsqueda con los nombres de reportes existentes, ver Flujo Alterno 1.
3	Selecciona un reporte de la lista de reportes existentes en el sistema.	
4		Resalta el reporte seleccionado y muestra una vista previa del mismo.
5		Termina el caso de uso.
Flujos alternos		
Nº Evento < No coincide el criterio de búsqueda con los nombres de reportes existentes>		
	Actor	**Sistema**
1.		2.1 La lista de reportes se visualiza vacía.

Relaciones	**CU Incluidos**	
	CU Extendidos	CU Exportar Reportes.

2.3.2. Matriz de Trazabilidad

La matriz de trazabilidad es una técnica que relaciona los requisitos funcionales a los diferentes elementos del desarrollo. Es una herramienta que se utiliza para saber que requisito quedan cubiertos por casos de uso y de esta forma garantizar que todos los elementos para el desarrollo del módulo sean ejecutados correctamente. (ver Figura 5)

Requisitos	CU Visualizar Reportes	CU Exportar Reportes
RF1 Visualizar lista de reportes existentes en el sistema agrupados por categorías.	X	
RF2 Buscar reporte registrado en el sistema.	X	
RF3 Actualizar lista de reportes existentes en el sistema.	X	
RF4 Mostrar vista previa del reporte seleccionado.	X	
RF5 Exportar reporte a HTML		X
RF6 Exportar reporte a PDF		X
RF7 Exportar reporte a Excel		X
RF8 Exportar reporte a CSV		X
RF9 Exportar reporte a ODT		X
RF10 Exportar reporte a RTF		X
RF11 Exportar a XML		X
RF12 Configurar Opciones de exportación para HTML		X
RF13 Configurar Opciones de exportación para PDF		X
RF14 Configurar Opciones de exportación para Excel		X
RF15 Configurar Opciones de exportación para CSV		X

Figura 5: Matriz de Trazabilidad

2.4. Diseño

La fase de Diseño es fundamental en el proceso de desarrollo de software. Para lograr un sistema con un alto grado de aceptación por los usuarios, es necesario tener en cuenta un cúmulo de elementos que permitan la creación de aplicaciones amigables y que favorezcan su usabilidad. Este flujo de trabajo es un refinamiento del análisis que tiene en cuenta los requisitos no funcionales del sistema. Los propósitos del diseño son:

- Transformar los requerimientos al diseño del futuro sistema
- Desarrollar una arquitectura sólida para el sistema.
- Crear una entrada apropiada y un punto de partida para actividades de implementación, capturando los requisitos o subsistemas individuales, interfaces y clases.
- Adaptar el diseño para que sea consistente con el entorno de implementación.

2.4.1. Diagrama de clases del diseño

Un tipo de diagrama de vital importancia para la definición del sistema es el diagrama de clases. El diagrama de clases de diseño representan los atributos, métodos así como la colaboración y las

responsabilidades de las diferentes clase, necesarias para realizar los casos de uso. A continuación se presentará el diagrama de clases del diseño del casos de uso Visualizar Reportes (ver Figura 6).

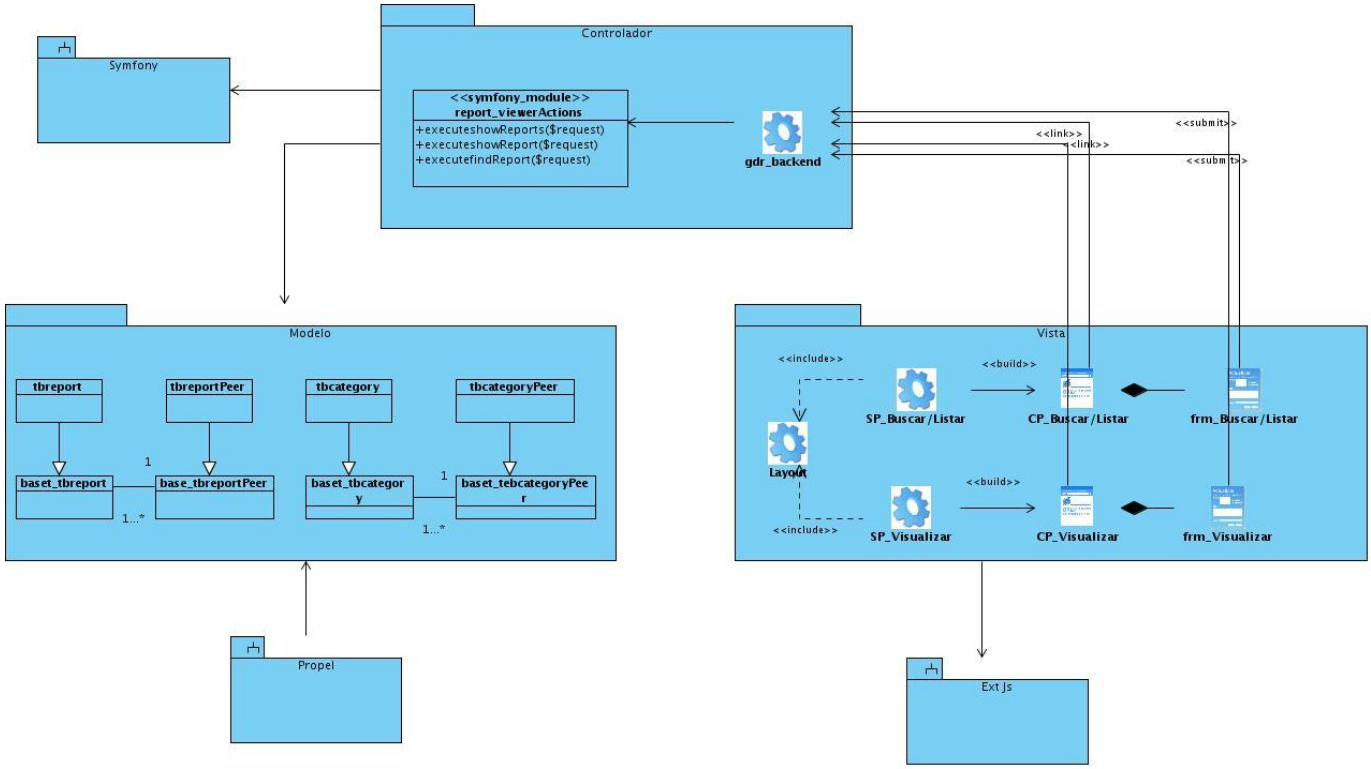


Figura 6: Diagrama de Clases del Diseño del caso de uso Visualizar Reportes.

Descripción de clases

En la capa Controlador está la clase **report_viewerActions**, esta contiene todas las acciones en dependencia del caso de uso que se trate, en este caso contiene todas las acciones referentes al CU Visualizar Reportes, que permiten mostrar el listado de los reportes existentes agrupados por categorías (showReportsAction ()), buscar un reporte teniendo como criterio de búsqueda el nombre del mismo (findReportAction ()) y visualizarlo (showReportAction ()). También se encuentra la clase **gdr_backend.php**, esta es el punto de entrada a la aplicación, es la que recibe las peticiones y decide el flujo que debe seguir para dar respuesta a una petición del usuario.

En la capa Modelo se encuentran las clases que genera el ORM Propel. Por cada tabla de la base de datos se generan 4 clases cada una con sus funciones específicas. Para el Caso de Uso Visualizar Reportes se utilizan **tbreport**, **tbreportPeer**, **tbcategory** y **tbcategoryPeer** que son clases de acceso a datos, responsables de interactuar con las clases de abstracción de datos y devolver los objetos que necesita el controlador y **baset_tbreport**, **baset_tbreportPeer** **baset_tbcategory** y

baset_tbcategoryPeer que son clases de abstracción de datos, responsables de realizar las operaciones con la base de datos.

Las clases que se encuentran en la capa Vista representan las interfaces de usuario de la aplicación, con las cuales los usuarios podrán interactuar.

Diagramas de Interacción

Los diagramas UML llamados diagramas de interacción (secuencia y colaboración) son algunos de los artefactos más importantes que se realizan en el análisis y diseño. Son utilizados con el objetivo de modelar los aspectos dinámicos de un sistema lo que conlleva modelar instancias concretas o prototípicas de clases interfaces, componentes y nodos, junto con los mensajes enviados entre ellos, todo en el contexto de un escenario que ilustra un comportamiento.

Los diagramas de secuencia muestran gráficamente las interacciones entre los objetos que estarán presentes en la aplicación destacando el orden temporal de los mensajes durante un escenario concreto.

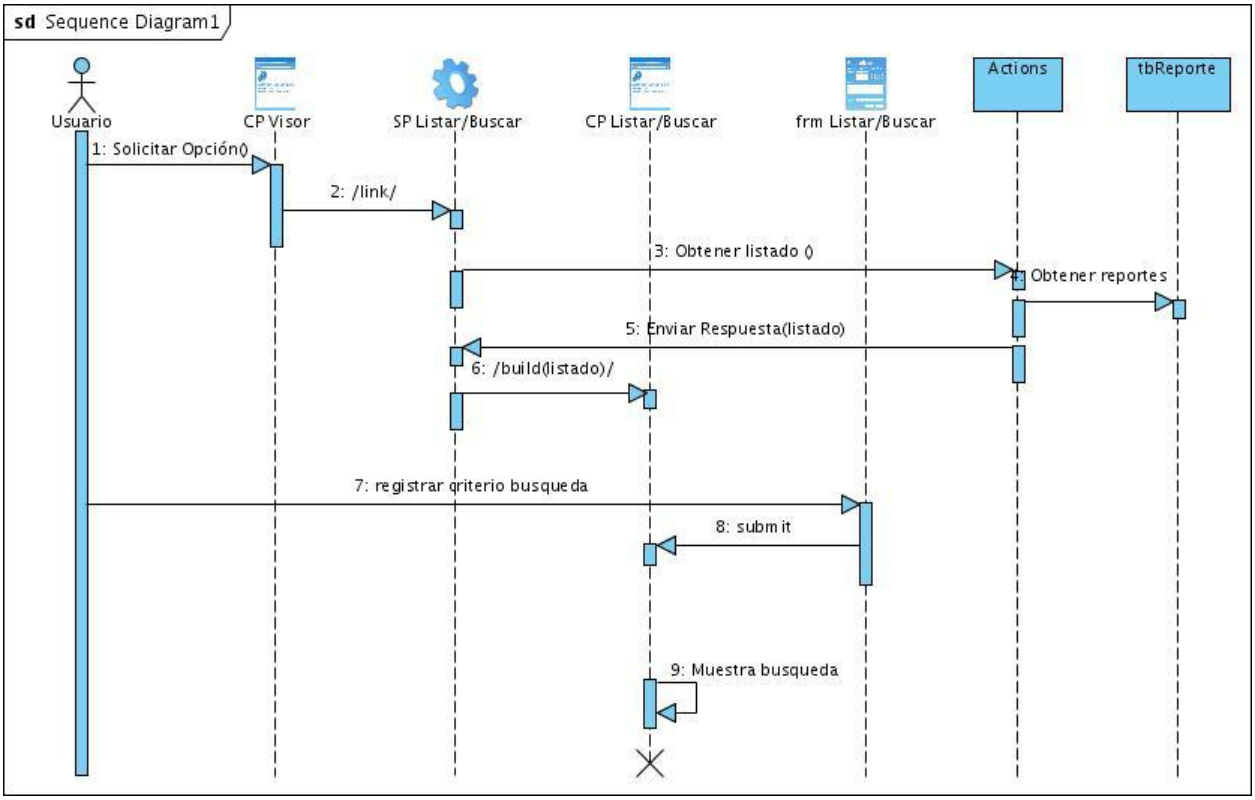


Figura 7: Diagrama de secuencias del caso de uso Visualizar Reportes

2.5. Fundamentación del uso de patrones

Las soluciones exitosas tienen sus raíces en patrones. Los patrones son una descripción de un problema y la esencia de su solución, de forma que esta pueda ser reutilizada en diferentes situaciones.

"Una arquitectura orientada a objetos bien estructurada está llena de patrones. La calidad de un sistema orientado a objetos se mide por la atención que los diseñadores han prestado a las colaboraciones entre sus objetos. Los patrones conducen a arquitecturas más pequeñas, más simples y más comprensibles". (Grady Booch[5])

Según la escala o nivel de abstracción los patrones se clasifican en: Patrones de arquitectura, Patrones de Diseño y Patrones de Idiomas.

2.5.1. Patrones arquitectónicos

Los patrones arquitectónicos constituyen una plantilla de construcción. Definen la estructura básica de un sistema, Proporciona un subconjunto de subsistemas predefinidos para conectarlos y pautas para su organización. Algunos patrones representan soluciones a problemas de rendimiento y otros pueden ser utilizados con éxito en sistemas de alta disponibilidad.

MVC (Modelo- Vista- Controlador)

El patrón arquitectónico Modelo Vista Controlador (MVC) ha demostrado ser fundamental a la hora de diseñar sistemas interactivos. El mismo propone la separación de los datos, la interfaz de usuario, y la lógica de control en componentes distintos:

Modelo: es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos.
Vista: presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario. Maneja la presentación visual de los datos representados por el Modelo.
Controlador: responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

Entre sus características se encuentran:

5 Grady Booch (nació el 27 de febrero de 1955) Diseñador de software. Director científico de Rational Software (ahora parte de IBM) y editor de una serie de Benjamin/Cummings.

- Existe una clara separación entre los componentes de un programa, lo cual permite implementarlos por separado.
- Existe un API muy bien definido, cualquiera que use el API, podrá reemplazar el Modelo, la Vista o el Controlador, sin aparente dificultad.
- La conexión entre el Modelo y sus Vistas es dinámica, se produce en tiempo de ejecución, no en tiempo de compilación. (27)

Entre sus ventajas se encuentran:

- Posee soporte para múltiples vistas**,** debido a que la Vista se separa del Modelo y no hay ninguna dependencia directa entre ambos.
- Proporciona un mecanismo de configuración a componentes complejos mucho más tratable que el puramente basado en eventos.
- Permite un mayor soporte a los cambios, debido a que los requisitos de interfaz tienden a cambiar más rápidamente que las reglas de negocio. Puesto que el modelo no depende de las vistas, la adición de nuevos tipos de vista al sistema generalmente no afectan al modelo. Por tanto, el ámbito del cambio se limita a la vista. (27)

2.5.2 Patrones de diseño

"Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí adaptada para resolver un problema general de diseño en un contexto particular."

Los patrones de diseño son abstracciones de alto nivel que documentan soluciones de diseño exitosas. Son fundamentales para reutilizar el diseño en el desarrollo orientado a objetos. Proveen un esquema para refinar los subsistemas o componentes de un sistema de software, o las relaciones entre ellos. Cada patrón se adapta a un cierto tipo de problema.

Los patrones de diseño pretenden:

- Proporcionar catálogos de elementos reusables en el diseño de sistemas software.
- Evitar la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente.
- Formalizar un vocabulario común entre diseñadores.
- Facilitar el aprendizaje de las nuevas generaciones de diseñadores condensando conocimiento ya existente.

Para el desarrollo de la aplicación se hará uso de los patrones GRASP para lograr un buen diseño orientado a objetos y algunos patrones GOF.

Patrones GRASP

Experto: Lo que plantea el patrón experto es asignar una responsabilidad a la clase que tiene toda la información necesaria para cumplir con dicha responsabilidad. Es un principio usado continuamente en el diseño orientado a objetos. Este patrón se evidencia en la capa Modelo, donde existen dos tipos de clases, las de acceso a datos y las de abstracción de datos, estas últimas (Peer del Modelo) poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen la información necesaria de la tabla que representan. Por ejemplo la clase baset_tbreport no conoce los atributos para interactuar con la base de datos tan solo se debe implementar la responsabilidad en la que se le avise a la clase baset_tbreportPeer que es la contenedora de los datos necesarios para ejecutar la acción. (ver Figura 9)

Creador: este patrón guía la asignación de responsabilidades relacionadas con la creación de objetos. Su propósito fundamental es encontrar un creador que debe ser conectado con el objeto producido en cualquier evento. El patrón Creador proporciona soporte a un bajo acoplamiento.

Bajo acoplamiento: El bajo acoplamiento soporta el diseño de clases más independientes, que reducen el impacto de los cambios. Se evidencia en la capa Modelo, ya que las clases de acceso a datos tienen poca dependencia de las de abstracción de datos, lo que permite mayor reutilización. En el módulo se pueden modificar las clases del modelo sin que se afecten las del controlador, y viceversa ya que estas son acciones independientes.

Controlador: El patrón controlador asigna la responsabilidad del manejo de los eventos del sistema a una clase. En 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. Con el uso de este patrón se puede obtener como beneficio el incremento del potencial de los elementos que pueden ser reutilizados.

Patrones GOF

Front Controller (Controlador frontal): se evidencia en el gdr_backend.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. (ver Figura 10)

Decorador: está presente en la layout, la cual contiene todo el código HTML y a su vez decora todas las plantillas a usar, en dependencia de la petición del usuario. (ver Figura 8)

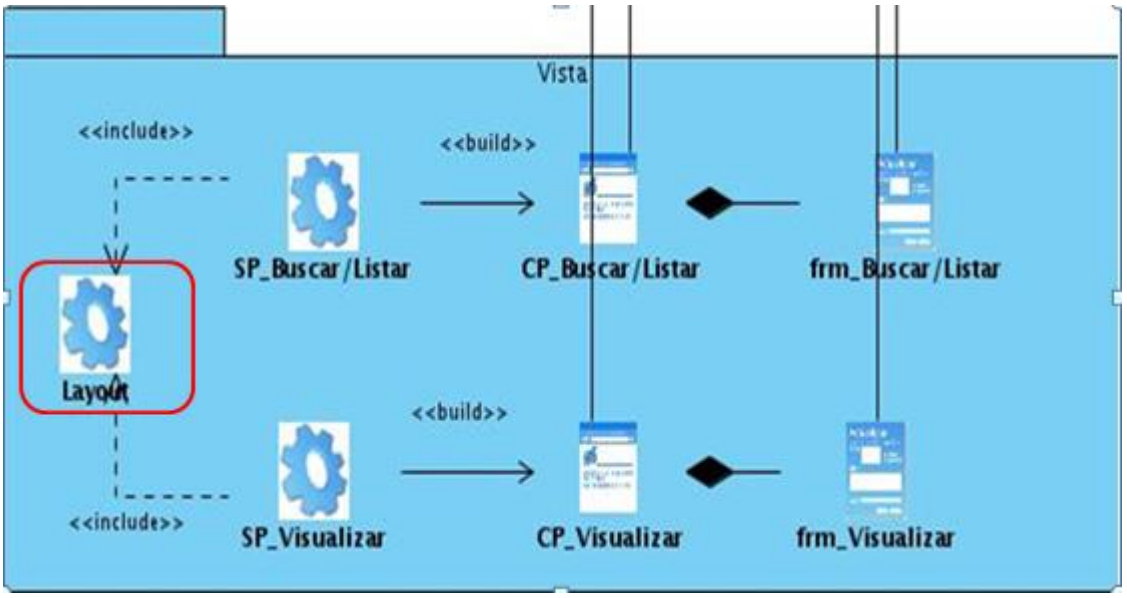


Figura 8: Patrón Decorador

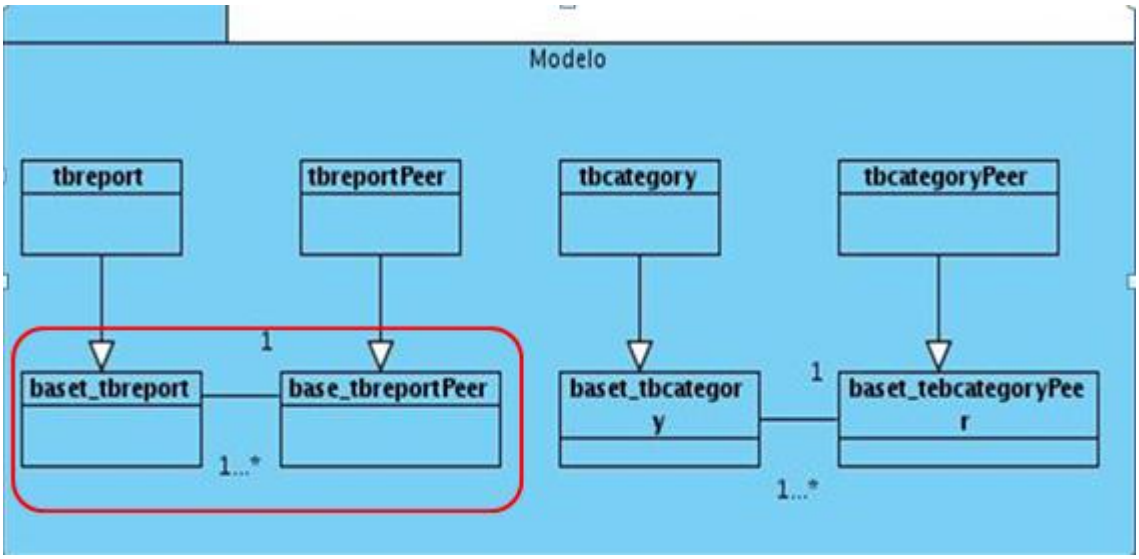


Figura 9: Patrón Experto

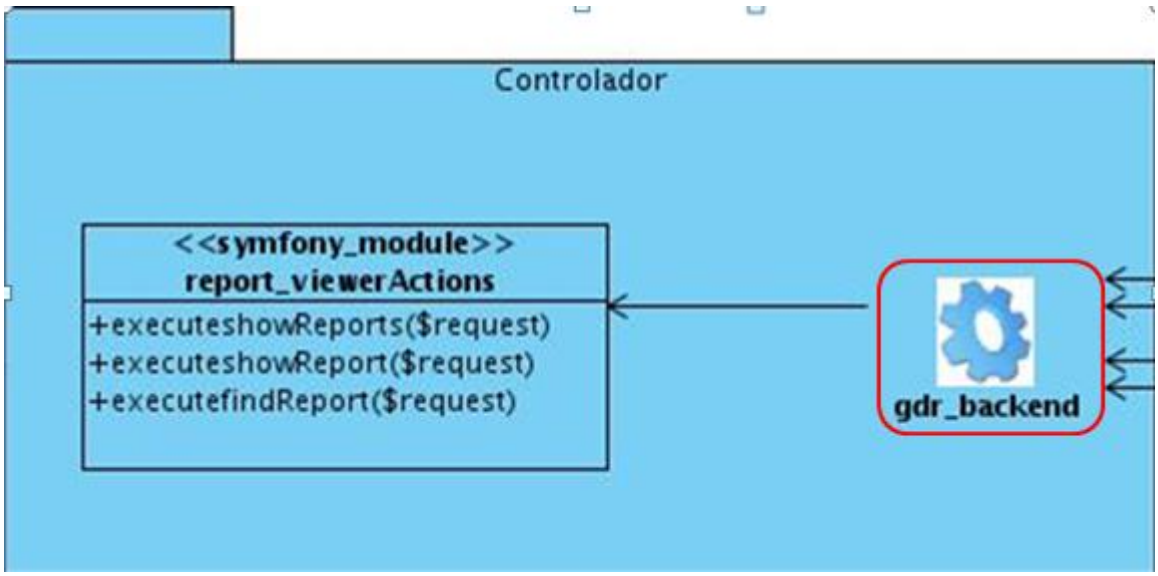


Figura 10: Patrón Front Controller

2.6. Conclusiones del capítulo

En este capítulo se realizó el modelo de dominio del negocio, además se identificaron los requisitos, tanto no funcionales como funcionales a cumplir por el sistema. Los requerimientos funcionales fueron agrupados por Casos de Uso del Sistema (CUS), quedando definidos dos: Visualizar Reportes y Exportar Reportes. Se presentaron los diagramas de clases del diseño y así como los de interacción y se describieron los patrones arquitectónicos y de diseño empleados como parte de la propuesta de solución.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBA

3. Introducción

En el presente capítulo quedan plasmados los detalles referentes al modelo de implementación que pone en práctica el diseño de la solución que se va a realizar. Durante el flujo de implementación se describe como los elementos del modelo de diseño se implementan en términos de componentes y cómo estos se organizan en el modelo de despliegue. Además, se describen las pruebas a realizar con el objetivo de comprobar que el sistema funciona correctamente.

3.1. Modelo de implementación

El modelo de implementación describe cómo los elementos de diseño se implementan en componentes. Describe los componentes a construir y su organización en nodos físicos en los que funcionará el sistema. Esta descripción es de gran utilidad a la hora de implementar el sistema, facilita la organización del trabajo y lo hace más entendible a los desarrolladores. Está compuesto por los diagramas de despliegue y componentes.

3.2 Mecanismos de Implementación

- El sistema se desarrolla bajo la arquitectura Modelo-Vista-Controlador.
- La parte cliente se implementará haciendo uso generalizado del framework de desarrollo ExtJS versión 3.3.1, para las interfaces de usuario.
- La parte del servidor se implementa en el lenguaje de programación PHP5, utilizando Symfony 1.4.8 como framework que se enfoca al desarrollo en dicho lenguaje.
- Todas las librerías y clases de apoyo implementadas están incluidas en la carpeta /lib del proyecto, haciendo una estructura de carpetas lógica con previa aprobación por el equipo de arquitectura.
- El estándar de codificación será el propuesto por el IDE NetBeans 7.0 para PHP.
- La capa del modelo (acceso a datos) se genera utilizando Propel como ORM.
- El intercambio de información entre el cliente y el servidor se hace únicamente en formato JSON.

3.1.1. Diagrama de componentes

Los diagramas de componentes son usados para estructurar el modelo de implementación en términos de subsistemas de implementación y mostrar las relaciones entre sus elementos. Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes de software, sean éstos componentes de código fuente, binarios o ejecutables. Desde el punto de vista del

diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo. Los elementos de modelado dentro de un diagrama de componentes serán componentes y paquetes. A continuación se muestra el Diagrama de Componente (ver Figura 11) para la solución propuesta:

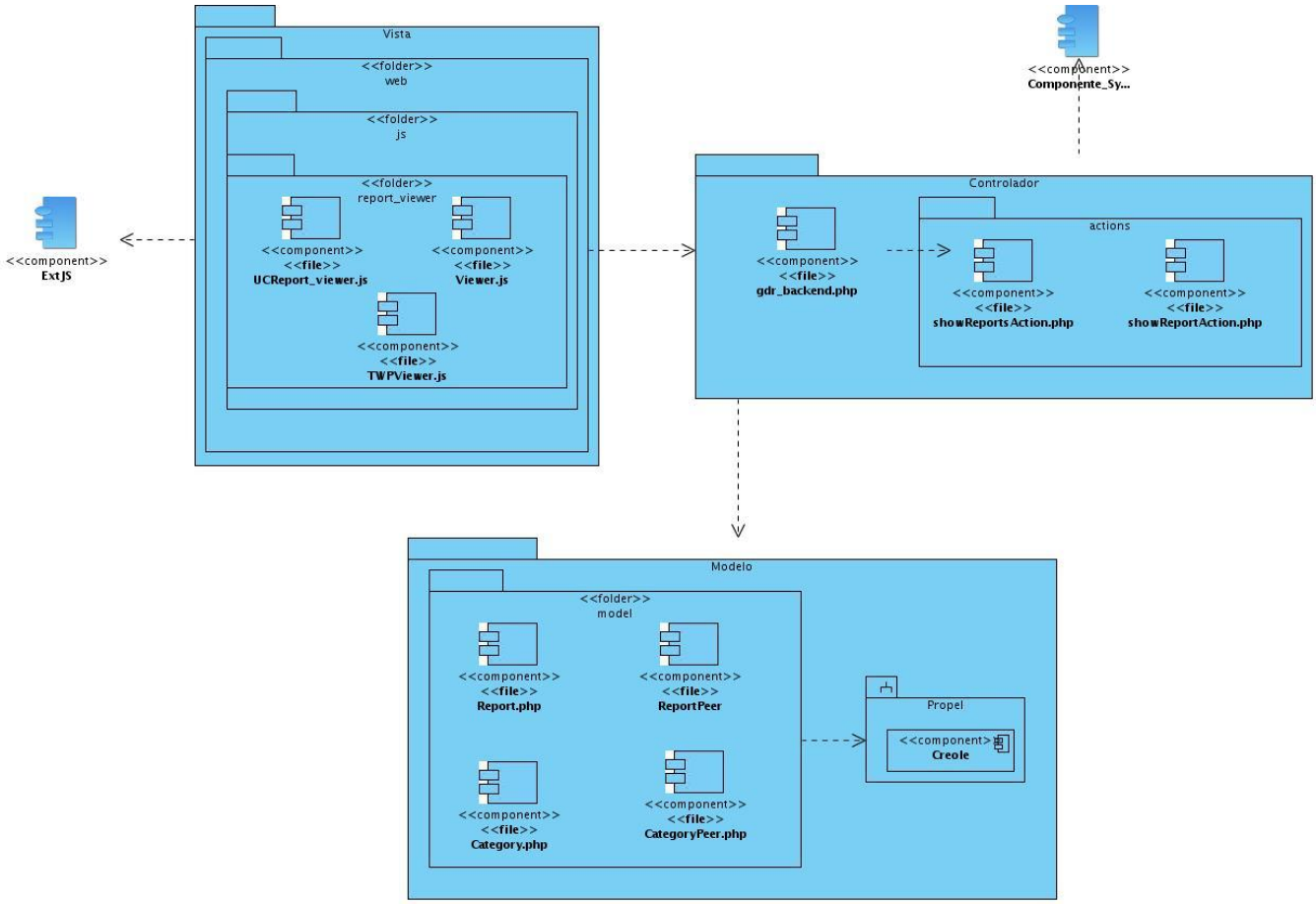


Figura 11: Diagrama de Componentes del CU Visualizar Reportes.

En el diagrama representado se observa el patrón Modelo-Vista-Controlado (MVC). La capa Vista contiene todos los archivos de tipos javascript, css y las imágenes que responden a la interfaz de la aplicación para este caso de uso. Los componentes del Modelo corresponden a las 4 clases que son generadas por el subsistema Propel por cada tabla de la base de datos de la aplicación del sistema. También se muestran los paquetes de componentes ExtJs y Symfony, necesarios para el diseño de interfaces y para manejar las peticiones del usuario respectivamente.

3.1.2. Vista de despliegue

Esta vista constituye la visión física del sistema, donde se describe este como un conjunto de nodos y sus conexiones y se establece la comunicación entren ellos a través de protocolos o enlaces.

Diagrama de despliegue

Un diagrama de despliegue es un tipo de diagrama UML con el cual queda representado un modelado del hardware utilizado en la elaboración del software. En la figura se muestran los dispositivos necesarios para la utilización del sistema, es decir, desde dónde se encuentra instalado, dónde se puede utilizar, hasta dónde se conecta para tomar los datos (ver Figura 12).

Se necesitan dos servidores web, uno como entorno de ejecución con Apache2 con soporte para PHP 5.3.3 y sistema operativo basado en GNU/Linux y el otro servidor como entorno de ejecución con Apache Tomcat para la instalación de JasperReports 4.0, esto es debido a que la aplicación está desarrollada en PHP pero la librería para la generación de reportes utilizada(JasperReports) está desarrollada en Java por lo que se hace necesario vincular PHP/Java mediante servicios utilizando SOAP como protocolo de comunicación entre ambos servidores para el intercambio de información. La base de datos puede estar instalada en el mismo servidor Apache2 o en uno distinto en dependencia de las posibilidades de la organización.

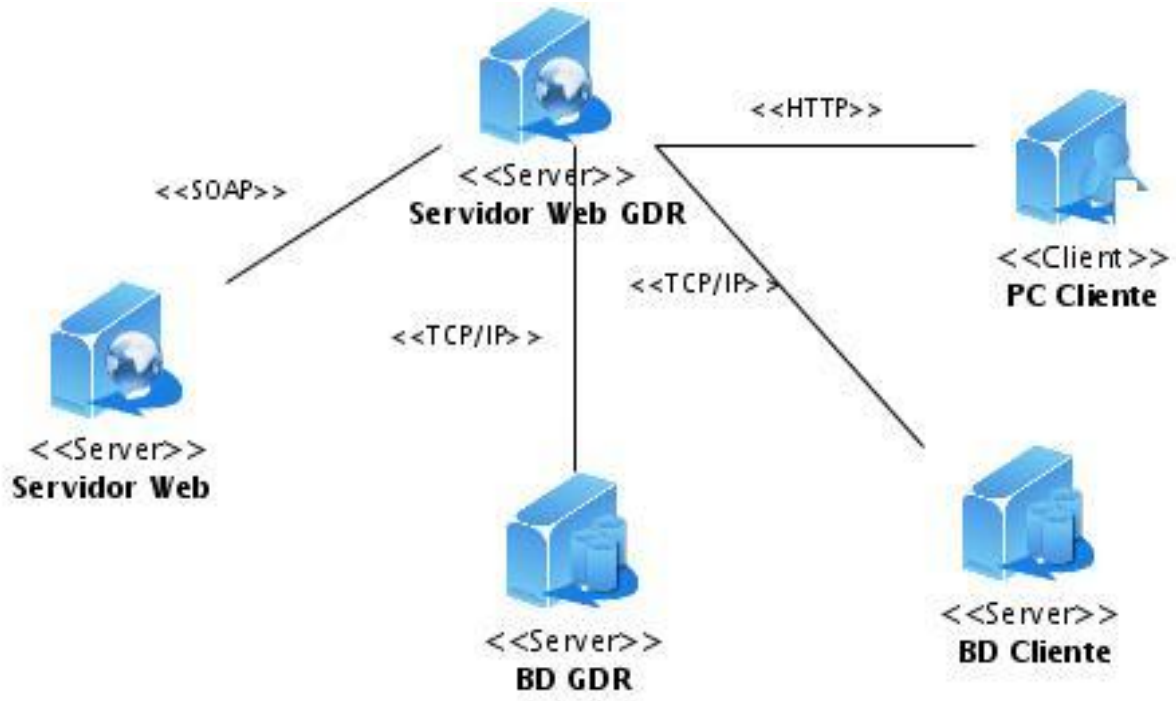


Figura 12: Diagrama de despliegue.

3.2. Código fuente

El código fuente de un programa informático (o software) es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa. Por tanto, en el código fuente de un programa está descrito por completo su funcionamiento. Este bloque de texto está escrito según las reglas sintácticas de algún lenguaje de programación destinado a ser legible por humanos.

3.2.1. Ejemplos de código fuente.

La imagen muestra el código para mostrar los reportes (ver Figura 13), este permite que el usuario pueda ver todos las reportes existentes en el sistema agrupados por categorías, cada vez que se adiciona un reporte se actualiza la lista de reportes existentes por tanto se ejecuta este método.

.

```
class showReportsAction extends sfAction {
    public function execute($request) {
        $id = $request->getParameter('node');
        $actions = explode('#', $id);
        $index = count($actions) - 1;
        $category = $actions[$index - 1];
        $option = $actions[$index];

        switch ($option) {
            case 'first':

                $categories = $this->Categories();
                $reports = $this->reportsByCategory(NULL);
                $result = array_merge($categories, $reports);
                return $this->renderText(json_encode($result));

            case 'c':

                $categories = $this->Categories(false, $category);
                $reports = $this->reportsByCategory($category);
                $result = array_merge($categories, $reports);
                return $this->renderText(json_encode($result));
            break;
```

Figura 13: Acción que muestra el listado de reportes existentes.

La imagen representa el código para la interfaz donde se muestra el listado de reportes existentes. Se evidencia el uso del componente TreePanel.(Figura 14)

```
if (typeof(rv_category) == "undefined")
    rv_category = '#first';
this.treeview = new Ext.tree.TreePanel({
    title: patdsi.gdr.report_viewer.TWPViewer.ADMINISTRATOR_PANEL_TITLE,
    region:'west',
    layout: 'fit',
    width: "25%",
    id : 'treeReport',
    collapsible: true,
    split: true,
    tbar:getSearchToolBar(searchReports,refreshSearchReport,'idM-panel', 'btSearchReports'),
    ddGroup : 'organizerDD',
   rootVisible: false,
    root: new Ext.tree.AsyncTreeNode({
        text : 'Reportes',
        singleClickExpand : true,
        expanded : true,
        id : rv_category
    }),
    loader : new Ext.tree.TreeLoader({
        dataUrl : '/gdr_backend.php/report_viewer/showReports',
        clearOnLoad:false,
   listeners : {
        beforeload : function(loader, node) {
             loadingViewer = true;
             loadreports = new Ext.LoadMask('treeReport', {
                msg : "Cargando...",
                removeMask : true
            });
            loadreports.show();
        },


              load : function(loader, node) {
                    loadreports.hide();
                    loadingViewer = false;
                searchReports();
              }}}),
             autoScroll: true
         });

            this.panel = new Ext.Panel({
             title: 'Reportes y parametros',
             region:'west',
             layout: 'accordion',
             width: "25%",
            collapsible: true,
             split: true,
             items:[this.treeview]

         });
```

Figura 14: Código para la interfaz donde se muestra el listado de reportes.

3.2.2. Interfaces de la aplicación

El diseño de la interfaz de la aplicación se encaminó a lograr interfaces agradables, sencillas y atractivas mediante la utilización del framework de presentación ExtJS.

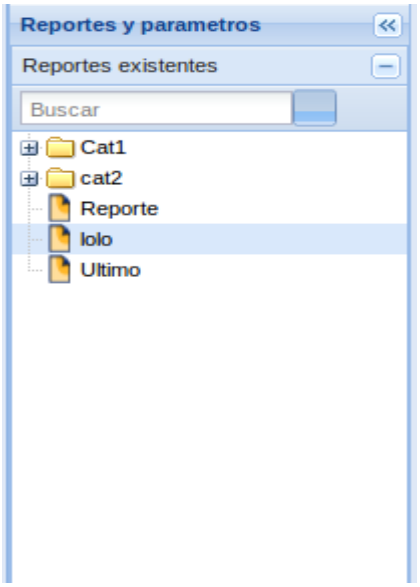


Figura 15: Mostrar listado de reportes existentes.

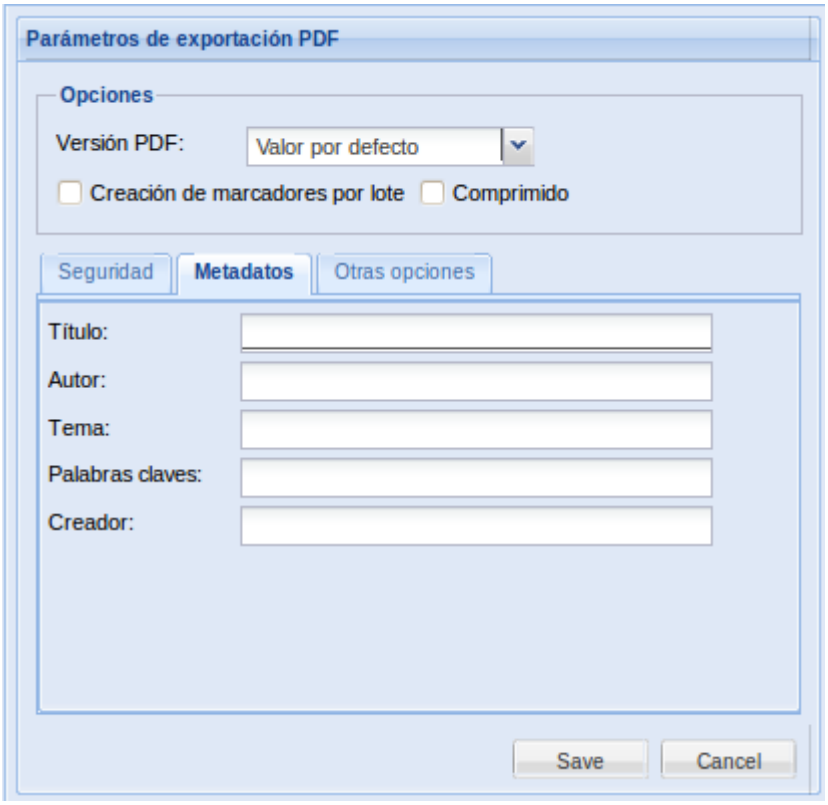


Figura 16: Configurar parámetros para la exportación a PDF.

3.3. Prueba

Las pruebas o procesos de validación son las actividades que garantizan la excelencia del desempeño de un programa. Son utilizadas para identificar posibles fallos de implementación o usabilidad. Involucran las operaciones del sistema bajo condiciones controladas y evaluando los resultados. Las pruebas pueden intencionalmente esforzar al programa y producir errores en las respuestas para determinar si los eventos ocurren cuando no tendrían que ocurrir o viceversa.

Existe una extensa variedad de técnicas para encontrar problemas en un programa. Estas van desde el uso del ingenio por parte del personal encargado de realizar las pruebas hasta el uso de herramientas automatizadas que contribuyen a aliviar el peso y el costo de tiempo de esta actividad. Según Pressman las pruebas del software son un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación.

Tipos de Pruebas

A continuación se describen algunos de los principales tipos de pruebas que pueden ser aplicados a un software:

Pruebas unitarias

Una prueba unitaria es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente. Cada pruebas deberá ser lo más independiente posible de las demás. La idea es escribir casos de prueba para cada función no trivial o método en el módulo de forma que cada caso sea independiente del resto. Esto último es la esencia de una prueba unitaria: se prueba al componente de forma aislada a todos los demás.

Pruebas funcionales

Se conocen como pruebas funcionales a aquellas cuyo objetivo es validar cuando el comportamiento observado del software probado cumple o no con sus especificaciones. Las pruebas funcionales toman el punto de vista del usuario.

Pruebas de aceptación

Son pruebas funcionales, pero vistas directamente desde el cliente. Son aquellas pruebas que demuestran al cliente que el sistema está terminado y funciona correctamente.

Pruebas de regresión

Son una estrategia de prueba en la cual las pruebas que se han ejecutado anteriormente se vuelven a realizar en la nueva versión modificada, para asegurar la calidad después de añadir las nuevas funcionalidades. El propósito de estas pruebas es asegurar que los defectos identificados en la ejecución anterior de la prueba se han corregido y que los cambios realizados no han introducido nuevos defectos o reintroducido defectos anteriores. (28)

Las pruebas que se realizarán son pruebas funcionales que tienen como objetivo asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos y obtención de resultados. Este tipo de prueba utiliza el método de Caja Negra el cual permite demostrar que las funcionalidades de un software son operativas. Básicamente el enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y en los de salida, esto generalmente se define en los casos de prueba diseñados antes del inicio de las pruebas.

Las pruebas de Caja Negra permiten encontrar:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las Bases de Datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación. (29)

3.3.1. Casos de prueba

Para desarrollar software de calidad y libre de errores, es de suma importancia la realización de los casos de prueba. Se entiende por caso de prueba, según la Ingeniería del software, al conjunto de condiciones o variables bajo las cuáles el analista determinará si el requisito de una aplicación es parcial o completamente satisfactorio. Un caso de prueba bien diseñado tiene gran posibilidad de llegar a resultados más fiables y eficientes, mejorar el rendimiento del sistema, y reducir los costos.

A continuación se presenta el caso de prueba diseñado para el caso de uso Visualizar Reportes.

Tabla 4: Secciones a probar en el Caso de Uso Visualizar reportes.

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Listar reportes existentes.	EC 1.1: Listar reportes.	Se muestra el listado de reportes existentes agrupados por categorías.
SC2: Buscar reportes.	EC 2.1: Buscar reportes introduciendo como criterio de búsqueda el nombre de un reporte existente en la BD. EC 2.2: Buscar reportes dejando en blanco el campo de texto donde se introduce el criterio de búsqueda. EC 2.3: Buscar reportes introduciendo como criterio de búsqueda el nombre de un reporte que no existe en la BD.	Se reduce la lista de reportes en concordancia con el nombre introducido.
SC3: Visualizar reportes.	EC3.1: Visualizar reportes.	Se muestran los datos correspondientes al reporte seleccionado.

Tabla 5: Tabla 3: Caso de prueba para la SC Buscar reportes.

Escenario	Descripción	Criterio	Respuesta del sistema.	Flujo central
EC 2.1: Buscar reportes introduciendo como criterio de búsqueda el nombre de un reporte existente en la BD.	Se realiza la búsqueda para el caso donde el criterio de búsqueda coincide con al menos un reporte de existentes en la BD.	V (Nombre de un reporte existente en la BD.)	El sistema muestra todos los reportes que coinciden con el criterio de búsqueda insertado	1-Se introduce el nombre del reporte que se desea buscar. 2-El sistema reduce la lista de reportes en concordancia con el nombre introducido.
EC 2.2: Buscar reportes dejando en blanco el campo de texto donde se introduce el criterio de búsqueda.	Se realiza la búsqueda para el caso donde el criterio de búsqueda está en blanco.	N/A	El sistema muestra, la lista completa de reportes, incluyendo aquellos reportes que se han creado mientras era usado el Visor.	1-Se deja el campo de texto donde se introduce el criterio de búsqueda en blanco. 2-Se presiona el botón aceptar. 3-Se muestra, la lista completa de reportes, incluyendo aquellos reportes que se han creado mientras era usado el Visor.

EC 2.3: Buscar reportes introduciendo como criterio de búsqueda el nombre de un reporte que no existe en la BD.	Se realiza la búsqueda para el caso donde el criterio de búsqueda no coincide con los reportes existentes en la BD.	I (Nombre de un reporte no existente en la BD.)	El sistema muestra una lista vacía o las carpetas de las categorías vacías en caso de que estas existan.	1-Se introduce el nombre no válido del reporte que se desea buscar. 2-Se presiona el botón buscar. 3- Se muestra una lista vacía o las carpetas de las categorías vacías en caso de que estas existan.

Tabla 6: Tabla 2: Descripción de la variable.

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Criterio	Campo de texto	Si	Debe ser una serie de caracteres que coincidan con el nombre de un reporte.

Al aplicar las pruebas funcionales al módulo se encontraron 6 no conformidades significativas, las cuales fueron resueltas en la 3ra iteración, culminando el presente trabajo con resultados satisfactorios.

3.4. Conclusiones del capítulo

En el capítulo se abarcaron los temas de implementación y prueba. Se obtuvo como resultado la confección de los diagramas de componentes y sus descripciones, así como el diagrama de despliegue, logrando mostrar las organizaciones y dependencias lógicas entre los componentes del sistema desarrollado. También se hace referencia a las pruebas en particular a las pruebas funcionales.

CONCLUSIONES

Una vez culminado el trabajo es posible afirmar que se han cumplimido los objetivos trazados para el mismo:

- Se realizó el análisis de las herramientas, metodologías y tecnologías existentes para el desarrollo del software, que arrojó como resultado la selección de OpenUp como metodología de desarrollo, JavaScript y PHP5 como lenguajes de programación del lado del cliente y del servidor respectivamente, utilizando como IDE Netbeans.
- Partiendo de las funcionalidades definidas se realizó el análisis y diseño del módulo Visor de Reportes, obteniéndose como resultado los artefactos necesarios para dar inicio a su implementación: diagramas de clases del diseño, diagrama de secuencia, diagrama de componentes y diagrama de despliegue.
- En el proceso de validación se realizaron pruebas funcionales, que demostraron que el sistema cumple satisfactoriamente con los requisitos.
- Como resultado final, se obtuvo el módulo Visor Reportes que posibilita la visualización y la configuración para la exportación de los reportes a un mayor número de formatos.

.

RECOMENDACIONES

Luego del cumplimiento de los objetivos trazados, se recomienda:

- Incorporar al módulo la funcionalidad que permita crear informes que contengan un conjunto de reportes.

REFERENCIAS BIBLIOGRÁFICAS

1. GestioPolis. *Los sistemas integrados de gestión de la información.* [En línea] [Citado el: 7 de Noviembre de 2011.] http://www.gestiopolis.com/administracion-estrategia-2/sistemas-integrados-gestion-informacion.htm.

2. **Centro de Tecnologías y Gestión de Datos.** *Caracterización del centro.*

3. Definicion.de. *Definicion de reporte.* [En línea] [Citado el: 7 de Noviembre de 2011.] http://definicion.de/reporte/.

4. Definición ABC. [En línea] [Citado el: 7 de Noviembre de 2011.] http://www.definicionabc.com/comunicacion/reporte.php.

5. Software De Computo. [En línea] [Citado el: 8 de Noviembre de 2011.] http://www.mitecnologico.com/Main/SoftwareDeComputo.

6. **Olivares, José Rolando Lafaurie.** *Sistema para la generación de reportes en la plataforma alasGRATO: Desarrollo delmódulo Reportador.* Universidad de las Ciencias Informáticas, La Habana : s.n., 2008.

7. **Marrero, Ernesto Leiva.** *Implementación del módulo de diseño de reportes para el SCADA Guardián del ALBA.* Ciudad de la Habana : s.n., 2009.

8. **Rangel, Eustaquio.** *PHPReports Manual.* Brasil : s.n., 2006.

9. geproin. Consultoría, análisis y programación en la plataforma de desarrollo de aplicaciones empresariales Velneo V7. *Aplicación de la semana en Velneo.* [En línea] [Citado el: 12 de noviembre de 2011.] http://www.geproin.es/aplicacion-semana-velneo-adinfrep/.

10. ComponenteSource. [En línea] [Citado el: 2 de Octubre de 2011.] http://www.componentsource.com/products/activereports-6/summary-es.html..

11. Microsoft SQL Server 2005 Reporting Services. [En línea] [Citado el: 2 de Octubre de 2011.] http://justindeveloper.wordpress.com/2008/10/20/microsoft-sql-server-2005-reporting-services-part-i/.

12. Presentación de Reporting Services. [En línea] [Citado el: 3 de Octubre de 2011.] http://msdn.microsoft.com/es-es/library/ms155786%28v=sql.90%29.aspx.

13. Definicion.org. *Definicion de lenguaje de programación.* [En línea] [Citado el: 7 de noviembre de 2011.] http://www.definicion.org/lenguaje-de-programacion.

14. **Mariño, Carlos Vázquez.** *Programación en PHP5. Nivel Básico.* septiembre de 2008.

15. **Pacheco Escribano, Lorenzo Fernandez.** Ejercicios prácticos tutorizados. [En línea] [Citado el: 15 de Noviembre de 2011.] books.google.com.

16. **Potencier, Fabien y Zaninotto, Francois.** librosweb. *Symfony, la guía definitiva.* [En línea] [Citado el: 25 de noviembre de 2011.] http://www.librosweb.es./symfony_1_2/.

17. EcuRed. *Ext_JS.* [En línea] [Citado el: 18 de noviembre de 2011.] http://www.ecured.cu/index.php/Sencha_Ext_JS.

18. **Alvarez, Miguel Angel.** maestros del web. *Evaluando Zend Studio.* [En línea] [Citado el: 28 de noviembre de 2011.] http://www.maestrosdelweb.com/editorial/zendstudio/.

19. EcuRed. *Zend Studio.* [En línea] [Citado el: 28 de noviembre de 2011.] http://www.ecured.cu/index.php/Zend_Studio.

20. Netbeans. [En línea] [Citado el: 17 de noviembre de 2011.] http://netbeans.org.

21. Modelado de Sistemas com UML. *Popkin Software and Systems.* [En línea] [Citado el: 5 de Diciembre de 2011.] http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/doc-modelado-sistemas-uml.pdf..

22. EcuRed. [En línea] [Citado el: 5 de Diciembre de 2011.]

23. **Duarte, Ailin Orjuela y Rojas, Mauricio.** *Las Metodologías para Desarrollo Ágil como una Oportunidad para la Ingeniería de Software Educativo.* Colombia : s.n., Mayo de 2008.

24. *Metodologías de Desarrollo de software.* [En línea] [Citado el: 13 de noviembre de 2011.] http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema04.pdf..

25. Scribd. [En línea] septiembre de 2009. [Citado el: 19 de noviembre de 2011.] http://es.scribd.com/doc/297224/RUP.

26. **Flores, Carmina Lizeth Torres y Salinas, Germán Harvey Alférez.** *Establecimiento de una Metodología de Desarrollo de Software.* abril de 2008.

27. **Burbeck, Steve.** *Application programming in Smalltalk-80 : How to use Model-View-Controller (MVC).* 1992.

28. Tipos de Pruebas . [En línea] [Citado el: 5 de Mayo de 2012.]

29. **Pressman, Roger S.** *Ingeniería de software. Un enfoque práctico.* s.l. : 6ta Edición.

BIBLIOGRAFÍA

1. GestioPolis. *Los sistemas integrados de gestión de la información.* [En línea] [Citado el: 7 de Noviembre de 2011.] http://www.gestiopolis.com/administracion-estrategia-2/sistemas-integrados-gestion-informacion.htm.

2. **Centro de Tecnologías y Gestión de Datos.** *Caracterización del centro.*

3. Definicion.de. *Definicion de reporte.* [En línea] [Citado el: 7 de Noviembre de 2011.] http://definicion.de/reporte/.

4. Definición ABC. [En línea] [Citado el: 7 de Noviembre de 2011.] http://www.definicionabc.com/comunicacion/reporte.php.

5. Software De Computo. [En línea] [Citado el: 8 de Noviembre de 2011.] http://www.mitecnologico.com/Main/SoftwareDeComputo.

6. **Olivares, José Rolando Lafaurie.** *Sistema para la generación de reportes en la plataforma alasGRATO: Desarrollo delmódulo Reportador.* Universidad de las Ciencias Informáticas, La Habana : s.n., 2008.

7. **Marrero, Ernesto Leiva.** *Implementación del módulo de diseño de reportes para el SCADA Guardián del ALBA.* Ciudad de la Habana : s.n., 2009.

8. **Rangel, Eustaquio.** *PHPReports Manual.* Brasil : s.n., 2006.

9. geproin. Consultoría, análisis y programación en la plataforma de desarrollo de aplicaciones empresariales Velneo V7. *Aplicación de la semana en Velneo.* [En línea] [Citado el: 12 de noviembre de 2011.] http://www.geproin.es/aplicacion-semana-velneo-adinfrep/.

10. ComponenteSource. [En línea] [Citado el: 2 de Octubre de 2011.] http://www.componentsource.com/products/activereports-6/summary-es.html..

11. Microsoft SQL Server 2005 Reporting Services. [En línea] [Citado el: 2 de Octubre de 2011.] http://justindeveloper.wordpress.com/2008/10/20/microsoft-sql-server-2005-reporting-services-part-i/.

12. Presentación de Reporting Services. [En línea] [Citado el: 3 de Octubre de 2011.] http://msdn.microsoft.com/es-es/library/ms155786%28v=sql.90%29.aspx.

13. Definicion.org. *Definicion de lenguaje de programación.* [En línea] [Citado el: 7 de noviembre de 2011.] http://www.definicion.org/lenguaje-de-programacion.

14. **Mariño, Carlos Vázquez.** *Programación en PHP5. Nivel Básico.* septiembre de 2008.

15. **Pacheco Escribano, Lorenzo Fernandez.** Ejercicios prácticos tutorizados. [En línea] [Citado el: 15 de Noviembre de 2011.] books.google.com.

16. **Potencier, Fabien y Zaninotto, Francois.** librosweb. *Symfony, la guía definitiva.* [En línea] [Citado el: 25 de noviembre de 2011.] http://www.librosweb.es./symfony_1_2/.

17. EcuRed. *Ext_JS.* [En línea] [Citado el: 18 de noviembre de 2011.] http://www.ecured.cu/index.php/Sencha_Ext_JS.

18. **Alvarez, Miguel Angel.** maestros del web. *Evaluando Zend Studio.* [En línea] [Citado el: 28 de noviembre de 2011.] http://www.maestrosdelweb.com/editorial/zendstudio/.

19. EcuRed. *Zend Studio.* [En línea] [Citado el: 28 de noviembre de 2011.] http://www.ecured.cu/index.php/Zend_Studio.

20. Netbeans. [En línea] [Citado el: 17 de noviembre de 2011.] http://netbeans.org.

21. Modelado de Sistemas com UML. *Popkin Software and Systems.* [En línea] [Citado el: 5 de Diciembre de 2011.] http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/doc-modelado-sistemas-uml.pdf..

22. EcuRed. [En línea] [Citado el: 5 de Diciembre de 2011.]

23. **Duarte, Ailin Orjuela y Rojas, Mauricio.** *Las Metodologías para Desarrollo Ágil como una Oportunidad para la Ingeniería de Software Educativo.* Colombia : s.n., Mayo de 2008.

24. *Metodologías de Desarrollo de software.* [En línea] [Citado el: 13 de noviembre de 2011.] http://alarcos.inf-cr.uclm.es/doc/ISOFTWAREI/Tema04.pdf..

25. Scribd. [En línea] septiembre de 2009. [Citado el: 19 de noviembre de 2011.] http://es.scribd.com/doc/297224/RUP.

26. **Flores, Carmina Lizeth Torres y Salinas, Germán Harvey Alférez.** *Establecimiento de una Metodología de Desarrollo de Software.* abril de 2008.

27. **Burbeck, Steve.** *Application programming in Smalltalk-80 : How to use Model-View-Controller (MVC).* 1992.

28. Tipos de Pruebas . [En línea] [Citado el: 5 de Mayo de 2012.]

29. **Pressman, Roger S.** *Ingeniería de software. Un enfoque práctico.* s.l. : 6ta Edición.

30. **Duarte, Ailin Orjuela y Rojas, Mauricio.** Las Metodologías para Desarrollo Ágil como una Oportunidad para la Ingeniería de Software Educativo. Colombia : s.n., Mayo de 2008.

31. **Heffelfinger, David R.** JasperReports 3.5 for Java Developers. s.l. : Gagandeep Singh, Agosto 2009.

32. **Danciu, Teodor y Chirita, Lucian**. *The Definitive Guide to JasperReports*™. 2007.

33. **Rovelo, Efren A.** symfony. [En línea]. http://www.symfony-project.org/.

34. **Eguíluz Pérez, Javier**. Introducción a JavaScript. [En línea] 2008. http://www.librosweb.es.

35. **Murua Olalde, Juan**. Metodologías de desarrollo de software. [En línea] 12 de Noviembre de 2011.

36. Lenguajes de la Web. [En línea] [Citado el: 8 de noviembre de 2011.] http://www.maestrosdelweb.com/principiantes/los-diferentes-lenguajes-de-programacion-para-la-web/.

37. **Jamir Antonio Avila Mojica**. *Introducción a patrones de software*.

38. **Prof. Ivette C. Martínez**. Diseño con Patrones. Universidad Simón Bolívar. Ing. de Software : s.n.

39. **Pressman.** Aspectos y metricas de la calidad del Software. 1998.

40 **Vanina Barotto;Mauricio Demonte**. Definición de patrones de diseño a través del metamodelo UML Río Cuarto : s.n., 2005.

41. **Pressman, Roger S**.; 2007. “Ingeniería de software. Un enfoque práctico”. 6ta Edición. Parte I.

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

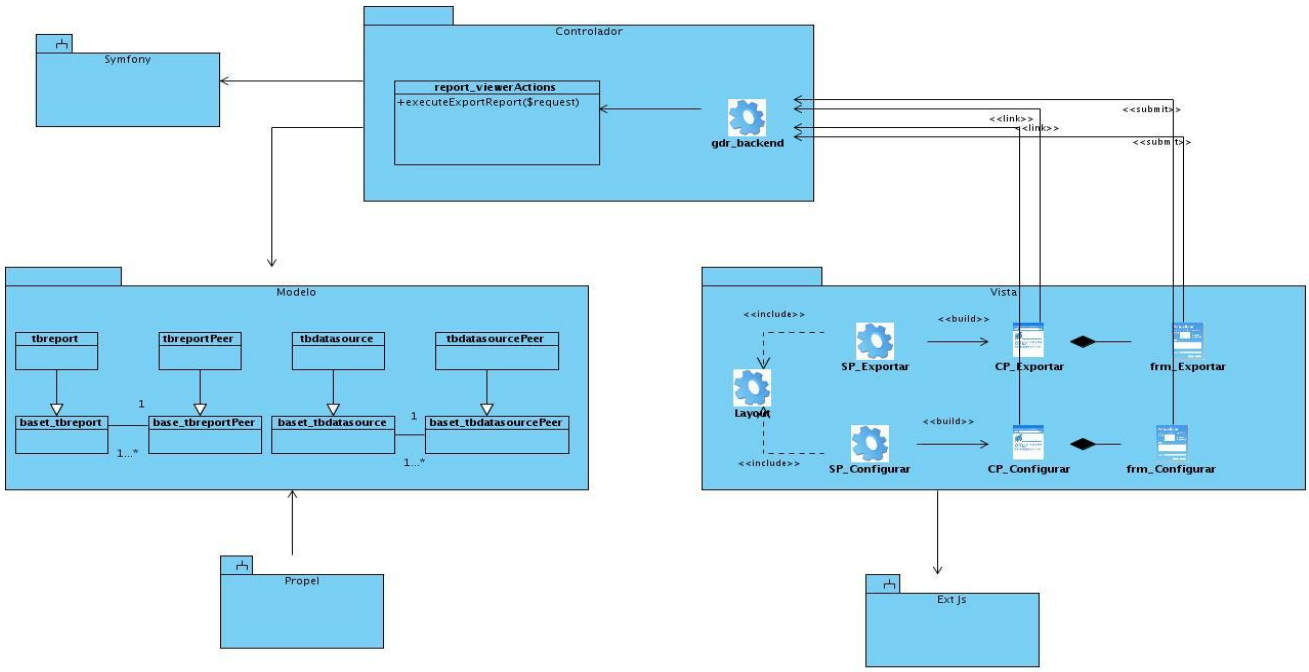
43. **Rodríguez, Jorge**. Pruebas Unitarias.

44. **Luis Artola** . Tipos de pruebas automatizadas de software . [En línea] 2009. http://www.programania.net/diseno-de-software/tipos-de-pruebas-automatizadas-de-software/.

45. **Popkin** Software and Systems. Modelado de Sistemas com UML. http://es.tldp.org/Tutoriales/doc-modelado-sistemas-UML/doc-modelado-sistemas-uml.pdf.

ANEXOS

Anexo 1: Diagrama de Clases del Diseño del Caso de Uso Exportar Reportes.



Anexo 2: Especificación del Caso de Uso Exportar Reportes.

Objetivo	Configurar los parámetros para la exportación a los diferentes formatos y exportar los reportes.
Actores	Consultante del reporte.
Resumen	El caso de uso se inicia al seleccionar un reporte, permite exportar dicho reporte a diferentes formatos. Además brinda la posibilidad de configurar los parámetros para la exportación. Esos parámetros pueden ser: salida de datos por cantidad de páginas, algunas opciones de seguridad, entre otros. El caso de uso finaliza al ser exportado un reporte.
Complejidad	Media
Prioridad	Media
Precondiciones	Debe existir al menos un reporte Debe estar seleccionado un reporte
Postcondiciones	El reporte queda exportado a un formato (HTML, PDF, CSV, EXCEL, etc).

Flujo de eventos		
Flujo básico <Nombre del flujo básico>		
	Actor	**Sistema**
1.		El sistema visualiza la interfaz para la exportación Permite la acción de: - Configurar los parámetros de exportación a los diferentes formatos, ver sección 1: "Opciones de exportación."
2.	Selecciona la opción:"Formato".	
3.		Muestra los formatos disponibles.
4.	Selecciona un formato	
5.		Muestra el formato seleccionado.
6.	Introduce el rango de datos a visualizar	
7.		Actualiza la vista previa del reporte con los datos de las filas que se encuentran en el rango introducido. Si el rango introducido no es válido ver Flujo Alterno 1
8.	Selecciona la opción:"Exportar".	
9.		Construye el reporte seleccionado en el formato especificado.
10.		Exporta el reporte seleccionado.
Flujos alternos		
Nº Evento < >		
	Actor	**Sistema**
1.		7.1 Muestra un mensaje informando que el rango introducido no es válido.
Sección 1: "Opciones de exportación"		
Flujo básico <Nombre del flujo básico>		
	Actor	**Sistema**
1.	Selecciona la pestaña "Opciones de	

	Exportación".	
2.		El sistema muestra los formatos existentes y los parámetros comunes a dichos formatos.
3.	Selecciona el formato deseado	
4.		El sistema muestra los parámetros asociados al formato seleccionado configurados por defecto.
5.	Configura los parámetros según desee.	
6.		El sistema muestra la nueva configuración de los parámetros. En caso de presionar la opción "Cancelar", ver Flujo Alterno 1 Permite: Seleccionar otro formato, ver paso 3 del Flujo Normal de Eventos.
7.	Selecciona la opción aceptar	
8.		Guarda las nuevas configuraciones y envía un mensaje indicando que se ha guardado la nueva configuración.
9.		Termina el caso de uso.
Flujos alternos		
Nº Evento <Condición que dio lugar a la extensión>		
	Actor	**Sistema**
1.		6.1 Se restablecen las configuraciones predeterminadas de los parámetros.
2.		

Relaciones	**CU Incluidos**	
	CU Extendidos	
Requisitos no funcionales		

GLOSARIO DE TÉRMINOS

API: (*Application Programming Interface* - Interfaz de Programación de Aplicaciones) es un conjunto de convenciones internacionales que definen cómo debe invocarse una determinada función de un programa desde una aplicación.

BD: Base de datos.

CASE: Ingeniería de Software Asistida por Computación.

CU: Caso de uso.

Framework: Marco de trabajo.

GDR: Generador dinámico de reportes.

GOF: *Gang of Four*.

GPL: Licencias Open Source.

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

IDE: Entorno desarrollo integrado**.**

JSON: acrónimo de *JavaScript Object Notation* (Notación de Objetos de JavaScript), es un formato ligero para el intercambio de datos.

MVC: Modelo – Vista – Controlador.

Multiplataforma: Es un término usado para referirse a los programas, sistemas operativos, lenguajes de programación, u otra clase de software, que puedan funcionar en diversas plataformas.

OpenUp: Metodología para el proceso del desarrollo del software.

ORM: Acrónimo de *Object-Relational Mapping* (Mapeo Objeto-Relacional), es una técnica de programación para convertir datos entre el sistema de tipos utilizado en un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional.

PDF: Acrónimo del inglés *Portable Document Format* (Formato de Documento Portátil), es un formato de almacenamiento de documentos, desarrollado por la empresa Adobe Systems,

POO: Programación orientada a objetos.

RTF: acrónimo de *Rich Text Format* (Formato de Texto Enriquecido), es un formato de archivo informático desarrollado por Microsoft en 1987 para el intercambio de documentos multiplataforma.

SOAP:. *Simple Object Access Protocol*. Es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

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

UI: Interfaces de usuario.

YUI: es una serie de bibliotecas escritas en JavaScript, para la construcción de aplicaciones interactivas (RIA). Dichas bibliotecas son utilizadas para el desarrollo web específicamente para ser usadas como la programación de aplicaciones de escritorio, con componentes vistosos y personalizables y con una amplia implementación con AJAX.