Universidad de las Ciencias Informáticas

Facultad 6

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Título:

Generación de reportes a partir de fuentes de datos XML, JSON y CSV para GDRv2.0

Autor:

Leonardo Javier Nieves González

Tutores:

MSc. Yunet González Mulet

Ing. Fernando Henríquez Amaya

La Habana, junio de 2015

"Ano 57 de la Revolución"

Declaración de autoría

Declaro ser el único autor 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 _____.

Leonardo Javier Nieves González

_____

Firma del Autor

MSc. Yunet González Mulet

_____

Firma del Tutor

Ing. Fernando Henríquez Amaya

_____

Firma del Tutor

Datos de contacto

Tutora: MSc. Yunet González Mulet.

Máster en Bioinformática con más de 7 años de experiencia en esta rama. Profesora de Inteligencia Artificial del Departamento de Técnicas de Programación de la facultad 6.

Formación Académica: Ingeniera en Ciencias Informáticas (2007) y Máster en Bioinformática (2010)

Centro Laboral: Universidad de las Ciencias Informáticas (UCI).

Email: ygonzalezmu@uci.cu

Tutor: Ing. Fernando Henríquez Amaya

Formación Académica: Ingeniero en Ciencias Informáticas, graduado en el año 2013 en la Universidad de Ciencias Informáticas. Cuenta con un año de experiencia trabajando en el centro DATEC de la facultad 6 de la Universidad de Ciencias Informáticas.

Centro Laboral: Universidad de las Ciencias Informáticas (UCI).

Email: fhenriquez@uci.cu

Agradecimientos

A mis padres por apoyarme siempre y dejarme tomar mis propias decisiones. A mi familia por ser tan unida, en especial mis abuelos siempre tan cariñosos y atentos.

A mis tutores Fernando y Yunet sin los cuales no hubiera podido terminar este trabajo. Gracias por la ayuda.

A Luis Carlos por siempre ayudarme en los momentos de apuros. A Rainier y Keniel por ser excelentes amigos. Le agradezco al piquete de Lázaro, Yoel, Lenadro, Cesar, Gianni, Scofee, El Tibu, "El Zeta" en fin todos los "indeseables".

Realmente cinco años de estudios se dice fácil pero es un largo camino. Gracias a todos mis compañeros de estudio por compartir conmigo esos cincos duros años.

Dedicatoria

Le dedico este logro a una persona que aunque no esté conmigo hoy siempre lo recordaré, mi hermano. Donde quiera que estés nunca te olvidaré.

A mi familia por el apoyo que siempre me han brindado e especial a mis padres Rosa y Luis siempre tan atentos. Son los mejores.

Quiero dedicar este trabajo a todas aquellas personas que tuvieron un sueño y lo intentaron cumplir sin importar las dificultades.

Resumen

Los Sistemas de Gestión de Información permiten el desarrollo de las empresas contribuyendo en el proceso de toma de decisiones. Un ejemplo de estos sistemas es el Generador Dinámico de Reportes (GDR): aplicación web desarrollada en el Centro de Tecnologías y Gestión de Datos (DATEC). Su principal característica es la posibilidad de generar o visualizar reportes a partir de un origen o fuente de datos. En su última versión 2.0 el GDR permite extraer información partir de las fuentes de datos PostgreSQL, MySQL y SQLite. Sin embargo no cuenta con otras que igualmente son utilizadas por varias entidades para el intercambio de datos. El objetivo de este trabajo se enfocó en el desarrollo de componentes que permitan el desarrollo del proceso de diseño de reportes a partir de orígenes de datos XML, CSV y JSON. Para la solución se utilizó NetBeans 8.0 como entorno de desarrollo, los lenguajes de programación PHP 5.3 y JavaScript, Symfony 2.0.18 como marco de trabajo y para el diseño de la interfaz gráfica de usuario se utilizó Ext JS 3.4. Se validaron las funcionalidades implementadas a partir de la elaboración de casos de pruebas, garantizando que la calidad de la solución fuera la requerida.

Palabras Claves

GDR, origen de datos, XML, CSV, JSON

Abstract

The Information Management Systems allow the development of enterprises contributing to the decision-making process. An example of these systems is the Dynamic Report Generator (GDR) web application developed at the Center for Technology and Data Management (DATEC). Its main feature is the ability to generate and view reports from a source or data source. In its latest version 2.0 the GDR can only extract information from sources PostgreSQL, MySQL, SQLite, Microsoft SQL Server database. However there has others that are also used by various entities to exchange data. The objective of this work focused on the development of components to enable the development of the design process reports from data sources such as XML, CSV and JSON. NetBeans 8.0 was used as a development environment for the solution, PHP 5 programming language, Symfony 2.0.18 as a framework for the design of graphical user interface Ext JS 3.4 was used. The functionality implemented from the development of validated test cases, ensuring that the quality of the solution is required.

Keywords

GDR, datasource, XML, CSV, JSON

Índice

Índice de figuras

Índice de Tablas

Introducción

Desde el surgimiento de las Tecnologías de la Información y las Comunicaciones (TICs), la gestión de la información ha sido considerada de gran importancia para las organizaciones permitiendo la optimización de los recursos, mejoras en la comunicación, sirviendo de gran ayuda para la toma de decisiones y el desarrollo de productos o servicios con mayor eficiencia y calidad. Durante la gestión de la información se lleva a cabo la obtención de conocimiento a partir del cual se genera un gran volumen de datos a procesar, dificultando el estudio y análisis de la información recopilada.

Para llevar a cabo la manipulación de la información se utiliza los Sistemas de Gestión de Información (SGI) que permiten utilizar los recursos de información tanto internos como externos, con el objetivo de servir de ayuda en el proceso de toma de decisiones y así satisfacer las necesidades propias de la organización. Una de los principales aspectos que caracteriza un SGI es la capacidad de crear informes, para ello utilizan herramientas complementarias llamadas generadores de reportes, los cuales se emplean principalmente para la Inteligencia de Negocio.

El departamento de Desarrollo de Componentes del Centro de Tecnologías y Gestión de Datos (DATEC), perteneciente a la infraestructura productiva de la Universidad de las Ciencias Informáticas (UCI), se dedica al desarrollo de sistemas informáticos encaminados a satisfacer las necesidades de gestión de la información existentes en una empresa, una institución o un país en general. Entre estos sistemas informáticos se encuentra el Generador Dinámico de Reportes (GDR) que permite obtener la información almacenada a partir de distintas fuentes de datos para luego mostrarla en forma de reportes, con el objetivo de servir de ayuda para la toma de decisiones y la realización de estudios, haciendo de esta una herramienta útil para el manejo de datos. (Brito Rodríguez, 2014)

Actualmente, se trabaja en la versión GDRv2.0 la cual contiene notables mejoras, destacándose el uso del motor de reportes JasperReports para mejorar la calidad de los informes y la exportación de reportes en formatos como HTML, PDF, CSV, XLS entre otros. Además brinda una solución al problema de generar reportes a partir de bases de datos relacionales como es el caso de PostgreSQL, MySQL, SQLite. Sin embargo, existen otros orígenes de datos no relacionales basados en el usos de archivos de texto plano como es el caso de CSV (valores separados por coma), XML (lenguaje de marcado extensible), XLS (Microsoft Excel) entre otros utilizados en la actualidad por algunos de los principales generadores de reportes del mundo como Crystal Reports, Stimulsoft Reports. La posibilidad de poder utilizar otras fuentes

de datos para la generación de reportes le otorgaría al sistema mayor relevancia y capacidad de adaptación para satisfacer las necesidades del cliente.

Teniendo en cuenta la situación antes expuesta, se determina como **problema a resolver**: ¿Cómo obtener la información contenida en fuentes de datos con formatos XML, JSON y CSV haciendo uso del Generador Dinámico de Reportes?

Se define como **objeto de estudio**: Proceso de visualización de la información, enmarcado en el **campo de acción:** Visualización de la información almacenada en fuentes de datos con formatos XML, JSON y CSV en el Generador Dinámico de Reportes v2.0.

Se define como **objetivo general**: Desarrollar componentes para la visualización de la información en el Generador Dinámico de Reportes v2.0 a partir de las fuentes de datos XML, CSV y JSON.
Se definieron como **objetivos específicos:**

1. Realizar un estudio referente al uso de los generadores de reportes en la actualidad y la obtención de la información contenida en orígenes de datos XML, CSV y JSON.
2. Realizar el análisis y diseño de los componentes necesarios para añadir fuentes de datos con formato XML, JSON y CSV y la generación de reportes a partir de los mismos.
3. Realizar la implementación de los componentes.
4. Realizar pruebas para validar la solución.

Para darle cumplimiento a los objetivos específicos se definieron las siguientes **tareas de la investigación:**

1. Revisión bibliográfica del marco conceptual referente a los principales generadores de reportes usados en el mundo
2. Revisión bibliográfica sobre las fuentes de datos XML, JSON y CSV.
3. Manipulación e interpretación de los metadatos obtenidos desde fuentes de datos XML, JSON y CSV.
4. Selección de metodología, tecnologías y herramientas a utilizar.
5. Diseño de los componentes para la conexión y generación de reportes a partir de las fuentes de datos XML, JSON y CSV.
6. Implementación de los componentes para la conexión y generación de reportes.
7. Realización de pruebas a la solución.

Para guiar la investigación se definieron las siguientes **preguntas de la investigación:**

- ✓ ¿Cuáles son las características que poseen los orígenes de datos XML, CSV y JSON que posibilita la obtención de reportes?
- ✓ ¿Cuáles son las ventajas que brinda el uso de estos orígenes de datos para la generación de reportes?
- ✓ ¿Es factible realizar la obtención de información a partir de las fuentes de datos XML, CSV y JSON?
- ✓ ¿Cómo realizar el proceso de implementación de los componentes necesarios para lograr incorporar las fuentes de datos?
- ✓ ¿Cómo medir el funcionamiento adecuado de los componentes?

Durante la investigación, se utilizaron varios métodos científicos para lograr un mayor entendimiento del tema. Los mismos se describen y clasifican a continuación:

Métodos teóricos: los métodos teóricos permiten revelar las relaciones esenciales del objeto de investigación que no son observables directamente. Posibilitan la interpretación conceptual de los datos empíricos encontrados, además de explicar los hechos y profundizar en las cualidades fundamentales de los procesos, hechos y fenómenos

1. Método de modelación: se utilizó para la creación de modelos de diseño y componentes.
2. Método Histórico-Lógico: utilizado para la realización del estudio detallado de los principales Generadores de Reportes utilizados en la actualidad.

Métodos empíricos: los métodos empíricos revelan y explican las características fundamentales del objeto a través de procedimientos prácticos.

1. Entrevistas: se establecieron diálogos con los especialistas en GDR con el fin de obtener información precisa.

El presente trabajo está compuesto por 3 capítulos, además de la Introducción, Conclusiones, Referencias Bibliográficas, Recomendaciones y Anexos.

Capítulo 1. Fundamentación Teórica: En este capítulo se realiza un estudio del estado actual de la temática a desarrollar, con el objetivo de caracterizar las herramientas y tecnologías necesarias para dar cumplimiento al objetivo de la presente investigación.

Capítulo 2. Análisis y diseño del sistema: En este capítulo se describe el proceso de análisis y diseño del sistema, haciendo uso de los artefactos pertenecientes a la metodología de trabajo OpenUP.

Capítulo 3. Implementación y prueba: En este capítulo se explica lo relacionado con la implementación del sistema, así como los elementos que conforman la estrategia de prueba aplicada a la solución.

1 Capítulo I: Fundamentación teórica

Introducción

El objetivo del presente capítulo es abordar los principales conceptos relacionados con la gestión de información a partir del uso del GDR teniendo en cuenta las fuentes u orígenes de datos. Se definen las herramientas, tecnologías y metodología de desarrollo empleadas para dar cumplimiento a los objetivos de este trabajo.

1.1 Sistemas de Gestión de Información (SGI)

Un Sistema de Gestión de Información está compuesto por un conjunto de procedimientos que permiten recopilar información de un grupo de fuentes para luego visualizarla de forma sencilla y entendible para el usuario. Su objetivo es aprovechar al máximo los recursos de la información en función de la mejora continua del proceso de la toma de decisiones organizacional. A continuación se muestran algunos de los elementos que caracterizan a los SGI:

- Capacidad para almacenar datos y facilitar información para ser recuperada por los usuarios del sistema.
- Capacidad de generar reportes que ayuden en el proceso de toma de decisiones.
- Integración.
- Escalabilidad: capacidad que tiene el sistema para modificarse en función de las necesidades de una organización. (Hamlett, 2014)

Los SGI utilizan generadores de reportes como herramientas complementarias incluidas para la visualización de la información. Para el acceso al contenido almacenado en las bases de datos utilizan un lenguaje transparente para el usuario, una vez obtenida la información es visualizada en forma de reportes. Para un mejor entendimiento se realiza un estudio de algunas de las principales herramientas para la generación de reportes usados en el mundo.

1.2 Generadores de reportes

- ✓ **JasperReports:** es una librería de creación de informes que tiene la habilidad de entregar contenido enriquecido al monitor, a la impresora o a ficheros PDF, HTML, XLS, CSV y XML. Está escrito completamente en Java y puede ser usado en gran variedad de aplicaciones de Java, incluyendo J2EE o aplicaciones web, para generar contenido dinámico. Se encuentra bajo licencia libre GNU, por lo que es Software libre. (TIBCO Software Inc., 2015)

- ✓ **Crystal Reports:** herramienta muy potente utilizada para la creación e integración de reportes con datos provenientes de múltiples fuentes de datos. Entre sus características presenta el más completo acceso a datos, incluyendo más de 30 drivers para acceso a bases de datos relacionales, fuentes de datos XML y cubos OLAP. Es considerado líder en el desarrollo de reportes ya que permite transformar rápidamente cualquier origen de datos en contenido interactivo e integrar estrechamente capacidades de diseño, modificación y visualización en aplicaciones .NET o Java. Permitiendo a los usuarios finales acceder e interactuar con los reportes a través de portales Web, dispositivos móviles y documentos de Microsoft Office. (DSCallards, 2015)
- ✓ **FastReport.Net:** Generador de informes con amplísimas posibilidades para Windows Forms, ASP.NET. Compatible con Microsoft Visual Studio 2005, 2008 y Microsoft Visual Studio 2010-2015, Delphi Prism, NET Framework 2.0. Permite la conexión a cualquier base de datos, utilizar cualquiera de sus tablas o crear consultas propias en el lenguaje SQL. Admite diversas bibliotecas para el acceso a la base de datos: FireDAC, ADO, BDE, DBX, IBX y FIBPlus, así que proporciona acceso a prácticamente todas las bases de datos, incluyendo Oracle y muchas bibliotecas de terceros. (Fast Reports Inc., 2015)
- ✓ **Stimulsoft Reports.Web:** herramienta de información diseñada para la creación y presentación de informes en la web. Permite implementar con facilidad y eficacia el ciclo completo de trabajo con informes y la automatización de las tareas de procesamiento y presentación adecuada de la información a los usuarios. Para la generación de reportes utiliza fuentes de datos como MS-SQL, Oracle, PostgreSQL, SQLite entre otros. (Stimulsoft, 2015)
- ✓ **SQL Server Reporting Services:** es una plataforma de informes basada en servidor que proporciona la funcionalidad completa de generación de reportes para una gran variedad de orígenes de datos. Incluye un conjunto completo de herramientas para la creación, administración y entrega de reportes, y las API que permiten a los desarrolladores integrar o ampliar el procesamiento de datos e informes en aplicaciones personalizadas. Las herramientas de Reporting Services funcionan en el entorno de Microsoft Visual Studio y están totalmente integradas con las herramientas y componentes de SQL Server. Con Reporting Services, puede crear informes interactivos, tabulares, gráficos o de forma libre a partir de orígenes de datos relacionales, multidimensionales o basados en XML. Los informes pueden incluir visualización de datos

avanzada, como diagramas, mapas y minigráficos. Contiene varios formatos de visualización y exporta informes a otras aplicaciones, como Microsoft Excel. (Microsoft, 2015)

- ✓ **PHP Report Maker:** es una poderosa herramienta diseñada para crear reportes dinámicos PHP desde una base de datos MySQL. Las páginas generadas son basadas en PHP puro, ningún servidor o cliente son necesarios para dicho proceso. PHP Report Maker está diseñado para trabajar con mucha facilidad ya que contiene numerosas opciones para generar los reportes de acuerdo a las necesidades del cliente. (World Technology Limited, 2015)

A partir del estudio realizado se evidencian algunas de las principales características que definen la calidad de un generador de reportes: la capacidad de integración con otros sistemas y la posibilidad de obtención de información a partir de determinado orígenes de datos. A continuación se realiza una tabla comparativa de algunos de los principales generadores de reportes teniendo en cuenta las características antes mencionadas:

Tabla 1: Principales Generadores de reportes del mundo

Nombre	Acceso a datos	Integración	Sistema Operativo
JasperReports	PostgreSQL, SQLite, Oracle, Acces, CSV, XML, JSON, JavaBeans entre otros	Se usa comúnmente con iReport, pero puede ser usado en gran variedad de aplicaciones de Java, incluyendo J2EE o aplicaciones web.	Multiplataforma
Fast Reports	Proporciona acceso a prácticamente todas las bases de datos, incluyendo Oracle y muchas bibliotecas de terceros.	Compatible con Microsoft Visual Studio 2005, 2008 y Microsoft Visual Studio 2010-2015, Delphi Prism, NET Framework 2.0	Windows
Crystal Reports	Incluye más de 30 drivers de conexión: ODBC, OLE DB y JDBC nativos a fuentes de datos relacionales, OLAP, de servicios web, XML,	Integra estrechamente capacidades de diseño, modificación y visualización en aplicaciones .NET, Java o COM.	Microsoft Windows, UNIX (Linux, Solaris, AIX, HP-UX), Mac

Stimulsoft Reports	Soporta 25 orígenes de datos entre los que se encuentran MS-SQL, Oracle, PostgreSQL, SQLite, JSON y XML.	Utilizado en aplicaciones .NET y Microsoft Visual Studio	Windows 8.1, Windows 8, Windows 7, Windows Vista, Windows XP, Windows Server 2003 y Windows 2000
PHP Reports Maker	Soporte para los SGBD MySQL, PostgreSQL, Microsoft Access, Microsoft SQL Server y Oracle	Se utiliza para aplicaciones web	Microsoft Windows y Linux
SQL Server Reporting Services	Soporta bases de datos relacionales, XML.	Funciona en el entorno de Microsoft Visual Studio y está totalmente integrado con las herramientas y componentes de SQL Server	Windows

Con los resultados mostrados en la tabla anterior se concluye que muchos de los generadores de reportes actuales utilizan como orígenes de datos archivos con formato XML, CSV y JSON con el objetivo de otorgarle al sistema mayor impacto social y brindarle al cliente nuevas alternativas para la obtención de reportes. Esto evidencia la necesidad de añadir el uso de archivos con estos formatos como fuentes de datos para GDR v2.0.

1.3 Generador Dinámico de Reportes (GDR v2.0)

El GDR es una aplicación multiplataforma que permite consultar información a partir de diferentes fuentes de datos como PostgreSQL, MySQL, SQLite. Tiene como objetivo facilitar a los usuarios abstraerse de conocimientos relacionados con los gestores de bases de datos, agilizar la toma de decisiones y generar reportes en varios formatos con gran variedad de opciones en su diseño, marcando una diferencia entre los reportes tradicionales y los reportes dinámicos. Como ventaja fundamental permite que otros sistemas puedan integrarse y consumir los reportes almacenados en su base de datos a través de peticiones web. Se encuentra en desarrollo la versión 2.0 para la cual se utiliza el lenguaje de programación PHP 5.3, el

marco de trabajo Symfony 2.0.18 y para el diseño de la interfaz gráfica de usuario se utiliza Ext JS 3.4. (Brito Rodríguez, 2014)

Aunque GDRv2.0 constituye una excelente herramienta, su capacidad de generación de reportes solo se enfoca en el uso de bases de datos relacionales, sin embargo, no son las únicas fuentes de datos que existen.

1.4 Orígenes o fuentes de datos

Un origen de datos lo constituye cualquier recurso o medio informático que permita obtener información estructurada y supervisada con el propósito de mejorar el proceso de gestión del conocimiento. Todo origen o fuente de datos se caracteriza por contener no solo un conjunto de datos específicos, sino también la información requerida para tener acceso a esos datos y la ubicación física del origen. Existen muchas fuentes de datos, sin embargo, este documento se enfoca solo en las necesarias para darle solución al problema planteado al inicio de la investigación. (Microsoft, 2015)

1.4.1 XML

XML es un lenguaje basado en SGML (Lenguaje de Marcado Generalizado Estándar), capaz de describir cualquier tipo de información en forma personalizada. XML no es un verdadero lenguaje de marcas, sino más bien un sistema o mecanismo para definir lenguajes de marcas adecuados a usos determinado.

Algunas de las ventajas del uso de XML como origen de datos son:

- ✓ Es extensible: después de diseñado y puesto en producción, es posible extender XML con la adición de nuevas etiquetas, de modo que se lo pueda continuar utilizando sin complicación alguna. No depende de etiquetas preestablecidas, como sucede con HTML (Lenguaje de Marcas de Hipertexto).
- ✓ El analizador es un componente estándar: no es necesario crear un analizador específico para cada versión de lenguaje XML. Esto posibilita el empleo de cualquiera de los analizadores disponibles. De esta manera se evitan errores y se acelera el desarrollo de aplicaciones.
- ✓ Si un tercero decide usar un documento creado en XML, es sencillo entender su estructura y procesarla.
- ✓ Mejora la compatibilidad entre aplicaciones: como el formato XML se basa en texto plano, se evita la necesidad de tener instalados los programas que entienden formatos específicos de bases de datos. XML es fácilmente portable entre bases de datos diferentes y distintas plataformas.
- ✓ Transforma datos en información: pues les añade un significado concreto y los asocia a un contexto, lo que le otorga flexibilidad para estructurar documentos. (Montero Ayala, 2001)

Los archivos escritos en XML no solo proporcionan información sino que además se describen por si solos, convirtiéndose en un formato ideal para intercambiar datos entre aplicaciones. Tiene el formato de un archivo de texto plano, lo que facilita enormemente la transferencia de información, logrando independencia con respecto a diferentes plataformas.

1.4.2 JSON

JSON (Notación para objetos JavaScript, siglas en inglés JavaScript Object Notation) es un formato ligero para la serialización e intercambio de datos entre sistemas y tecnologías. Describe la información con una sintaxis dedicada que se usa para identificar y gestionar datos. Está basado en un subconjunto del lenguaje de programación JavaScript que permite representar estructuras de datos (arreglos) y objetos (arreglos asociativos) en forma de texto. Una de las ventajas que tiene el uso de JSON es que puede ser leído por varios lenguajes de programación como es el caso de C, C++, Java, JavaScript, Perl, PHP y Python. Por lo tanto, puede ser usado para el intercambio de información entre distintas tecnologías. (ECMA International, 2013)

En los últimos años se ha convertido en una alternativa al formato XML, ya que es más fácil de leer y escribir, además de ser mucho más conciso. No obstante, XML es superior técnicamente porque se utiliza para definir lenguajes de marcado, mientras que JSON es simplemente un formato para intercambiar datos. Sin embargo su uso es frecuente en entornos donde el tamaño del flujo de los datos entre cliente y servidor es de vital importancia. (Pérez, 2008)

JSON es utilizado no solo para el intercambio de datos, sino que también se puede usar para almacenar estructuras de datos complejas en archivos o en bases de datos. Un objeto entero puede ser almacenado en un solo campo de una tabla o en un archivo de texto local. Como ejemplo se encuentra MongoDB, gestor de base de datos noSQL que utiliza la notación clave-valor de JSON para almacenar el contenido en colecciones de objetos.

1.4.3 CSV

CSV (Valores separados por comas, siglas en inglés Comma Separated Value) es un archivo que contiene conjuntos de datos separados por comas, donde cada salto de línea representa una fila nueva en la base de datos, y cada fila de base de datos tiene una o más campos separados por una coma (o cualquier delimitador que se defina). Se pueden organizar mediante un programa de hoja de cálculo o se insertan en una base de datos. Su principal uso se encuentra en al aplicaciones de hojas de cálculo como es el caso de KSpread, OpenOffice Calc y Microsoft Excel.

Como se utiliza en Microsoft Excel, se ha convertido en un estándar para toda la industria, incluso entre las plataformas que no son de Microsoft. CSV se utiliza comúnmente para la transferencia de datos entre bases de datos en un formato simple basado en texto. (Santos, 2012)

Desde el surgimiento del XML algunos formatos de intercambio como CSV se han convertido en algo así como formatos heredados. Muchas aplicaciones que deseen incluir un formato de exportación generalmente utilizan XML, sin embargo los archivos CSV se han convertido en un estándar por lo que su uso se mantiene en muchos de los Sistemas de Gestión de Información implementados en la actualidad. Por su simplicidad y altos niveles de soporte el uso de esta extensión como fuentes de datos se mantiene en la actualidad casi como un estándar para algunas de las principales herramientas para la generación de reportes.

1.5 Lenguaje de consultas XPath

Para el procesamiento de los ficheros XML se utiliza el lenguaje de consultas XPath como forma de acceder a los elementos o metadatos del archivo. Es un lenguaje sofisticado y complejo que permite construir expresiones para recorrer un documento XML y calcular valores a partir de su contenido. Se basa en la representación del documento como un árbol de nodos, los cuales se pueden clasificar en: raíz, elemento, atributo, texto, comentario, instrucción de procesamiento, espacio de nombres. (Montero Ayala, 2001)

Una vez realizado el estudio de la temática se seleccionan las herramientas y metodología a emplear en la solución.

1.6 Herramientas y metodología

El proceso de desarrollo de un sistema requiere el uso de herramientas y técnicas que posibiliten el cumplimiento de los requerimientos solicitados por el cliente. Para la realización de la solución propuesta se utilizaron herramientas y metodología definidas con anterioridad por el grupo de desarrollo de GDR. Es por ello que la presente investigación se enfoca solo en el estudio de las principales características de las herramientas utilizadas.

1.6.1 Symfony (v2.0.18)

Symfony es un marco de trabajo PHP que permite optimizar el desarrollo de aplicaciones y sitios web basado en el patrón Modelo Vista Controlador. Utiliza varios componentes independientes creados por el proyecto Symfony. Entre sus principales características se encuentran:

- ✓ Su código y el de todos los componentes y librerías que incluye, se publican bajo la licencia del Instituto Tecnológico de Massachusetts (MIT, Massachusetts Institute of Technology) de software libre.
- ✓ Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web.
- ✓ Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja.
- ✓ Es compatible con la mayoría de gestores de bases de datos como MySQL, PostgreSQL, Oracle y Microsoft SQL Server.
- ✓ Es multiplataforma.
- ✓ Automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.
- ✓ La documentación del proyecto también es libre e incluye varios libros y decenas de tutoriales específicos. (Symfony.es, 2015)

Algunas de las ventajas que propician el uso de Symfony en el desarrollo de la solución son:

- ✓ **Versátil:** ya que se ajusta al desarrollo de grandes, medianas y pequeñas aplicaciones. Su estructura de componentes y el uso de algunas librerías externas como doctrine se unen para formar los independientes y reutilizables bundles.
- ✓ **Útil:** desde la perspectiva del trabajo con elementos importantes en el trabajo de aplicaciones web tales como la persistencia de datos, validación, internacionalización y la importante seguridad de las mismas.
- ✓ **Buenas prácticas:** basadas en la mayoría de excelentes marco de trabajo como Spring, Django, y Rails de los cuales incluye elementos fundamentales como la configuración de la seguridad de acceso a usuarios y el uso de plantillas para la presentación del contenido.
- ✓ **Flexible:** para los diferentes tipos de desarrolladores con específicos niveles de conocimientos en el lenguaje.
- ✓ **Rendimiento:** con excelentes resultados ya que por medio del mecanismo de cache convierte todos los ficheros de configuración en código ejecutable PHP y lo almacena en cache para mejorar el rendimiento, acelerando los tiempos de respuesta de la aplicación.
- ✓ **Documentación:** aunque todavía pequeña cuenta con buenos libros donde de forma explicativa se aborda las principales elementos de este marco de trabajo y además todo ello de forma gratuita. (Brito Rodríguez, 2014)

1.6.2 Marco de trabajo Lycan

Lycan es una solución perteneciente al departamento de Integración de Soluciones del Centro de Tecnologías de Gestión de Datos (DATEC) y se basa en el uso de la biblioteca JavaScript Ext JS en su versión 3.4.0. Este marco no solo permite aprovechar las ventajas propias de Ext JS sino que además incorpora nuevas características como:

- ✓ **Área de trabajo:** se hace una óptima organización de los componentes para aprovechar al máximo el área de trabajo. Para lo cual se definen los diferentes módulos como elementos que se registrar en forma de pestañas en la aplicación. Además se le define una pequeña y minimizable barra de herramientas en forma de ribbon o listón y una extensa área de trabajo para facilitar la gestión al usuario.
- ✓ **Estructura de CRUD:** propone un conjunto de interfaces concebidas para las entidades del negocio donde se realizan las operaciones elementales de un CRUD, entiéndase por crear, listar, modificar, eliminar y buscar un objeto.
- ✓ **Uso de extensiones para CRUD:** definiendo además de las interfaces convencionales para gestionar esta operación, otras opciones como menú contextuales a través de la opción con el clic secundario y en la barra inferior del módulo, que realizan de igual manera las funcionalidades del CRUD.
- ✓ **Validación de formularios:** con personalizaciones importantes donde se validan los campos de forma tal que resulta imposible terminar una operación hasta cuanto no se haya llenado correctamente la información que requiere la interfaz.
- ✓ **Eventos y comportamientos:** que acompañan a cada una de las interfaces y de acorde a los principales escenarios donde se involucran. Resolviendo aspectos generales de este tipo de aplicación y dejando en manos de los desarrolladores necesidades específicas de los negocios tratados. (Brito Rodríguez, 2014)

1.6.3 Ext JS (v3.4)

Ext JS es una biblioteca JavaScript desarrollada por Sencha. Se define como una herramienta para facilitar el trabajo al desarrollar páginas o aplicaciones web con contenido dinámico, aplicaciones cliente-servidor o animar componentes de página usando tecnologías como AJAX, DHTML y DOM.

Además permite crear aplicaciones complejas utilizando componentes predefinidos así como un manejador de diseños similar al que provee Java Swing, gracias a esto provee una experiencia

consistente sobre cualquier navegador, evitando el problema de validar que el código escrito funcione bien en cada uno.

Entre los beneficios que brinda el uso de Ext JS se encuentran:

- ✓ Balance entre Cliente – Servidor: la carga de procesamiento se distribuye, permitiendo al servidor un funcionamiento más eficiente.
- ✓ Comunicación asíncrona: 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 lo note.
- ✓ Eficiencia de la red: El tráfico de red puede disminuir al permitir que la aplicación elija que información desea transmitir al servidor y viceversa. (Shea Frederick, 2008)

1.6.4 Entorno de desarrollo integrado (IDE)

Un IDE (siglas en inglés Integrated Development Environment) es un programa informático que contiene un conjunto de herramientas de programación. Puede funcionar como un sistema en tiempo de ejecución, donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto. Los componentes de cualquier entorno de desarrollo integrado son: un editor de texto, un compilador, un intérprete, un depurador y un constructor de interfaz gráfica (GUI). (Programacion Desarrollo, 2015)

Están diseñados para aumentar la productividad del programador, brindan componentes muy unidos con interfaces de usuario similares. Entre las características que puede tener un IDE se encuentra: ser multiplataforma, brindar soporte para diversos lenguajes de programación, permitir la integración con Sistemas de Control de Versiones, reconocimiento de Sintaxis, integración con Framework populares, depurador, importar y exportar proyectos entre otras.

1.6.4.1 NetBeans (v7.4)

NetBeans es un entorno de desarrollo gratuito y de código abierto creado para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Tiene un excelente balance entre una interfaz con múltiples opciones y un aceptable completamiento de código. Además permite el uso de un amplio rango de tecnologías de desarrollo tanto para escritorio, como aplicaciones Web, o para dispositivos móviles. Da soporte a las siguientes tecnologías: Java, PHP, Groovy, C/C++, HTML5 entre otras. Además puede instalarse en varios sistemas operativos: Windows, Linux, Mac OS. El código fuente está disponible para su reutilización de acuerdo con las licencias Common Development and Distribution License (CDDL) v1.0 y GNU General Public License (GPL) v2. (Oracle, 2015)

1.6.5 Lenguajes de programación

Un lenguaje de programación es un lenguaje formal diseñado para expresar procesos que pueden ser llevados a cabo por máquinas como las computadoras. Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana. Está formado por un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones. Al proceso por el cual se escribe, se prueba, se depura, se compila (de ser necesario) y se mantiene el código fuente de un programa informático se le llama programación. Los lenguajes de programación se dividen en tres grandes grupos: los lenguajes de máquina, los de bajo nivel y los de alto nivel. (Bonanata, 2003)

1.6.5.1 PHP 5.3

Es un lenguaje de programación de alto nivel que puede ser embebido en páginas HTML y ejecutado en el servidor. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. Entre sus características se encuentran:

- ✓ Orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos.
- ✓ Capacidad de conexión con la mayoría de los motores de base de datos que se utilizan.
- ✓ Permite aplicar técnicas de programación orientada a objetos.
- ✓ El código fuente escrito en PHP es invisible al navegador web 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. (Vázquez, 2003)

1.6.5.2 JavaScript

Es un lenguaje de programación interpretado y orientado a objetos, no es necesario compilar los programas para interpretarlo, se pueden probar directamente en cualquier navegador. Se utiliza principalmente para crear páginas web dinámicas del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario.

Algunas de las características que lo hacen viable son:

- ✓ Es un lenguaje de alto nivel, sencillo de aprender, rápido y potente.
- ✓ Fácil manejo de datos.
- ✓ Compatible con la mayoría de los navegadores modernos.
- ✓ Fácil de integrar.

- ✓ Permite agregar interactividad a elementos web.

Entre las ventajas del uso de JavaScript se encuentra la validación de datos de un formulario del lado del cliente. Un formulario desarrollado en JavaScript no permite que la información sea enviada incorrectamente, reduciendo la carga del servidor, pues no tiene que darle solución a peticiones adicionales. Otra de las potencialidades de este lenguaje es la creación de efectos dinámicos tales como imágenes dinámicas y presentaciones de diapositivas. Como la ejecución de JavaScript es dentro del navegador del cliente se puede cambiar el aspecto de la pantalla en el dispositivo de los usuarios después que la página ha sido enviada por el servidor, permitiendo al desarrollador web crear efectos dinámicos, mejorando la experiencia que recibe el usuario cuando accede a un sitio web. (Pérez, 2009)

1.6.6 Metodología de desarrollo

Se denomina metodología de desarrollo al conjunto de procedimientos, técnicas, herramientas y soporte documental necesario para la implementación del proceso de software. Se clasifican en dos tipos: pesadas y ágiles. Una buena elección de metodología posibilita la reducción de costos y retrasos de proyectos así como mejora la calidad del software. A continuación se caracteriza la metodología empleada para el desarrollo de la solución. (Pressman, 2005)

1.6.6.1 OpenUP (Open Unified Process)

Metodología dirigida a la gestión y desarrollo de proyectos informáticos basados en un desarrollo iterativo, ágil e incremental. Es un proceso mínimo y suficiente, es decir, solo el contenido fundamental necesario es incluido. Muchos de los elementos que componen OpenUP están creados para mejorar el intercambio de información entre los equipos de desarrollo.

Los principios fundamentales de la metodología OpenUP son:

- ✓ Colaborar para sincronizar intereses y compartir conocimiento. Este principio promueve prácticas que impulsan un ambiente de equipo saludable, facilitan la colaboración y desarrollan un conocimiento compartido del proyecto.
- ✓ Equilibrar las prioridades para maximizar el beneficio obtenido por los interesados en el proyecto. Este principio promueve prácticas que permiten a los participantes de los proyectos desarrollar una solución que maximice los beneficios obtenidos por los participantes y que cumple con los requisitos y restricciones del proyecto.
- ✓ Centrarse en la arquitectura de forma temprana para minimizar el riesgo y organizar el desarrollo.
- ✓ Desarrollo evolutivo para obtener retroalimentación y mejoramiento continuo. Este principio promueve prácticas que permiten a los equipos de desarrollo obtener retroalimentación temprana y

continua de los participantes del proyecto, permitiendo demostrar incrementos progresivos en la funcionalidad.

La metodología OpenUP propone las siguientes fases:

1. Concepción: Primera de las 4 fases en el proyecto del ciclo de vida, acerca del entendimiento del propósito y objetivos y obteniendo suficiente información para confirmar que el proyecto debe hacer. El objetivo de ésta fase es capturar las necesidades del cliente (stakeholder) en los objetivos del ciclo de vida para el proyecto.
2. Elaboración: Se tratan los riesgos significativos para la arquitectura. El propósito de esta fase es establecer la base la elaboración de la arquitectura del sistema.
3. Construcción: Esta fase está enfocada al diseño, implementación y prueba de las funcionalidades para desarrollar un sistema completo. El propósito de esta fase es completar el desarrollo del sistema basado en la arquitectura definida.
4. Transición: Es la última fase, cuyo propósito es asegurar que el sistema es entregado a los usuarios, y evalúa la funcionalidad y performance del último entregable de la fase de construcción.

Entre las ventajas del uso de OpenUP se encuentran:

- ✓ Es apropiado para proyectos pequeños y de bajos recursos, permite disminuir las probabilidades de fracaso en los proyectos pequeños e incrementar las probabilidades de éxito.
- ✓ Permite detectar errores tempranos a través de un ciclo iterativo.
- ✓ Evita la elaboración de documentación, diagramas e iteraciones innecesarias requeridos en la metodología RUP.
- ✓ Tiene un enfoque centrado al cliente y con iteraciones cortas.

Roles presentes en la metodología OpenUP:

- ✓ Gerente de Proyecto
- ✓ Analista
- ✓ Diseñador
- ✓ Arquitecto de software
- ✓ Desarrollador
- ✓ Stakeholders (interesados)
- ✓ Tester (Probador) (Ruiz Heredia, 2011)

1.6.7 Herramienta de modelado (Herramienta CASE)

Las herramientas de modelado se emplean para la creación de modelos de sistemas. Permiten determinar si están representados todos los requerimientos del sistema, así como definir cuales funcionalidades tendrá el sistema. Algunas de los objetivos de las herramientas de modelado son:

- ✓ Mejorar la productividad y calidad del software.
- ✓ Mejorar la planificación de un proyecto para reducir el tiempo y costo de desarrollo de un sistema informático.
- ✓ Automatizar el ciclo de desarrollo del software.
- ✓ Ayuda a la reutilización del software y la estandarización de la documentación. (Larman, 1999)

1.6.7.1 Visual Paradigm for UML (v8.0)

Herramienta CASE para el modelamiento UML muy potente, gratuita, fácil de instalar, utilizar y actualizar. Soporta el ciclo de vida completo de desarrollo de software, análisis y diseño orientado a objetos, construcción, pruebas y despliegues. Permite crear diversos tipos de diagramas UML, revertir y generar código fuente desde los diagramas UML. Incluye los objetos más recientes de este lenguaje, además de diagramas de casos de uso, diagramas de clase, diagramas de componentes. Conjuntamente soporta Java, C++, DotNet Exe/dll, XML, XML Schema, y Corba IDL, ofrece soporte para Rational Rose, integración con Microsoft Visio, además permite generar reportes y documentación en HTML/PDF. (Visual Paradigm, 2015)

1.6.8 UML 2.0 (Lenguaje de modelado unificado)

UML es lenguaje utilizado para visualizar, especificar y documentar cada una de las partes que conforman el proceso de desarrollo de software. Cuenta con una notación estándar y semánticas necesarias para el modelado de un sistema orientado a objetos. Entre las ventajas de su uso se encuentran:

- ✓ Alta reutilización y minimización de costos.
- ✓ Establecer conceptos y artefactos ejecutables.
- ✓ Mejores tiempos totales de desarrollo.
- ✓ Modelar sistemas utilizando conceptos orientados a objetos.
- ✓ Encaminar el desarrollo del escalamiento en sistemas complejos de misión crítica.
- ✓ Crear un lenguaje de modelado utilizado tanto por humanos como por máquinas.
- ✓ Mejor soporte a la planeación y al control de proyectos. (Larman, 1999)

Conclusiones del capítulo

En el presente capítulo se realizó la fundamentación teórica de la investigación a través del estudio de los conceptos relacionados con los principales generadores de reportes usados en el mundo y sus características; así como lo referente a los orígenes de datos XML, CSV y JSON. Además se efectuó la caracterización de la metodología, lenguajes, tecnologías y herramientas a emplear en el desarrollo de la solución. La investigación arrojó los siguientes resultados:

- ✓ Se evidenció la necesidad de desarrollar los componentes necesarios para la generación de reportes a partir de orígenes de datos CSV, XML y JSON con el objetivo de lograr un mayor aprovechamiento de estas fuentes.
- ✓ La metodología a utilizar para guiar el desarrollo de la solución fue OpenUP dado que la misma genera una serie de actividades y artefactos necesarios para el cumplimiento de los objetivos en cada una de las fases del ciclo de vida del sistema.
- ✓ Se utiliza UML como lenguaje de modelado y Visual Paradigm for UML v8.0 como herramienta para el modelado del sistema.
- ✓ Como lenguajes de programación se emplean PHP 5.3 y JavaScript.
- ✓ Como IDE se usa NetBeans v7.4 y como marcos de trabajo Symfony v2.0.18, Ext JS v3.4 y Lycan.

2 Capítulo II: Análisis y diseño del sistema

Introducción

En el presente capítulo se describe el proceso de análisis y diseño de la solución haciendo uso de los artefactos pertenecientes a la metodología de trabajo OpenUP. Se incluye la definición de los requisitos funcionales y no funcionales a partir del modelo de dominio confeccionado, los patrones de diseño usados, así como la representación de los diagramas de clases y de interacción según los casos de uso identificados.

2.1 Modelo de dominio

El modelo de dominio es una representación de las clases conceptuales del mundo real, no de componentes software. No se trata de un conjunto de diagramas que describen clases software, u objetos software con responsabilidades, sino que podría considerarse como un diccionario visual de las abstracciones relevantes. (Larman, 1999).

A continuación se presenta el Modelo de Dominio de la solución (ver figura 1) y la descripción de sus clases.

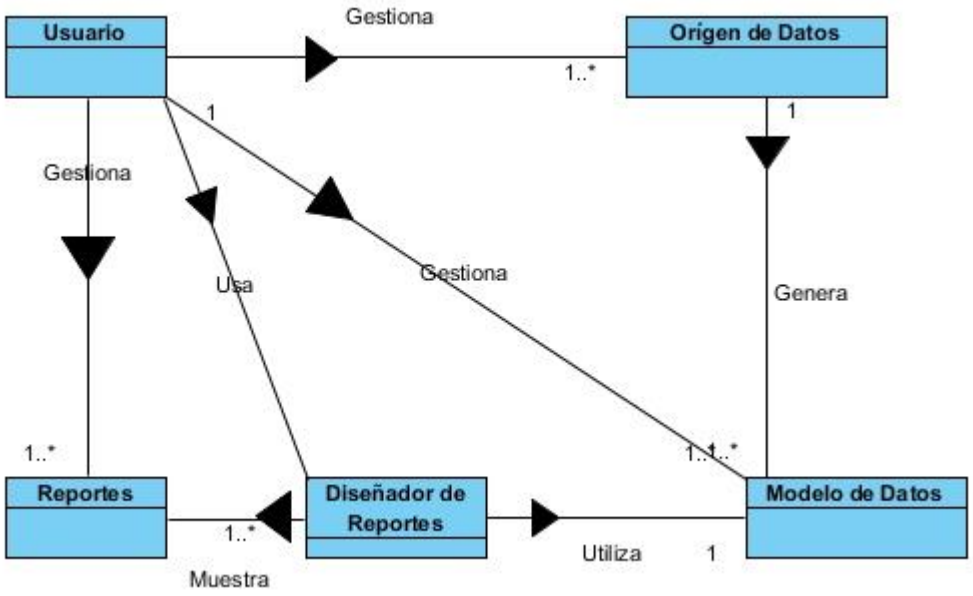


Fig. 1: Modelo de Dominio de la solución.

Descripción de las clases del dominio

- ✓ **Usuario**: persona responsable y capacitada para utilizar el sistema informático Generador Dinámico de Reportes.
- ✓ **Modelo de datos**: contiene los metadatos asociados a un origen de datos registrado en el sistema.
- ✓ **Origen de datos**: definición general de la fuente de información.
- ✓ **Diseñador de reportes**: es el encargado de diseñar la estructura de los reportes.
- ✓ **Reporte**: documento generado por el sistema que visualiza de manera estructurada y/o resumida los datos de acuerdo a un interés específico.

2.2 Requisitos del sistema

Los requisitos del sistema definen las características necesarias para que el sistema sea aceptado por el cliente, describiendo textualmente cómo funciona el sistema a partir de ciertas condiciones de entrada generando una salida. Algunas de las características que debe tener un requisito son:

- ✓ Consistencia: no puede existir conflicto con otros requisitos.
- ✓ Completitud: debe contener en sí mismo toda la información necesaria.
- ✓ Alcance: debe ser realista, es posible cumplirse con los recursos y esfuerzos disponibles.
- ✓ Conciso: debe ser comprensible para todos. (Pressman, 2010)

Existen tres grupos de requisitos: funcionales, no funcionales y otros requisitos relacionados con limitaciones externas. A continuación se describen los requisitos funcionales y no funcionales que conforman el sistema.

2.2.1 Requisitos funcionales

Los requisitos funcionales describen la interacción entre el sistema y su ambiente independientemente de su implementación. Son propiedades que el sistema debe ser capaz de cumplir. A continuación los requisitos funcionales definidos:

Tabla 2: Requisitos funcionales

Nº	Funcionalidad	Entrada	Salida	Descripción
RF1	Adicionar origen de datos XML	Se definen los parámetros: nombre del nuevo origen de datos y dirección física del archivo	Muestra un mensaje con el resultado de la operación y una lista actualizada con los orígenes de datos existentes en la base de datos.	Agrega en el sistema un nuevo origen de datos a partir de un archivo XML

		XML.		
RF2	Eliminar origen de datos XML	Se selecciona origen de datos a eliminar	Muestra una solicitud de confirmación antes de eliminar, luego un mensaje con el resultado de la operación y una lista actualizada con los orígenes de datos existentes en la base de datos.	Elimina el archivo XML y el origen de datos de la base de datos.
RF3	Modificar origen de datos XML	Se definen los nuevos valores del origen de datos: nombre del origen de datos y/o dirección física del archivo XML.	Muestra una lista actualizada con los orígenes de datos existentes en la base de datos.	Modificar parámetros de un origen de datos.
RF4	Adicionar origen de datos JSON	Se definen los parámetros: nombre del nuevo origen de datos y dirección física del archivo JSON.	Muestra un mensaje con el resultado de la operación y una lista con los orígenes de datos existentes en la base de datos.	Agrega en el sistema un nuevo origen de datos a partir de un archivo de tipo JSON
RF5	Eliminar origen de datos JSON	Se selecciona origen de datos a eliminar	Muestra una solicitud de confirmación antes de eliminar, luego un mensaje con el resultado de la operación y una lista actualizada con los orígenes de datos existentes en la base de datos.	Elimina el archivo JSON y el origen de datos de la base de datos.
RF6	Modificar origen de datos JSON	Se definen los nuevos valores del origen de datos: nombre del origen de datos y/o dirección física del	Muestra una lista actualizada con los orígenes de datos existentes en la base de datos.	Modificar parámetros de un origen de datos.

		archivo JSON.		
RF7	Adicionar origen de datos CSV	Se definen los parámetros: nombre del nuevo origen de datos y dirección física del archivo.	Muestra un mensaje con el resultado de la operación y una lista con los orígenes de datos existentes en la base de datos.	Agrega en el sistema un nuevo origen de datos a partir de un archivo CSV
RF8	Eliminar origen de datos CSV	Se selecciona origen de datos a eliminar	Muestra una solicitud de confirmación antes de eliminar, luego un mensaje con el resultado de la operación y una lista actualizada con los orígenes de datos existentes en la base de datos.	Elimina el archivo CSV y el origen de datos de la base de datos.
RF9	Modificar origen de datos CSV	Se definen los nuevos valores del origen de datos: nombre del origen de datos y/o dirección física del archivo CSV.	Muestra una lista actualizada con los orígenes de datos existentes en la base de datos.	Modificar parámetros de un origen de datos.
RF10	Adicionar modelo de datos	A partir de un origen de datos seleccionado se introducen los parámetros nombre del modelo y/o descripción.	Muestra una lista con los modelos de datos existentes en la base de datos.	A partir de un origen de datos crear un modelo.
RF11	Eliminar modelo de datos	Se selecciona el modelo de datos a eliminar.	Muestra una solicitud de confirmación antes de eliminar, luego un mensaje con el resultado de la operación y una lista actualizada con los modelos de datos existentes en la base de datos.	Eliminar un modelo de datos seleccionado

RF12	Renombrar modelo de datos	Se definen los nuevos valores del modelo de datos: nombre del modelo de datos y/o descripción.	Muestra una lista actualizada con los modelos de datos existentes en la base de datos.	Modificar los parámetros nombre y/o descripción de un modelo de datos seleccionado.
RF13	Diseñar consultas XPath a partir de una fuente de datos XML.	Se define la consulta XPath y los campos del reporte a partir del modelo de datos.	Muestra una vista previa del reporte.	A partir de una consulta XPath se accede a los datos del archivo XML y luego generar una vista previa del reporte
RF14	Diseñar consultas a partir de la fuente de datos CSV	Se definen los campos del reporte a partir del modelo de datos.	Muestra una vista previa del reporte.	A partir de una consulta se accede a los datos del archivo CSV y luego generar una vista previa del reporte
RF15	Diseñar consultas a partir de una fuente de datos JSON	Se define la consulta JSON y los campos del reporte a partir del modelo de datos.	Muestra una vista previa del reporte.	A partir de una consulta se accede a los datos del archivo JSON y luego generar una vista previa del reporte

2.2.2 Requisitos no funcionales

Los requisitos no funcionales describen aspectos o cualidades que hacen al sistema usable, incluyen restricciones como el tiempo de respuesta (desempeño), la precisión, recursos consumidos, seguridad. A continuación los requisitos no funcionales definidos:

RnF 1: Software requerido para desplegar y utilizar la aplicación

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

- ✓ Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debian 4 GNU/Linux o superior.

- ✓ Paquetes: apache2, php5, php5-cli, php5-mysql, php5-pgsql, php5-sybase, php5-sqlite, php5-xsl, php5-gd, php5-odbc, php5-curl, php-apc, php-xml-serializer, libapache2-mod-php5.
- ✓ Navegador Mozilla Firefox hasta la versión 38.0.
- ✓ Usuario con privilegios de administrador.

El servidor donde se instalará la Base de Datos debe cumplir con los siguientes requisitos:

- ✓ Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debian 4 GNU/Linux o superior.
- ✓ PostgreSQL versión 8.3 o superior.
- ✓ pgAdmin III u otro administrador compatible con 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.

RnF 2: Hardware requerido para desplegar y utilizar la aplicación

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

- ✓ 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.

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.

RnF 3: Interfaces de Comunicación

- ✓ El sistema puede ser desplegado sobre red LAN, MAN o WAM; siempre y cuando la velocidad de conexión sea mayor que 1 Mbit/s.

RnF 4: Estándares aplicables

- ✓ Durante el proceso de desarrollo de software cada entregable es sometido a pruebas de liberación, en las que se evalúan las características y sub-características de calidad definidas por la ISO/IEC 9126. Como estándar de codificación se usarán los definidos para Symfony 2 en los documentos PSR-0, PSR-1 y PSR-2.

RnF 5: Finalidad

- ✓ El objetivo que persigue la aplicación es generar de forma sencilla reportes dinámicos de una base de datos según las necesidades del usuario.

RnF 6: Tipo de aplicación informática

- ✓ La herramienta debe ser WEB, pero con características muy similares a las aplicaciones de escritorio en cuanto al diseño de las interfaces visuales y el tiempo de respuesta de la interacción del usuario con el sistema.

RnF 7: Interfaces de usuario

- ✓ Las interfaces de usuario serán diseñadas a modo de aplicaciones RIA (Rich Internet Application) lo que permite a los usuarios contar con aplicaciones web con una experiencia de usuario similar a la de las aplicaciones de escritorio. Para lograr este fin se usará la librería JavaScript ExtJS, la cual conjuga una serie de componentes visuales que proveen funcionalidades, que ayudan al diseño de este tipo de aplicaciones WEB con apariencia de escritorio. (Ver documento GDR2.0_ ERS_1s.doc)

2.3 Modelo de casos de uso del sistema

Los casos de uso describen las actividades que deben llevarse a cabo para desarrollar un proceso. Se define como el documento narrativo que describe la secuencia de eventos de un actor (agente externo) que utiliza un sistema para completar un proceso; no son exactamente los requerimientos ni las especificaciones funcionales, sino que ejemplifican e incluyen tácitamente los requerimientos en las historias que narran. (Larman, 1999)

2.3.1 Diagrama de casos de uso del sistema

Los diagramas de casos de uso del sistema ayudan a representar el comportamiento de un sistema a partir de su interacción con el usuario u otro sistema. Se utilizan para mostrar la relación entre los actores y los casos de uso del sistema. A continuación el diagrama de casos de uso desarrollado a partir de las funcionalidades que debe cumplir la aplicación.

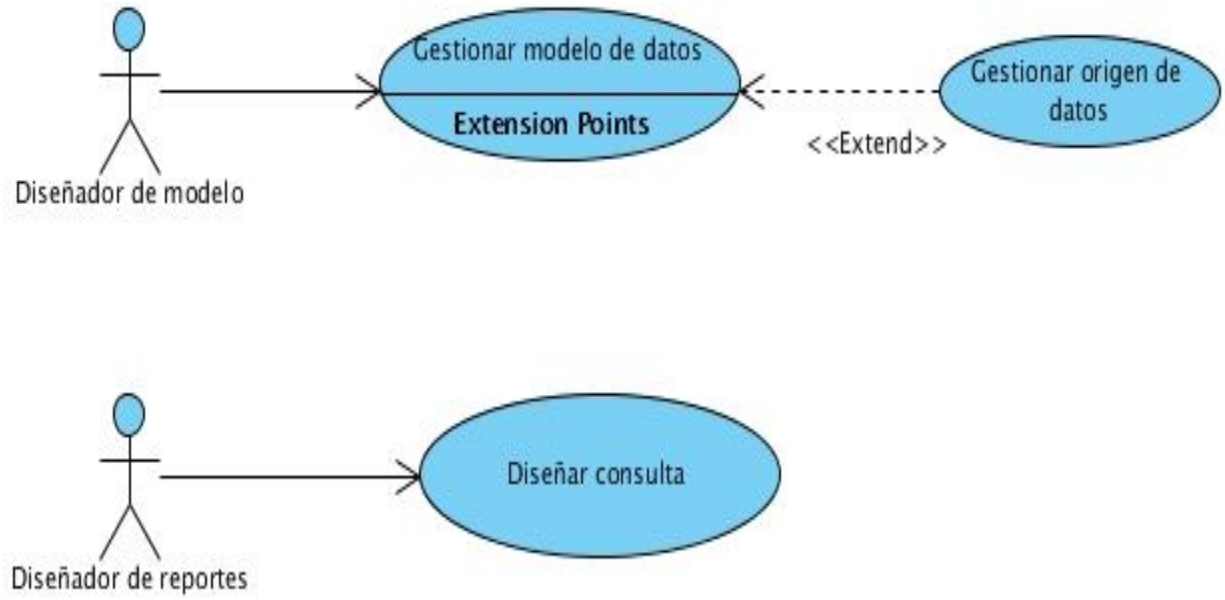


Fig. 2: Diagrama de Casos de Uso del Sistema

2.3.2 Descripción de los casos de uso

Gestionar origen de datos: el CUS permite crear, eliminar, modificar y mostrar los orígenes de datos a partir de un archivo XML, CSV o JSON. El CU tiene cuatro diversos flujos básicos: Adicionar origen de datos, Eliminar origen de datos, Modificar origen de datos, Mostrar origen de datos.

Gestionar modelo de datos: permite crear, eliminar, renombrar y mostrar modelos de datos partir de un origen de datos seleccionado El CU tiene cuatro diversos flujos básicos: Adicionar modelo de datos, Eliminar Modelo de Datos, Renombrar modelo de datos, Buscar modelo de datos y Mostrar modelo de datos.

Diseñar consulta: permite diseñar la consulta para un reporte determinado dependiendo del modelo de datos que se utilice. El CU tiene 3 flujos básicos: Agregar campo, Seleccionar modelo, Generar consulta.

2.3.3 Patrones de CU

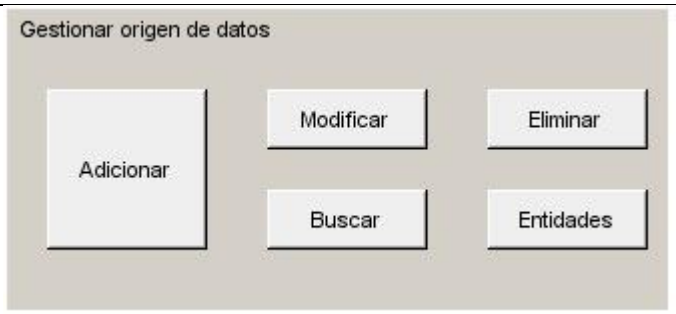
Extensión: Este patrón se aplica cuando un flujo puede extender el flujo de otro caso de uso así como ser realizado en sí mismo. Como se evidencia en la figura 2 el flujo de Gestionar modelo de datos no puede comenzar si no existe al menos un origen de datos en el sistema.

CRUD completo: Este patrón se evidencia en los casos donde se quiere realizar operación de adición, eliminar, modificación y consulta a alguna entidad del sistema. Como se observa en la figura 2 el CU Gestionar modelo de datos y el CU Gestionar origen de datos cumplen el patrón explicado.

2.3.4 Descripción del Caso de Uso arquitectónicamente significativo del sistema.

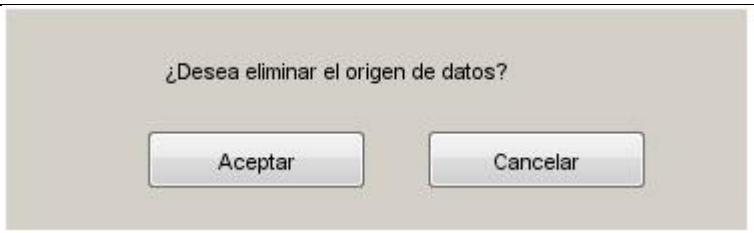
Tabla 3: Descripción del CU Gestionar origen de datos

Caso de uso:	Gestionar origen de datos
Actores:	Diseñador de modelo
Resumen	El CU permite mostrar, adicionar, eliminar y modificar orígenes de datos. El CU comienza cuando el usuario decide agregar un nuevo origen de datos, una vez adicionado puede realizar las operaciones de modificación y eliminación.
Referencias:	RF1, RF2, RF3, RF4, RF5, RF6, RF7, RF8, RF9
Precondiciones:	El usuario debe estar autenticado
Prioridad:	Alta(Crítico)
Flujo normal de eventos	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona la opción “Diseñador de modelos”.	2. Muestra la interfaz “Diseñador de Modelos” brindándole un listado de los orígenes de datos previamente creados. Si el actor decide crear un nuevo origen de datos, ver sección “Adicionar origen de datos”. Si el actor decide eliminar un origen de datos, ver sección “Eliminar origen de datos”. Si el actor decide modificar un origen de datos, ver sección “Modificar origen de datos”.
Prototipo de la interfaz	

<table>
<tr><td colspan="2"> 
</td></tr>
<tr><td>Poscondiciones</td><td></td></tr>
<tr><td colspan="2">Sección Adicionar origen de datos</td></tr>
<tr><td>Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td>1-Selecciona la opción adicionar origen de datos.</td><td>2-Muestra un listado con los tipos de orígenes de datos que posee.</td></tr>
<tr><td>3- Selecciona un tipo de origen de datos</td><td>4- El sistema muestra la interfaz ¨Adicionar origen de datos¨ con los campos: nombre y dirección del fichero a utilizar.</td></tr>
<tr><td>5-Introduce un nombre y selecciona el fichero.</td><td>6-Habilita la opción ¨Aceptar¨. En caso de que el actor seleccione la opción ¨Cancelar¨, ver Flujo Alterno 6.1</td></tr>
<tr><td>7-Selecciona el botón ¨Aceptar¨.</td><td>El sistema muestra un mensaje de confirmación, creando el nuevo origen de datos.</td></tr>
<tr><td colspan="2">Flujos Alternos</td></tr>
<tr><td>Acción del actor</td><td>Respuesta del Sistema</td></tr>
<tr><td></td><td>6.1. El sistema cierra la interfaz “Generar Modelo”.</td></tr>
<tr><td colspan="2">Prototipo de interfaz</td></tr>
</table>

Poscondiciones	Listado actualizado con los orígenes de datos presentes en la aplicación.

Sección ¨Eliminar origen de datos¨	
Acción del Actor	**Respuesta del Sistema**
1. Selecciona un origen de datos del listado	2. Resalta el origen de datos seleccionado.
3. Selecciona la opción ¨Eliminar origen de datos¨	4. El sistema muestra la interfaz ¨Eliminar origen de datos¨.
5. Selecciona la opción ¨Aceptar¨	6. Elimina el origen. En caso de que el actor seleccione la opción ¨Cancelar¨, ver Flujo Alterno 6.1
7. Selecciona el botón ¨Aceptar¨.	8. El sistema muestra un mensaje de confirmación, elimina el origen de datos.
Flujos Alternos	
Acción del actor	Respuesta del Sistema
	6.1. El sistema cierra la interfaz “Eliminar origen de datos”.
Prototipo de interfaz	

<table>
<tr><td colspan="2"> 
</td></tr>
<tr><td>Poscondiciones</td><td>Listado actualizado con los modelos de datos presentes en la aplicación.</td></tr>
<tr><td colspan="2">Sección Modificar origen de datos</td></tr>
<tr><td>Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td>1-Selecciona un origen de datos de la lista.</td><td>2- Resalta el origen de datos seleccionado.</td></tr>
<tr><td>3- Selecciona la opción ¨¨Modificar origen de datos¨.</td><td>4- El sistema muestra la interfaz ¨Modificar origen de datos¨ con los campos: nombre y dirección del fichero a utilizar.</td></tr>
<tr><td>5-Introduce un nombre y selecciona el fichero.</td><td>6-Habilita la opción ¨Aceptar¨. En caso de que el actor seleccione la opción ¨Cancelar¨, ver Flujo Alterno 6.1</td></tr>
<tr><td>7-Selecciona el botón ¨Aceptar¨.</td><td>El sistema muestra un mensaje de confirmación, creando el nuevo origen de datos.</td></tr>
<tr><td colspan="2">Flujos Alternos</td></tr>
<tr><td>Acción del actor</td><td>Respuesta del Sistema</td></tr>
<tr><td></td><td>6.1. El sistema cierra la interfaz “Generar Modelo”.</td></tr>
<tr><td colspan="2">Prototipo de interfaz</td></tr>
</table>

<table>
<tr><td colspan="2"> 
</td></tr>
<tr><td>Poscondiciones</td><td>Listado actualizado con los orígenes de datos presentes en la aplicación.</td></tr>
</table>

2.4 Modelo de diseño

Mediante el modelo de diseño se elaboran los diagramas que reflejan el comportamiento del sistema a partir de los requisitos identificados anteriormente. Entre los diagramas se encuentran los diagramas de clase del diseño y los diagramas de secuencia.

2.4.1 Diagrama de clases de diseño

Un diagrama de clases de diseño visualiza las especificaciones para cada clase (métodos y atributos) y la relación existente entre ellas. Además permiten modelar una vista estática del diseño del sistema. A continuación se muestra el diagrama de clases de diseño correspondiente al Caso de Uso Gestionar origen de datos:

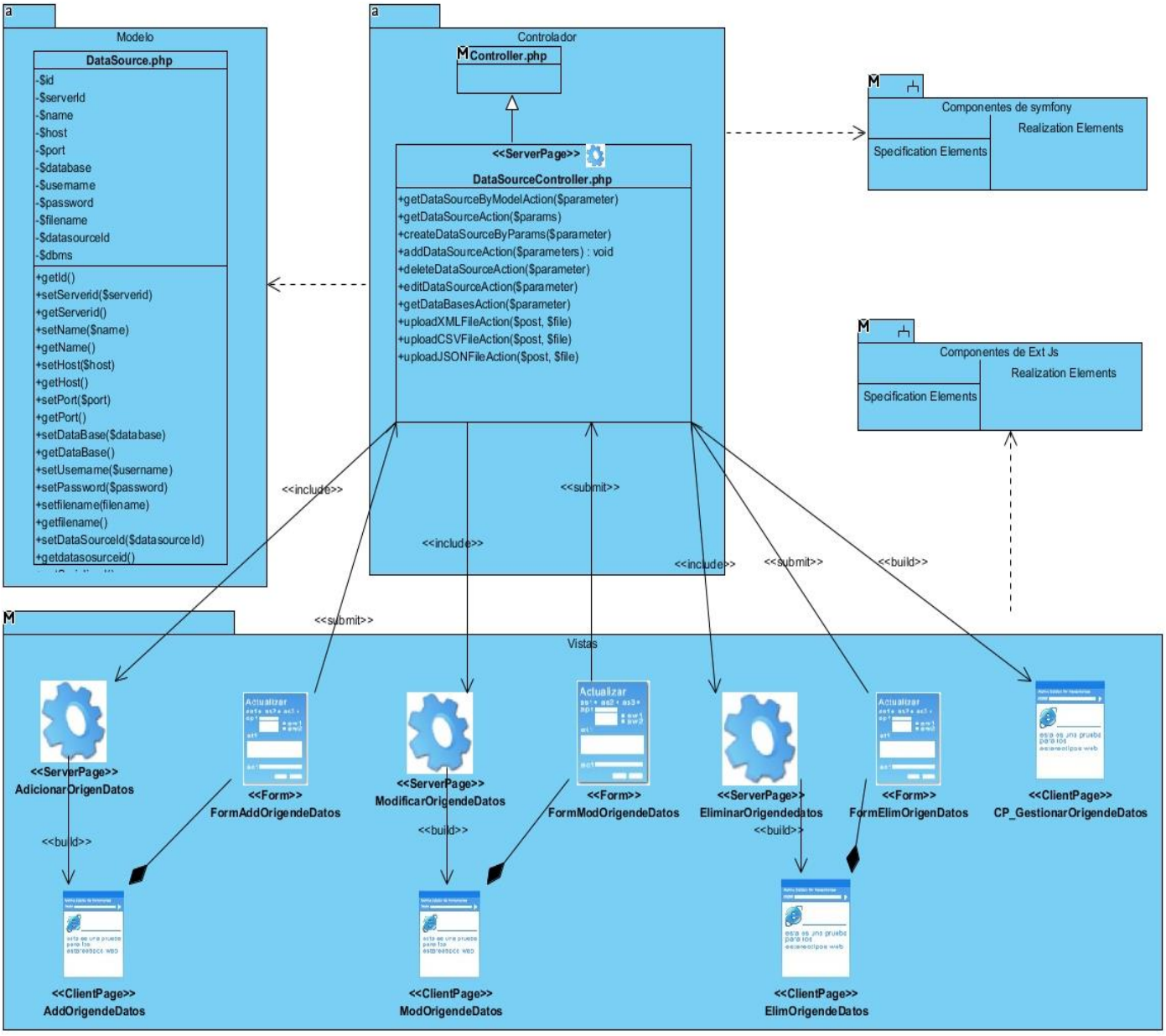


Fig. 3: Diagrama de clases del diseño del CU Gestionar Orígenes de Datos

En la figura 3 se muestra el diagrama de clases de diseño para el CU Gestionar Origen de Datos dividido en tres componentes fundamentales: el modelo que contiene la clase DataSource.php encargado de crear un objeto de tipo origen de datos; la vista compuesta por las interfaces para la interacción con el usuario; el controlador que contiene la clase DataSourceController.php gestiona las peticiones realizadas a través de los formularios.

2.4.2 Diagrama de secuencia de CU Gestionar Origen de Datos

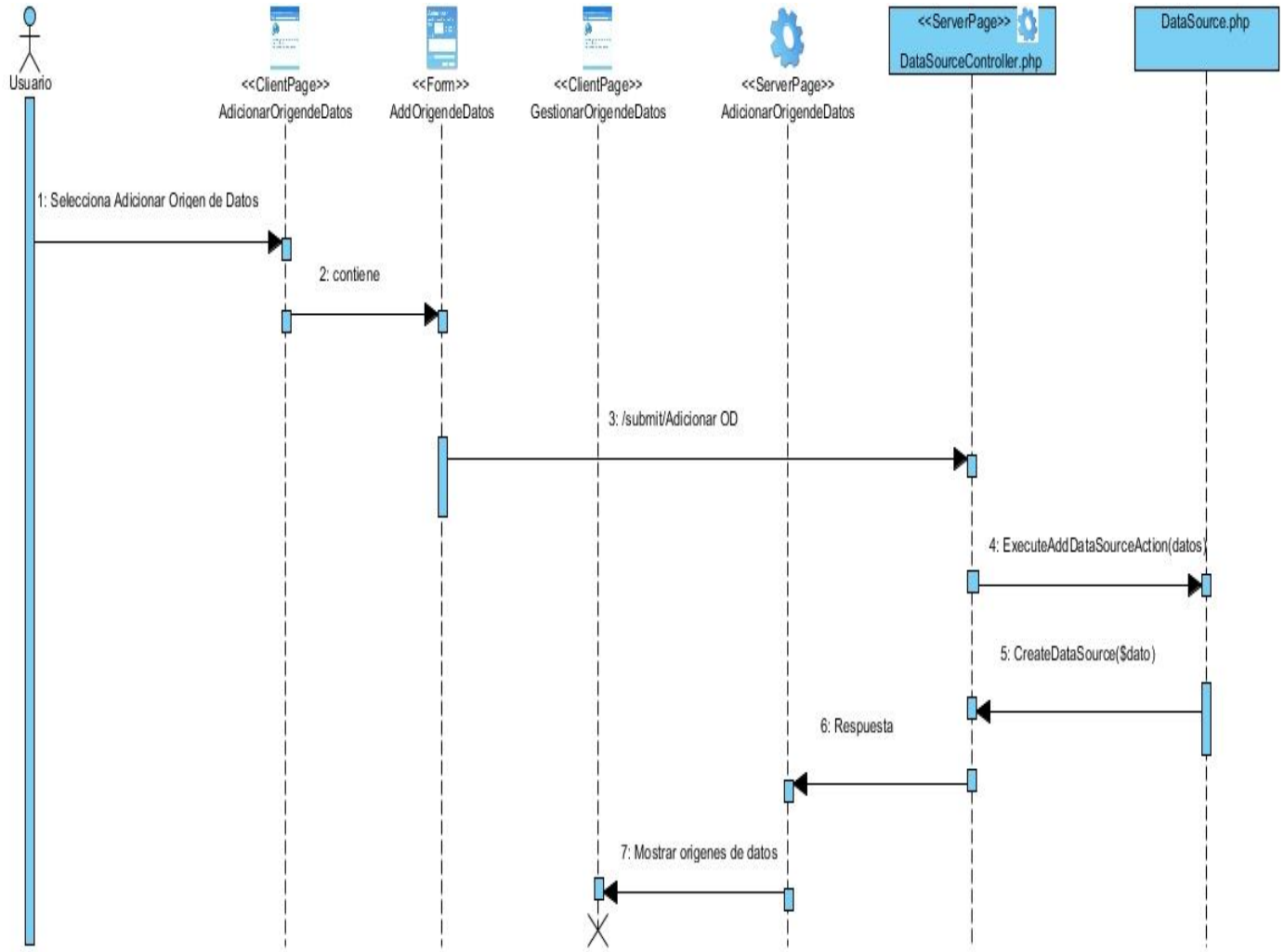


Fig. 4: Diagrama de secuencia del CU Gestionar Origen de Datos.

Como se muestra en la figura 5 el usuario selecciona la opción “Adicionar origen de datos”, luego la página CP_AdicionarOrigenDatos se encarga de mostrar el formulario ¨AddOrigendeDatos¨ con los parámetros necesarios. Cuando se acepta el formulario se envía una petición a la clase controladora DataSourceController.php ejecutándose el método AddDataSourceAction (), accede a la base de datos y crea un modelo utilizando la clase DataSource.php. Luego adiciona el origen de datos y envía una respuesta a la página principal SP_AdicionarOrigendeDatos que se encarga de mostrar la lista de orígenes creados en el sistema a través de la clase CP_GestionarOrigendeDatos.

2.5 Patrones de software

Los patrones de software son técnicas de diseño de software que ayudan a solucionar una serie problemas con características comunes. Son de gran importancia, pues permiten ahorrar en tiempo con la reutilización de estructuras de software que solucione un problema que ya ha sido resuelto en otra ocasión. Además, se logra una mayor uniformidad en el equipo de trabajo, pues obliga a los desarrolladores a utilizar una metodología estándar y con mejores resultados.

2.5.1 Patrones de arquitectura

Los patrones de arquitectura están orientados a representar los diferentes elementos que componen una solución de software y las relaciones entre ellos. Especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes. Los beneficios de la utilización de dichos patrones van desde la imposición de decisiones tempranas en el desarrollo hasta la reutilización.

2.5.1.1 Modelo-Vista-Controlador (MVC)

Es un patrón de arquitectura de software perteneciente a la familia de los estilos arquitectónicos de Llamada y retorno. Dicho patrón separa los datos, la interfaz de usuario y la lógica de negocio en tres componentes.

Modelo: representa la lógica de negocio y es encargado de acceder a los datos, así como su procesamiento, validación, asociación y cualquier otra tarea relativa a la manipulación de dichos datos.

Vista: es responsable del uso de la información de la cual dispone para producir cualquier interfaz de presentación de cualquier petición que se presente.

Controlador: gestiona las peticiones de los usuarios y funciona como intermediario entre la vista y el modelo.

La conexión entre los componentes de MVC se inicia generalmente cuando el usuario interactúa con la interfaz o vista y solicita una acción. El controlador recibe la solicitud y procede a gestionar el evento obteniendo los datos o actualizándolos a partir de modelo. Por último el controlador delega la responsabilidad de mostrar los resultados a la vista. (Bahit, 2011)

A continuación se muestra la interconexión entre los componentes de Symfony:

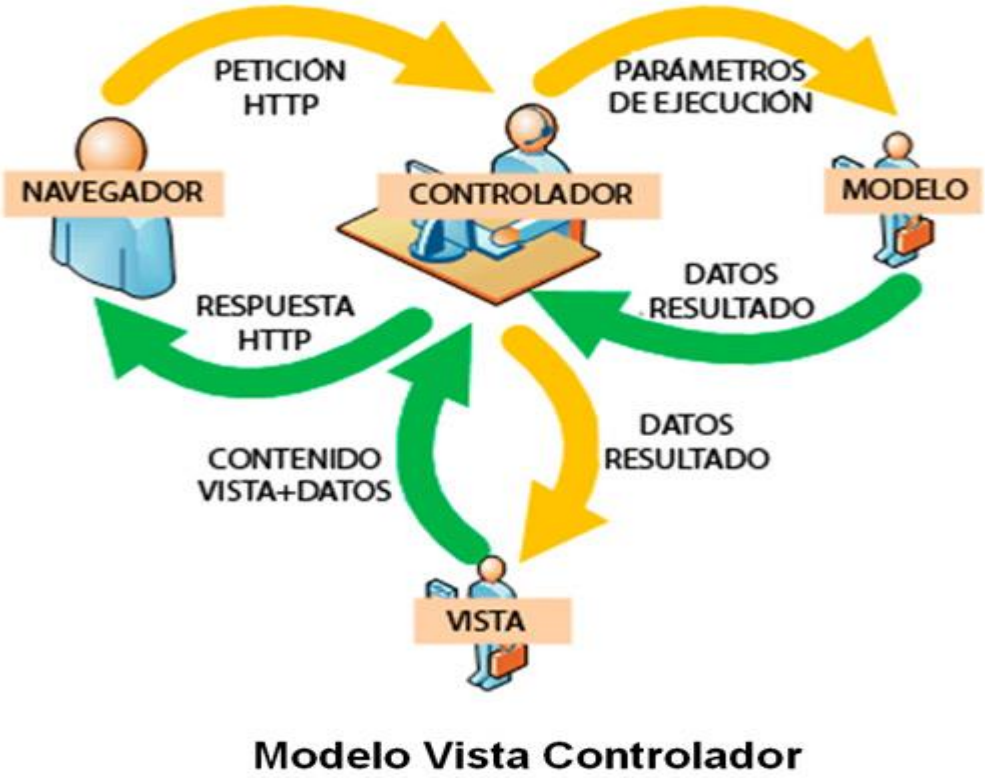


Fig. 5: Interacción entre los componentes de MVC con Symfony

2.5.2 Patrones de diseño

Los patrones de diseño brindan una solución ya probada y documentada a problemas típicos y recurrentes que se pueden encontrar a la hora de desarrollar un software. Ayudan a cumplir muchos de los principios o reglas de diseño y facilita la reutilización de arquitecturas y diseños de softwares exitosos, lo cual hace a cualquier aplicación más robusta. Los patrones se pueden clasificar en tres grandes grupos:

- ✓ Creacional: relacionado con la inicialización y configuración de objetos
- ✓ De comportamiento: Más que describir objetos o clases, describen la comunicación entre ellos.
- ✓ Estructurales: Separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes. (Larman, 1999)

2.5.2.1 Patrones GRASP

Los patrones de diseños GRASP (Patrones de principios generales para asignar responsabilidades) describen los principios fundamentales del diseño de objetos y la asignación de responsabilidades,

expresados como patrones. Entre los patrones de diseño GRASP presentes en el sistema se encuentran:

Experto: un experto es una clase que tiene toda la información necesaria para implementar una responsabilidad posibilitando el encapsulamiento de la información. Se evidencia en la clase *ModelController.php* debido a que posee toda la información necesaria para crear y obtener modelos de datos.

Fig. 6: Patrón GRASP experto

Creador: El patrón creador guía la asignación de responsabilidades relacionadas con la creación de objetos. Tiene como objetivo mantener un bajo acoplamiento lográndose mayor mantenibilidad y reutilización del código. (Larman, 1999)En la solución este patrón se evidencia en la clase *DataSourceController.php* encargada de crear instancias de DataSource.php.

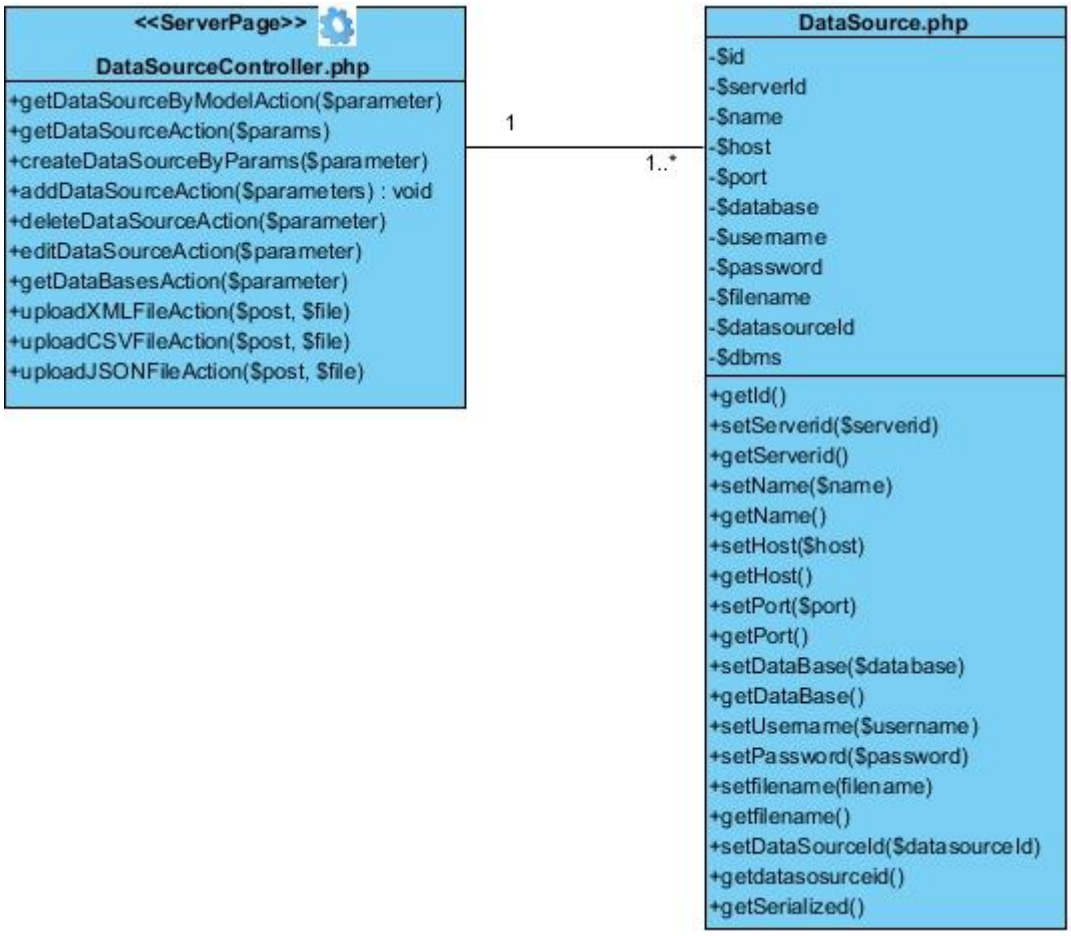
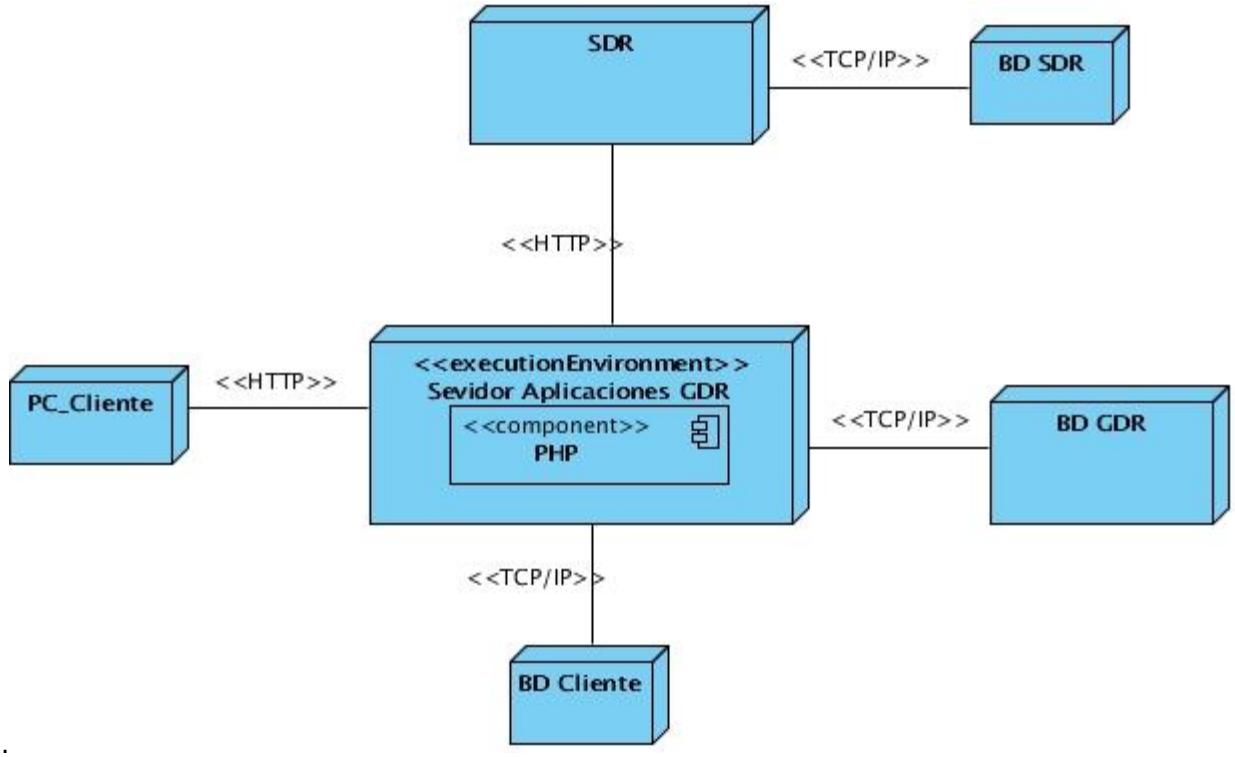


Fig. 7: Patrón GRASP creador

2.6 Diagrama de despliegue

El diagrama de despliegue permite modelar las relaciones entre los distintos nodos que conforman el sistema en tiempo de ejecución, es decir, ayuda a comprender la disposición física de los artefactos software que conforman la arquitectura de la aplicación. A continuación en la figura 8 se muestra el diagrama de despliegue del GDRv2.0:



.

Fig. 8: Diagrama de despliegue

- ✓ **PC_Cliente:** computadora desde donde se accede a la aplicación.
- ✓ **Servidor Aplicaciones GDR:** computadora donde está publicada la aplicación GDR.
- ✓ **SDR:** computadora donde está publicada la aplicación SDR
- ✓ **BD GDR:** contiene toda la información del GDR.
- ✓ **BD SDR:** contiene toda la información del SDR.
- ✓ **BD Cliente:** contiene toda la información del cliente.
- ✓ **TCP/IP:** protocolos de comunicación.
- ✓ **HTTP:** protocolo de comunicación.

Conclusiones del capítulo

En el presente capítulo se llevó a cabo el análisis y diseño del sistema a partir de la metodología de trabajo OpenUP arrojando los siguientes resultados:

✓ La realización del modelo de dominio permitió obtener una visión detallada de los procesos del sistema.

✓ La disciplina de requisitos posibilitó la identificación y descripción de todos los requisitos funcionales a incorporar en la solución. Se identificaron 15 requisitos funcionales agrupados en tres casos de uso.

✓ El análisis y diseño permitió la confección de los artefactos correspondientes a dicha disciplina, donde se obtuvo: la arquitectura del sistema utilizando el patrón de arquitectura MVC; el diseño y la descripción de los casos de uso del sistema; los diagramas de clases del diseño que muestran las clases, atributos, métodos y sus relaciones para llevar a cabo el desarrollo de la solución; los diagramas de secuencias; diagrama de despliegue.

3 Capítulo III: Implementación y pruebas

Introducción

Después de concluido el análisis y diseño de la solución, se procede a realizar las actividades de implementación y pruebas. La implementación es la creación de elementos (código, visuales) que forman parte del sistema, mientras que el proceso de pruebas se encarga de comprobar si todos los requisitos definidos fueron cumplidos en su totalidad.

3.1 Modelo de implementación

El Modelo de implementación está comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes podemos encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos.

3.1.1 Diagrama de componentes

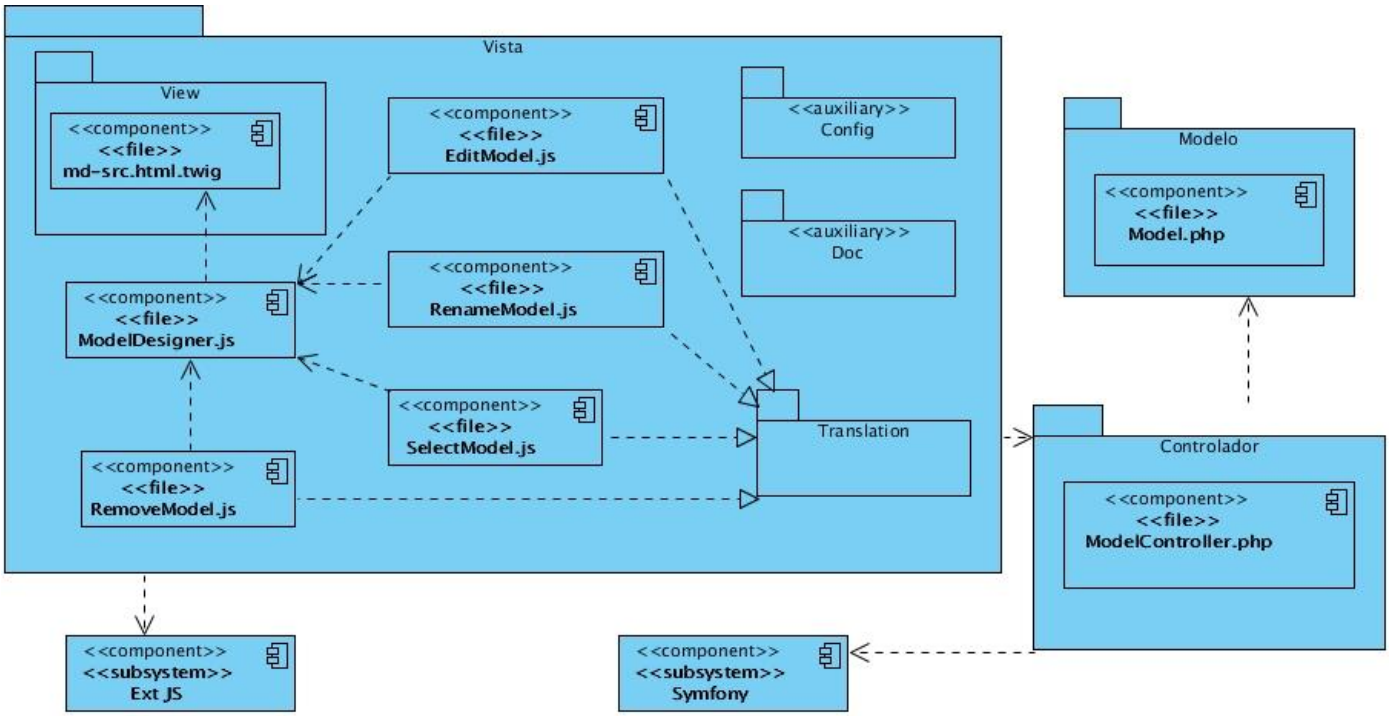


Fig. 9: Diagrama de componentes del CU Gestionar Modelos de Datos

3.2 Estándares de codificación

Un estándar de codificación se refiere a los aspectos relacionados con la generación de código. La legibilidad del código fuente no influye en la capacidad funcional de la aplicación pero si en la forma en que un programador pudiera comprender el sistema. El uso de buenas técnicas de programación y codificación sólida ayuda a generar un código de alta calidad, lo cual influye en la calidad del software. (Microsoft, 2015)

Para el desarrollo de la solución se utilizaron los estándares de codificación definidos para Symfony 2 los cuales se reflejan en los estándares PSR-0, PSR-1 y PSR-2 para PHP. A continuación se observan algunos de los estándares de estructura y nomenclatura utilizados para la codificación de la solución:

- ✓ Colocar las llaves en su propia línea para clases, métodos y en la declaración de funciones
- ✓ Separar las declaraciones condicionales (if, else, ...) y la llave de apertura con un solo espacio y sin ninguna línea en blanco.

- ✓ Usa minúsculas para escribir las constantes nativas de PHP: false, true y null. Lo mismo ocurre con array ().
- ✓ Usar cadenas en mayúsculas para constantes con palabras separadas por subrayados.
- ✓ Utilizar espacios de nombres (namespaces) para todas las clases.
- ✓ Utilizar caracteres alfanuméricos y subrayados para los nombres de archivo.
- ✓ Uso de mayúsculas intercaladas, no subrayados, para variables, funciones y nombres de métodos. (Fabien Potencier, 2015)

3.3 Implementación de los componentes

A continuación se evidencian algunos de los cambios implementados en el GDRv2.0 para la presente solución. En el módulo Diseñador de Modelos se modificaron las clases DataSourceController.php y ModelController.php, estas contienen los métodos necesarios para la conexión con los orígenes de datos y la obtención de los metadatos respectivamente. En el caso de la primera clase se agregaron las funcionalidades uploadXMLFileAction (), uploadCSVFileAction () y uploadJSONFileAction () que permiten copiar en el servidor el archivo perteneciente al origen de datos seleccionado; además valida que el archivo contenga el formato y tamaño requerido. En cuanto las vistas fueron creadas las clases LoadXML.js, LoadCSV.js y LoadJSON.js para el diseño de los formularios de adicionar origen de datos.

3.4 Pruebas de software

Durante y después del proceso de implementación la aplicación debe ser comprobada para asegurar el cumplimiento de los requisitos identificados durante el proceso de análisis. Las pruebas constituyen los exámenes que se realiza al software con el objetivo de identificar fallos en el proceso de desarrollo, verificar la integración adecuada de los componentes y proporcionar información sobre la calidad del producto. (Sommerville, 2005)

3.4.1 Niveles de prueba

Los niveles de prueba son maneras de comprobar y validar un producto de software en diferentes etapas del proceso de desarrollo. Se definieron los siguientes niveles de pruebas:

Pruebas de unidad:

Es la primera fase de las pruebas dinámicas y se realizan sobre cada módulo del software de manera independiente. El objetivo es comprobar que el módulo, entendido como una unidad funcional de un

programa independiente, está correctamente codificado. En estas pruebas cada módulo será probado por separado y lo hará, generalmente, la persona que lo creó. Estas pruebas permiten verificar al desarrollador que los componentes unitarios están codificados bajo condiciones de robustez. Para llevar a cabo esta prueba se empleó el siguiente procedimiento:

1. Se prueba la interfaz del módulo para asegurar que la información fluye de forma adecuada hacia y desde la unidad de programa que está siendo probada.
2. Se examinan las estructuras de datos locales para asegurar que los datos que se mantienen temporalmente conservan su integridad durante todos los pasos de ejecución del algoritmo.
3. Se prueban las condiciones límite para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones de procesamiento.
4. Se ejercitan todos los caminos independientes (caminos básicos) de la estructura de control con el fin de asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez.
5. Se prueban todos los caminos de manejo de errores. (Pressman, 2005)

Pruebas de integración:

Es una técnica sistemática para construir la estructura del programa mientras que, al mismo tiempo, se llevan a cabo pruebas para detectar errores asociados con la interacción. Su objetivo es identificar errores introducidos por la combinación de programas o componentes probados unitariamente, para asegurar que la comunicación, enlaces y los datos compartidos ocurran apropiadamente. (Pressman, 2005)

Pruebas de sistema: verifica que se han integrado todos los elementos del sistema. No prueba el detalle de cada módulo, sino más bien la integración de cada módulo en el sistema, buscando las discrepancias que ocurran y la falta de compatibilidad entre ellos.

Pruebas de aceptación:

Estas pruebas corresponden a la fase final del proceso de desarrollo del software y están enfocadas a probar los requisitos de usuario o los criterios de aceptación. Las pruebas de aceptación fueron ejecutadas por personal perteneciente al Departamento de Desarrollo de Componentes del Centro de Tecnología y Gestión de Datos de la facultad 6 de obteniéndose resultados satisfactorios.

3.4.2 Tipo de pruebas

Pruebas de funcionalidad: son aquellas pruebas que tienen por objetivo probar que el sistema cumpla cumplan con las funciones específicas para los cuales ha sido creado.

Prueba de regresión: es cualquier tipo de prueba que se ejecute una vez realizado cambios en el sistema. Su objetivo es verificar la no regresión de la calidad luego de un cambio. Asegurar que los cambios no introducen un comportamiento no deseado u errores adicionales. Implican la re-ejecución de alguna o todas las pruebas realizadas.

3.4.3 Técnicas de prueba

Pruebas funcionales o Caja Negra: El método de caja negra es aplicado con el fin de verificar que las funciones son operativas a través de la interfaz del software, que la entrada se acepta de forma adecuada y que se produce un resultado correcto, manteniendo la integridad de la información externa. Estas pruebas 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.

Dentro del método de caja negra se utilizó la técnica de la Partición de Equivalencia, pues permite examinar los valores válidos e inválidos de las entradas existentes en el software y descubrir de forma inmediata una clase de errores. La partición equivalente se basa en la definición de casos de pruebas que descubran clases de errores, reduciendo así en número de clases de prueba que hay que desarrollar.

Prueba estructurales o Caja blanca: el objetivo de este método es diseñar casos de prueba para que se ejecuten, al menos una vez, todas las sentencias del programa, y todas las condiciones tanto en su vertiente verdadera como falsa. La técnica de diseño de pruebas seleccionada para realizar las pruebas de caja blanca es la **Prueba del Camino Básico** y se basa en obtener una medida de la complejidad del diseño procedimental de un programa. Los pasos a realizar para aplicar esta técnica son: representar el programa en un grafo de flujo, calcular la complejidad ciclomática, determinar el conjunto básico de caminos independientes y derivar los casos de prueba que fuerzan la ejecución de cada camino.

3.4.4 Diseño de casos de prueba

El diseño de casos de prueba tiene como objetivo encontrar la mayor cantidad de errores con el mínimo esfuerzo y tiempo posible. A continuación se muestra la tabla de la sección de pruebas “Adicionar origen de datos basado en XML, CSV y JSON” probada para el caso de uso Gestionar Orígenes de Datos.

Tabla 4: Sección de prueba para el caso de uso Adicionar Origen de Datos**.**

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad
SC1 Adicionar origen de datos basado en XML, CSV y JSON	EC 1.1 Adicionar Origen de datos introduciendo todos los datos correctamente.	Se adiciona el origen de datos correctamente del cual se va a diseñar el modelo de datos, para XML, CSV y JSON
	EC 1.2 Adicionar Origen de datos introduciendo datos incorrectos.	Se introducen algunos datos incorrectos y el sistema no adiciona el nuevo origen de d
	EC 1.3 Adicionar Origen de datos con datos nulos.	Se dejan algunos campos vacíos y el sistema no adiciona el nuevo origen de datos.
	EC 1.4 Seleccionar la opción Cancelar.	Se cancela la adición del origen de datos.
	EC 1.5 Adicionar Origen de datos con nombre ya existente.	El usuario intenta añadir un nuevo origen de datos con el nombre de un origen existente en el sistema.

A partir de esta descripción se detallan las variables que se encuentran asociadas al caso de prueba.

Tabla 5: Descripción de las variables

No.	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Nombre	campo de texto	No	Nombre que se le da al nuevo origen de datos que se adiciona. No admite caracteres extraños. Admite letras mayúsculas y minúsculas, caracteres numéricos y guión bajo. Debe comenzar obligatoriamente con caracter alfabético.
2	Archivo	campo de dirección del	No	Dirección donde se encuentra el archivo a utilizar como origen

		fichero		de datos.

Una vez realizada la descripción de las variables se procede a la creación de la matriz de datos donde se prueba la validez de los datos introducidos en el sistema para la sección “Adicionar origen de datos basado en XML, CSV y JSON”.

Tabla 6: Matriz de datos

Escenario	Variables(enumeradas según la descripción)		Respuesta del sistema	Resultado de la prueba	Flujo Central
	1	2			
EC 1.1 Adicionar Origen de datos introduciendo todos los datos correctamente.	V	V	El sistema muestra el mensaje “El origen de datos se ha añadido correctamente”	Satisfactorio	1- Acceder al módulo Diseñador de Modelos. 2- Seleccionar botón Adicionar en el menú superior Gestionar Orígenes de datos. 3- Seleccionar el origen de datos a añadir. 4- Introducir los datos. 5- Pulsar botón Aceptar.
EC 1.2 Adicionar Origen de datos introduciendo datos incorrectos.	I	V	El sistema no permite la entrada de caracteres incorrectos.	Satisfactorio	1- Acceder al módulo Diseñador de Modelos. 2- Seleccionar botón Adicionar en el menú superior Gestionar Orígenes de datos. 3- Seleccionar el tipo de origen de datos a añadir. 4- Introducir los datos. 5- Pulsar botón Aceptar.
	I	V	El sistema deshabilita el botón Aceptar, impidiendo que se adicione un nuevo origen de datos con nombre incorrecto.	Satisfactorio	
	V	N/A	El sistema deshabilita el botón Aceptar, impidiendo que se adicione un nuevo origen de datos sin seleccionar el archivo.	Satisfactorio	

EC 1.3 Adicionar Origen de datos con datos nulos.	I	I	El sistema mantiene deshabilitado el botón Aceptar, impidiendo que se pueda adicionar el origen de datos con campos vacíos.	Satisfactorio	1- Acceder al módulo Diseñador de Modelos. 2- Seleccionar botón Adicionar en el menú superior Gestionar Orígenes de datos. 3- Seleccionar el tipo de origen de datos a añadir. 4- Introducir los datos. 5- Pulsar botón Aceptar.
EC 1.4 Seleccionar la opción Cancelar.	V	V	El sistema cancela la adición del nuevo origen de datos.	Satisfactorio	1- Acceder al módulo Diseñador de Modelos. 2- Seleccionar botón Adicionar en el menú superior Gestionar Orígenes de datos. 3- Seleccionar el tipo de origen de datos a añadir. 4- Pulsar botón Cancelar.
EC 1.5 Adicionar Origen de datos con nombre ya existente.	I	V	El sistema no adiciona el origen de datos y muestra el mensaje de error "El origen de datos ya existe".	Satisfactorio	1- Acceder al módulo Diseñador de Modelos. 2- Seleccionar botón Adicionar en el menú superior Gestionar Orígenes de datos. 3- Seleccionar el tipo de origen de datos a añadir. 4-Introducir los datos. 5- Pulsar botón Aceptar.

3.4.5 Resultado de las pruebas

Durante la realización de las pruebas para el CU Gestionar origen de datos se realizaron 3 iteraciones arrojando los siguientes resultados: en la primera iteración se encontraron 4 NC con un impacto bajo relacionadas con la validación de datos de entrada y no correspondencia con la descripción del CU; en la segunda iteración se obtuvieron 3 NC y en la tercera iteración no se encontraron NC. La siguiente tabla muestra los resultados obtenidos

Tabla 7: Resultados de las pruebas para el CU Gestionar origen de datos

Fecha	Iteración	Caso de Prueba	Tipo de error	Descripción
11/05/2015	1	CU Gestionar origen de datos	Validación	1-El campo nombre acepta caracteres incorrectos. 2-El botón aceptar no se deshabilita cuando hay un campo vacío del formulario
			No correspondencia	1-En el requisitos Adicionar origen de datos CSV, al dar clic en cancelar no muestra un mensaje de confirmación. 2-Cuando selecciona un origen de datos XML se deshabilita el botón Modificar de la sección Gestionar Origen de Datos.
15/05/2015	2	CU Gestionar origen de datos	Funcional	1- Al eliminar un origen de datos el archivo no se elimina del directorio. 2- En el requisito Adicionar origen de datos JSON no muestra un mensaje de confirmación una vez concluido el proceso.
18/05/2015			Ortografía	En el requisito Modificar Origen de datos CSV aparece en el formulario la palabra “origen” con tilde.
20/05/2015	3	CU Gestionar origen de datos	----	

La realización de las pruebas confirmó la correcta implementación de los requisitos identificándose las no conformidades (NC) asociadas cada caso de uso. Se realizaron 3 iteraciones obteniéndose 10 NC en la

primera, 5 NC en la segunda y no se detectaron NC en la tercera. A continuación se muestra la tabla con los resultados:

Tabla 8: Resumen de las NC encontradas

Casos de Uso	Iteración 1	Iteración 2	Iteración 3
Gestionar origen de datos	4	3	-
Gestionar modelo de datos	3	1	-
Gestionar consultas	3	1	-

En la siguiente gráfica se puede apreciar los resultados obtenidos a partir de la tabla anterior.

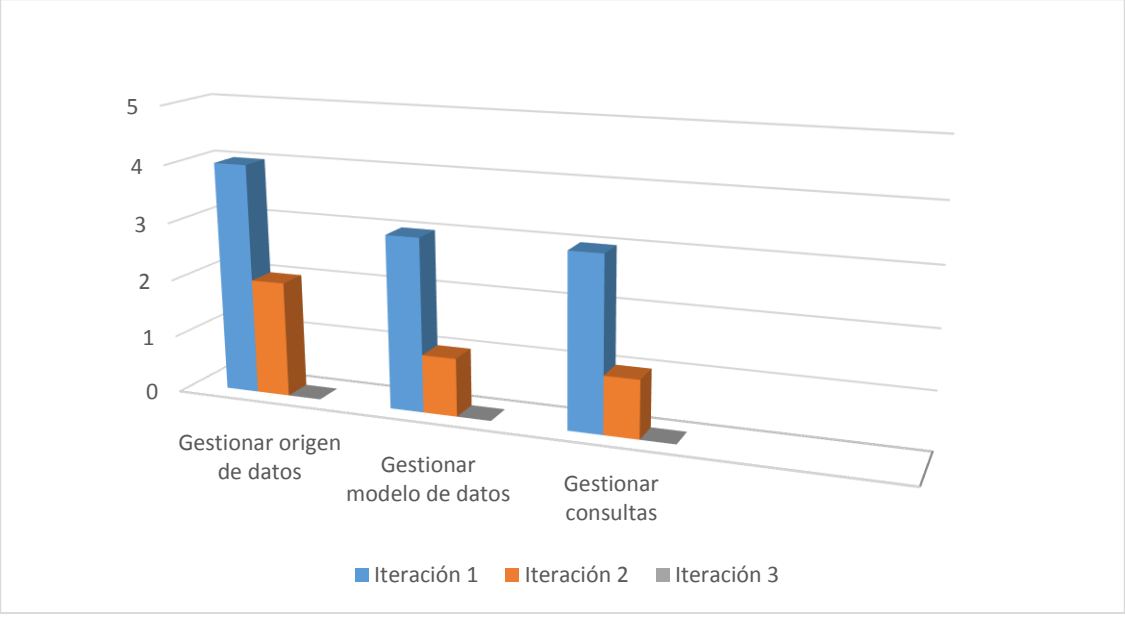


Fig. 10: Resultados de las pruebas

Conclusiones del capítulo

En este capítulo se llevó a cabo las actividades de implementación y prueba atendiendo a la metodología defina en el primer capítulo. Las mismas arrojaron los siguientes resultados:

- ✓ Se obtuvo el diagrama de componentes a partir del cual se logró definir la estructura general del sistema y el comportamiento de los componentes.
- ✓ Se estableció como estándares de codificación utilizados en el proyecto los definidos para Symfony 2 propiciando las reglas de escritura del código fuente, se obtuvo la programación de todos los requisitos identificados en el capítulo anterior y la descripción de sus clases, atributos y funcionalidades.
- ✓ La realización de las pruebas permitieron confirmar la correcta implementación de los requisitos definidos.

Conclusiones

Concluido el presente trabajo se puede afirmar que se logró el objetivo general propuesto. El cumplimiento de los objetivos específicos permitió arribar a las siguientes conclusiones:

- ✓ La investigación realizada sobre los Generadores de Reportes demostró la necesidad de implementar las funcionalidades que permitieran la generación de reportes a partir de orígenes de datos XML, CSV y JSON para GDRv2.0.
- ✓ La generación de los artefactos durante el desarrollo de la disciplina de Análisis y Diseño, proporcionaron la base para la implementación de la solución incidiendo notablemente en la estructura y organización del código.
- ✓ Con el desarrollo del presente trabajo se obtuvieron los componentes necesarios para la generación de reportes para GDR v2.0 a partir de los orígenes de datos XML, CSV y JSON otorgándole al sistema mayor relevancia.

Referencias bibliográficas

Bahit, Eugenia. 2011. *POO y MVC en PHP.* 2011.

Berthet, Charles. 1985. *Manual de informática.* Argentina : s.n., 1985. ISBN 950-02-5251-1.

Bonanata, Maximiliano. 2003. *Programación y algoritmos.* Buenos Aires, Argentina : MP Ediciones S.A, 2003. ISBN: 987-526-156-4 .

Brito Rodríguez, Julio César [et al.]. 2014. *Generador Dinámico de Reportes v2.0: desarrollo local.* 2014.

DSCallards. 2015. SAP Crystal solutions. [Online] 2015. [Cited: marzo 12, 2015.] http://www.crystalreports.co.uk/.

ECMA International. 2013. *ECMA-404 The JSON Data Interchange Standard.* 2013.

Fabien Potencier. 2015. Symfony. [En línea] 2015. http://symfony.com.

Fast Reports Inc. 2015. Fast Reports. [En línea] 2015. [Citado el: 20 de 4 de 2015.] https://www.fast-report.com/es/product/fast-report-vcl-5/.

Geeky Theory. 2013. GeekyTheory. [En línea] 13 de 10 de 2013. [Citado el: 24 de 3 de 2015.] https://geekytheory.com/json-i-que-es-y-para-que-sirve-json/.

Hamlett, Kenneth. 2014. La voz de Houston. [En línea] 10 de 12 de 2014. http://pyme.lavoztx.com/las-caractersticas-de-un-sistema-de-gestin-de-informacin-9451.html.

Larman, Craig. 1999. *UML y patrones. Introducción al análisis y diseño orientado a objetos.* México : Prentice Hall, 1999.

lenguajes-de-programacion.com. 2009. Lenguajes de Programacion. [En línea] 2009. http://www.lenguajes-de-programacion.com.

Microsoft. 2015. Microsoft Developer Network. [En línea] 2015. [Citado el: 20 de marzo de 2015.] https://msdn.microsoft.com/es-es/library/w558zwfc.aspx.

—. 2015. Microsoft Developer Network. [En línea] 2015. https://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx.

—. 2015. Micrososft SQL Server. [En línea] 2015. [Citado el: 3 de mayo de 2015.] https://technet.microsoft.com/es-es/library/ms166358%28v=sql.90%29.aspx.

Montero Ayala, Ramón. 2001. *XML Iniciación y referencia.* s.l. : McGraw-Hill, 2001. ISBN:84-481-2894-X.

Oracle. 2015. Netbeans. [En línea] 2015. [Citado el: 21 de 3 de 2015.] https://netbeans.org/index_es.html.

Pérez, Javier Eguíluz. 2008. *Introducción a Ajax.* 2008.

—. 2009. *Introducción a JavaScript.* 2009.

Potencier, Fabien. 2011. *Manual de Symfony2, Release 2.0.1.* 2011.

Pressman, Roger S. 2005. *Ingeniería de Software Un enfoque práctico 5ta edición.* 2005.

—. 2010. *Software Engineering: a practitioner's approach. Seventh Edition .* s.l. : McGraw-Hill, 2010. ISBN: 978-0-07-337 .

Programacion Desarrollo. 2015. Programacion Desarrollo. [En línea] 11 de 2 de 2015. [Citado el: 27 de 3 de 2015.] http://programaciondesarrollo.es/que-es-un-entorno-de-desarrollo-integrado-ide/.

Ruiz Heredia, Javier. 2011. *Comparación y tendencias entre metodologías ágiles y formales.* 2011.

Salas Cafferata, Daniel. 2009. CalidadySoftware.com. [En línea] 2009. [Citado el: 15 de 4 de 2015.] http://www.calidadysoftware.com/testing/casos_de_prueba.php.

Santos, Fernando. 2012. Excel Avanzado. [En línea] 14 de 4 de 2012. [Citado el: 26 de 3 de 2015.] http://www.excel-avanzado.com/1064/archivo-csv.html.

Sencha. 2015. Sencha. [Online] 2015. http://www.sencha.com/.

Shea Frederick, Colin Ramsay. 2008. *Learning Ext JS.* BIRMINGHAM - MUMBAI : Packt Publishing, 2008. ISBN 978-1-847195.

Sommerville, Ian. 2005. *Ingeniería del software Séptima Edición.* United Kingdom : s.n., 2005. ISBN:84-7829-074-5.

Stimulsoft. 2015. Stimulsoft Reports.Web. [En línea] 2015. [Citado el: 10 de marzo de 2015.] http://www.stimulsoft.com/es/products/reports-web.

Symfony.es. 2015. symfony.es. [En línea] 2015. http://symfony.es/que-es-symfony.

TIBCO Software Inc. 2015. Jaspersoft Community. [Online] 2015. [Cited: abril 15, 2015.] http://community.jaspersoft.com/project/jasperreports-library.

Vázquez, José Antonio Gallego. 2003. *Desarrollo Web con PHP y MySQL.* Madrid : Anaya Multimedia, 2003. ISBN: 84-415-1525-5.

Visual Paradigm. 2015. Visual Paradigm. [En línea] 2015. [Citado el: 15 de 3 de 2015.] http://www.visual-paradigm.com.

Walsh, Norman. XML.com. [Online] [Cited: 11 15, 2014.] http://www.xml.com/pub/a/98/10/guide0.html?page=2#AEN63.

World Technology Limited. 2015. PHP Report Maker. [En línea] 2015. [Citado el: 7 de febrero de 2015.] http://www.hkvstore.com/phpreportmaker/.

Bibliografía

Bahit, Eugenia. 2011. *POO y MVC en PHP.* 2011.

Berthet, Charles. 1985. *Manual de informática.* Argentina : s.n., 1985. ISBN 950-02-5251-1.

Bonanata, Maximiliano. 2003. *Programación y algoritmos.* Buenos Aires, Argentina : MP Ediciones S.A, 2003. ISBN: 987-526-156-4 .

Brito Rodríguez, Julio César [et al.]. 2014. *Generador Dinámico de Reportes v2.0: desarrollo local.* 2014.

DSCallards. 2015. SAP Crystal solutions. [Online] 2015. [Cited: marzo 12, 2015.] http://www.crystalreports.co.uk/.

ECMA International. 2013. *ECMA-404 The JSON Data Interchange Standard.* 2013.

Fabien Potencier. 2015. Symfony. [En línea] 2015. http://symfony.com.

Fast Reports Inc. 2015. Fast Reports. [En línea] 2015. [Citado el: 20 de 4 de 2015.] https://www.fast-report.com/es/product/fast-report-vcl-5/.

Geeky Theory. 2013. GeekyTheory. [En línea] 13 de 10 de 2013. [Citado el: 24 de 3 de 2015.] https://geekytheory.com/json-i-que-es-y-para-que-sirve-json/.

Hamlett, Kenneth. 2014. La voz de Houston. [En línea] 10 de 12 de 2014. http://pyme.lavoztx.com/las-caractersticas-de-un-sistema-de-gestin-de-informacin-9451.html.

Larman, Craig. 1999. *UML y patrones. Introducción al análisis y diseño orientado a objetos.* México : Prentice Hall, 1999.

lenguajes-de-programacion.com. 2009. Lenguajes de Programacion. [En línea] 2009. http://www.lenguajes-de-programacion.com.

Microsoft. 2015. Microsoft Developer Network. [En línea] 2015. [Citado el: 20 de marzo de 2015.] https://msdn.microsoft.com/es-es/library/w558zwfc.aspx.

—. 2015. Microsoft Developer Network. [En línea] 2015. https://msdn.microsoft.com/es-es/library/aa291591%28v=vs.71%29.aspx.

—. 2015. Micrososft SQL Server. [En línea] 2015. [Citado el: 3 de mayo de 2015.] https://technet.microsoft.com/es-es/library/ms166358%28v=sql.90%29.aspx.

Montero Ayala, Ramón. 2001. *XML Iniciación y referencia.* s.l. : McGraw-Hill, 2001. ISBN:84-481-2894-X.

Oracle. 2015. Netbeans. [En línea] 2015. [Citado el: 21 de 3 de 2015.] https://netbeans.org/index_es.html.

Pérez, Javier Eguíluz. 2008. *Introducción a Ajax.* 2008.

—. 2009. *Introducción a JavaScript.* 2009.

Potencier, Fabien. 2011. *Manual de Symfony2, Release 2.0.1.* 2011.

Pressman, Roger S. 2005. *Ingeniería de Software Un enfoque práctico 5ta edición.* 2005.

—. 2010. *Software Engineering: a practitioner's approach. Seventh Edition .* s.l. : McGraw-Hill, 2010. ISBN: 978-0-07-337 .

Programacion Desarrollo. 2015. Programacion Desarrollo. [En línea] 11 de 2 de 2015. [Citado el: 27 de 3 de 2015.] http://programaciondesarrollo.es/que-es-un-entorno-de-desarrollo-integrado-ide/.

Ruiz Heredia, Javier. 2011. *Comparación y tendencias entre metodologías ágiles y formales.* 2011.

Salas Cafferata, Daniel. 2009. CalidadySoftware.com. [En línea] 2009. [Citado el: 15 de 4 de 2015.] http://www.calidadysoftware.com/testing/casos_de_prueba.php.

Santos, Fernando. 2012. Excel Avanzado. [En línea] 14 de 4 de 2012. [Citado el: 26 de 3 de 2015.] http://www.excel-avanzado.com/1064/archivo-csv.html.

Sencha. 2015. Sencha. [Online] 2015. http://www.sencha.com/.

Shea Frederick, Colin Ramsay. 2008. *Learning Ext JS.* BIRMINGHAM - MUMBAI : Packt Publishing, 2008. ISBN 978-1-847195.

Sommerville, Ian. 2005. *Ingeniería del software Séptima Edición.* United Kingdom : s.n., 2005. ISBN:84-7829-074-5.

Stimulsoft. 2015. Stimulsoft Reports.Web. [En línea] 2015. [Citado el: 10 de marzo de 2015.] http://www.stimulsoft.com/es/products/reports-web.

Symfony.es. 2015. symfony.es. [En línea] 2015. http://symfony.es/que-es-symfony.

TIBCO Software Inc. 2015. Jaspersoft Community. [Online] 2015. [Cited: abril 15, 2015.] http://community.jaspersoft.com/project/jasperreports-library.

Vázquez, José Antonio Gallego. 2003. *Desarrollo Web con PHP y MySQL.* Madrid : Anaya Multimedia, 2003. ISBN: 84-415-1525-5.

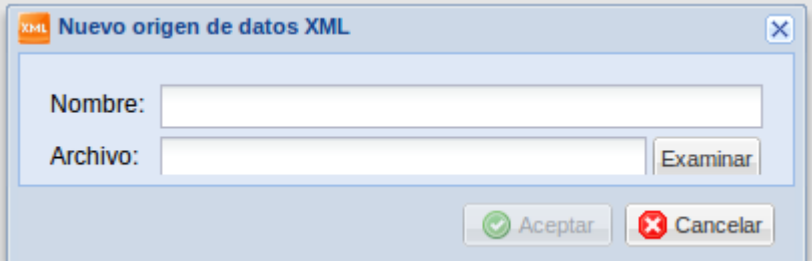
Visual Paradigm. 2015. Visual Paradigm. [En línea] 2015. [Citado el: 15 de 3 de 2015.] http://www.visual-paradigm.com.

Walsh, Norman. XML.com. [Online] [Cited: 11 15, 2014.] http://www.xml.com/pub/a/98/10/guide0.html?page=2#AEN63.

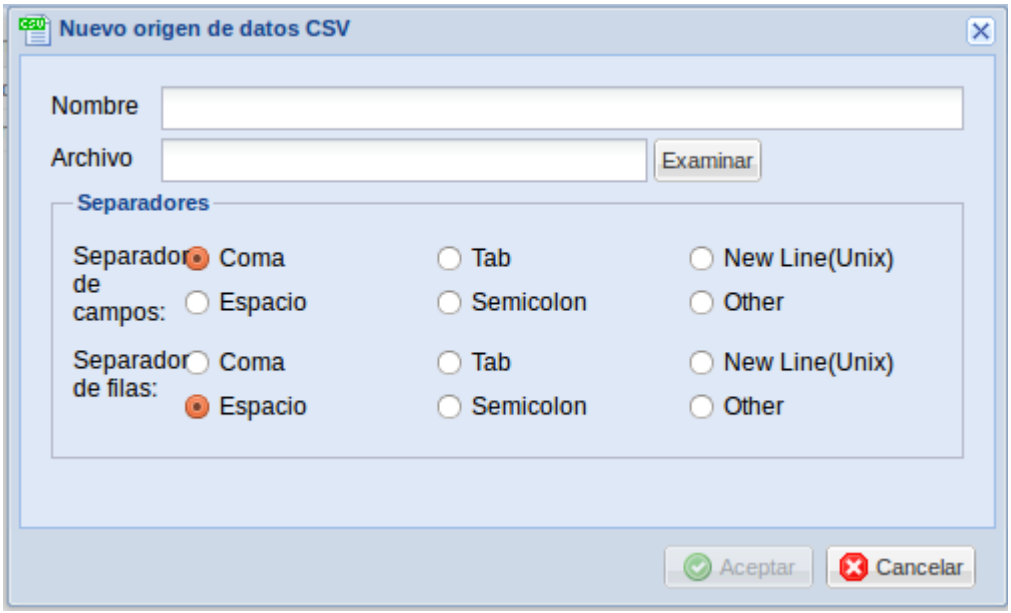
World Technology Limited. 2015. PHP Report Maker. [En línea] 2015. [Citado el: 7 de febrero de 2015.] http://www.hkvstore.com/phpreportmaker/.

Anexos

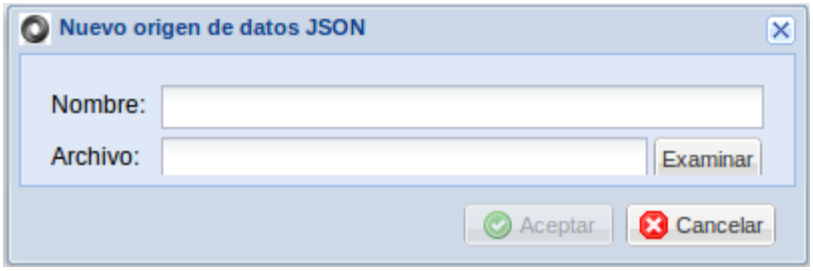
Anexo 1: Interfaz de la aplicación. Adicionar Origen de Datos XML



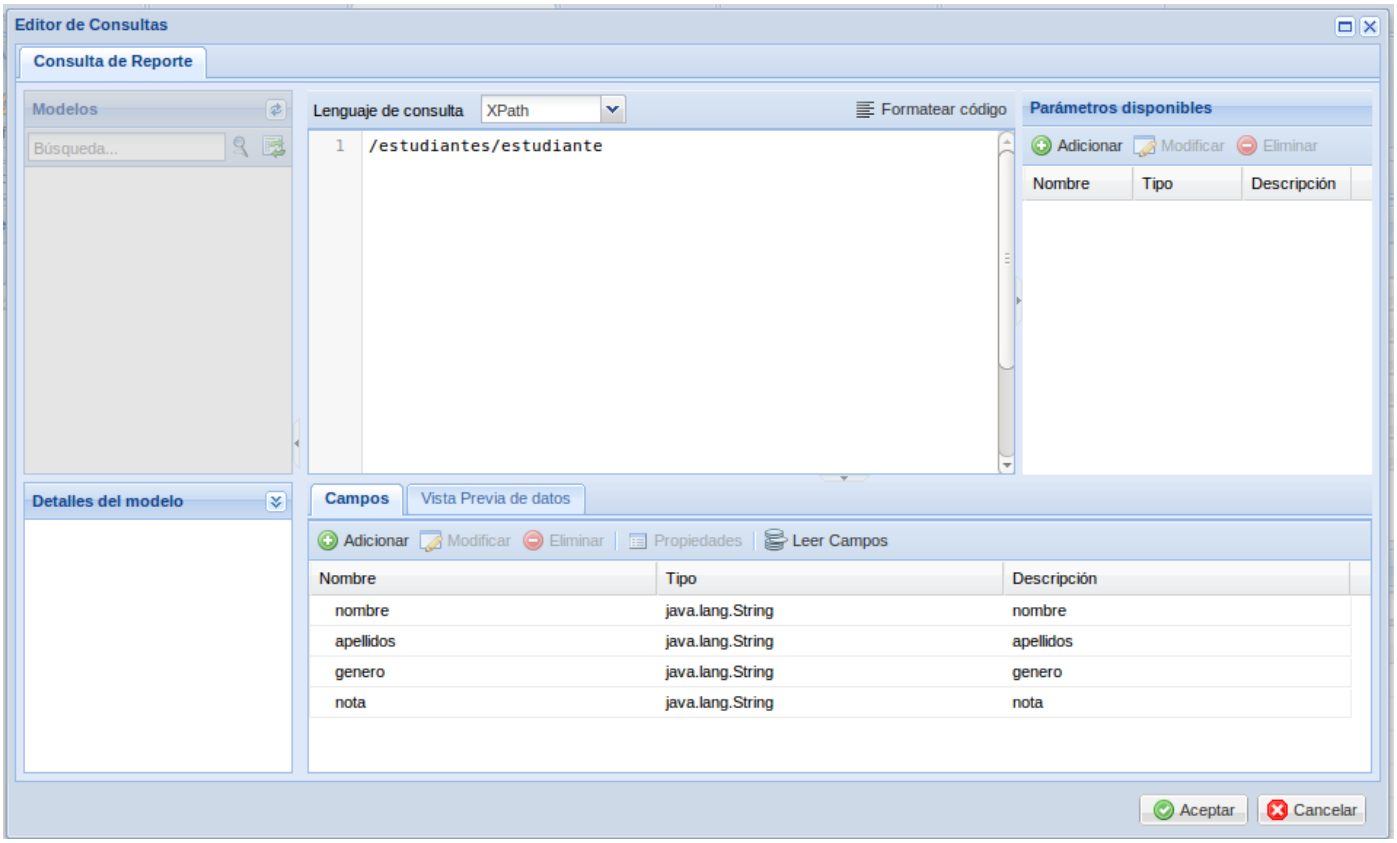
Anexo 2: Interfaz de la aplicación. Adicionar Origen de datos CSV



Anexo 3: Interfaz de la aplicación. Adicionar Origen de datos JSON



Anexo 4: Interfaz de la aplicación. Editor de consultas



Glosario de términos

CASE: Ingeniería de Software Asistida por Computación.

DATEC: Centro de Tecnologías de Gestión de Datos.

GRASP: Patrones de Software para la Asignación General de Responsabilidad.

IDE: Entorno Desarrollo Integrado.

JSON: Notación de Objetos de JavaScript.

MVC: Modelo – Vista – Controlador.

OpenUP: Metodología para el proceso del desarrollo del software.

UCI: Universidad de las Ciencias Informáticas.

UML: Lenguaje Unificado de Modelado.

SGI: Sistema de Gestión de Información

XML: Lenguaje de marcas extensible

CSV: valores separados por coma