Universidad de las Ciencias Informáticas

Facultad 6

Título: "Asistente de Reportes para el módulo Diseñador de Reportes del Generador Dinámico de Reportes versión 2.0"

Trabajo de diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autora: Ana Niuska Navarro Rodríguez

Tutor: Ing. Sergio Jesús García de la Puente

Co-tutor: Ing. Miguel Lezcano Ramos

Junio del 2011

DECLARACIÓN DE AUTORÍA

Declaro que soy la única autora de este trabajo y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Ana Niuska Navarro Rodríguez	Ing. Sergio Jesús García de la Puente
_____	_____
Firma de la Autora	Firma del Tutor

DATOS DE CONTACTO

Tutores:

- Ing. Miguel Lezcano Ramos

Email: mlezcano@uci.cu

Universidad de las Ciencias Informáticas

- Ing. Sergio Jesús García de la Puente

Email: sgarcia@uci.cu

Universidad de las Ciencias Informáticas

AGRADECIMIENTOS

A la Revolución por darme la oportunidad de formarme como profesional en esta universidad que nunca olvidaré.

A los profesores que he tenido durante toda la carrera, a los cuales admiro y respeto mucho: Anthony, Yuraysi, Sandy, Ileane y Nara.

A mi tutor Miguel, por su paciencia, por enseñarme todo lo que he aprendido en la realización de este trabajo, y por estar a mi lado siempre ayudándome en todo.

A mi tutor Sergio, por su apoyo incondicional y por demostrarme en todo momento que yo podía lograrlo.

A mis amigos, por haber compartido tantas experiencias juntos y estar a mi lado en los momentos más difíciles. Yaneisy González, Yaneisy Pedraza, Wilmis, Alianny, Eliza, Dainelis, Ailyn, Themis, Susel, Yiri, Katia, Glennis, Ariadna, Yuya, Yayi, Yili, Sura, Roberto, Salvador Luis Grabiel, Yariel, Omar, Henrry, Juaka, Armando, Yoandry y Yandy.

A mi grandísima familia, que siempre ha estado pendiente de mi vida y mis estudios.

A mi mamá, por ser la persona más noble que existe, por su preocupación, sus consejos y por estar siempre orgullosa de mí.

A mi papá, por parecerme a él en todo, por educarme y ser más que un ejemplo para mí.

A mi hermano, por su infinito amor y estar siempre dispuesto a ayudarme en todo desde que era un niñito.

A mi novio Sergio, por estar a mi lado en todo momento durante este último año, queriéndome y haciéndome muy feliz.

A mis suegros, por su preocupación y por acogerme como una hija más.

A todas las personas que de una forma u otra colaboraron en el desarrollo de este trabajo.

DEDICATORIA

A mi familia, por ser las personas que más amo, por ser especiales para mí y llenar mis días de felicidad.

A mi abuela Estela, por quererme tanto y enseñarme desde pequeña las cosas más importantes de la vida.

A mi primo Dariel, que siempre tenía una sonrisa para todos y hubiese estado muy orgulloso de mí.

RESUMEN

Los Generadores de Reportes son herramientas complementarias de los sistemas de información. Proveen una forma transparente al usuario para realizar consultas a la base de datos y obtener información de ella en forma de reporte. Los Asistentes de diseño predeterminado de reportes son herramientas complementarias de los sistemas Generadores de Reportes.

El presente trabajo de diploma describe la realización del Asistente de diseño predeterminado de reportes para el Sistema Generador Dinámico de Reportes, versión 2.0, donde se analizaron los diferentes tipos de reportes predeterminados que proveen los principales sistemas de generación de reportes, con el propósito de determinar los tipos de reportes que debía permitir diseñar el Asistente de Reporte. Además, se realizó el análisis y diseño de los tipos de reportes que se determinaron incluir, así como también la implementación y prueba del reporte predeterminado de Tabla Pivote.

El resultado fundamental se centra en la obtención de un Asistente de Reportes que se integre a la arquitectura base del sistema Generador Dinámico de Reportes y permita guiar a cualquier tipo de usuario mediante operaciones básicas y sencillas durante todo el proceso de diseño de reportes, facilitando un ahorro significativo de tiempo y esfuerzo.

Palabras Clave: Asistente de Reportes, Generadores de Reportes.

TABLA DE CONTENIDOS

INTRODUCCIÓN

En la actualidad, la mayoría de las instituciones requieren ser informatizadas, debido a la cantidad de información que estas procesan para su desempeño. Generalmente, dichas organizaciones necesitan de un mecanismo que les permita proporcionar información útil en tiempo real, por lo que existen varias herramientas que permiten a los usuarios obtener con facilidad datos de archivos o bases de datos de manera rápida en forma de reportes, entre las que se encuentran: Active Reports, iReport y Crystal Reports.

Cuba, a pesar de ser un país subdesarrollado y bloqueado, se encuentra inmersa en el avance de la informática y en el proceso de informatización de la sociedad, con el objetivo de propiciar un mayor desarrollo en el campo de la producción de software.

La Industria Cubana del Software (ICSW), pretende convertirse en una significativa fuente de ingresos nacionales, como resultado del correcto aprovechamiento de las ventajas del capital humano disponible.

La Universidad de las Ciencias Informáticas (UCI) surge como un proyecto del Gobierno cubano, denominado al principio "Proyecto Futuro", con dos objetivos esenciales: informatizar el país y desarrollar la industria cubana del Software para contribuir al desarrollo económico del mismo. Es la primera universidad creada bajo los propósitos de la Batalla de Ideas, que, cuenta con un plan de estudio diferente del resto de los centros de Educación Superior del país, siguiendo el principio martiano de la vinculación Estudio-Trabajo, donde se hace énfasis en la producción como parte del proceso de aprendizaje, implementando un sistema de desarrollo de Software, el cual dispone de una amplia Infraestructura Productiva, donde la producción se realiza, fundamentalmente, en los centros de desarrollo, los cuales garantizan la aplicación de tecnologías específicas y la investigación asociada,

así como también la formación universitaria desde la producción.

El Centro de Tecnologías de Gestión de Datos (DATEC) es un centro de desarrollo productivo que cuenta con cuatro departamentos:

- Departamento de Desarrollo de tecnologías de bases de datos PostgreSQL.
- Departamento de Almacenes de datos y soluciones de inteligencia de negocios.
- Departamento de Integración de soluciones.
- Departamento de Bioinformática.

DATEC tiene como objetivos principales:

- Proveer soluciones integrales y consultorías relacionadas con tecnologías de bases de datos y análisis de la información.
- Desarrollar nuevas tecnologías de bases de datos, de procesamiento y representación de la información a partir del desarrollo de proyectos de I+D (enfoque a la soberanía tecnológica).
- Contribuir con su trabajo al cumplimiento de las misiones fundamentales de la universidad: (la formación y la producción de software) con profesionales integrales y comprometidos con un alto nivel científico y productivo.

Para lograr el cumplimiento de estos objetivos, DATEC desarrolla en la actualidad varios productos. Específicamente, el departamento de Integración de soluciones se encuentra desarrollando el producto Plataforma de Ayuda a la Toma de Decisiones y Soluciones Integradas (PATDSI), que consiste en una plataforma integral para el análisis de datos e incluye otras aplicaciones tales como:

- Generador Dinámico de Reportes (GDR).
- Módulo para la Minería de Datos.
- Módulo de procesamiento estadístico (SIGE).

El Generador Dinámico de Reportes es una aplicación que combina las ventajas que ofrecen las aplicaciones Web y las tradicionales o de escritorio. Utiliza una especie de lenguaje transparente para el usuario por medio del cual este realiza consultas a la base de datos y obtiene información de ella en forma de reporte. Funciona como un sistema independiente, que se conecta a diferentes sistemas gestores de bases de datos tales como Oracle, MySQL, PostgresSQL, Microsoft SQL Server, SQLite y MySQL, y de esta forma permite confeccionar reportes de forma rápida, interactiva y con una amplia gama de alternativas para los usuarios. Presenta una arquitectura basada en componentes, que permite distribuir sus funcionalidades por módulos. Dichas partes se complementan entre sí, haciendo más fácil la generación de reportes, debido a que aumentan la reutilización y reducen la redundancia de información. Está compuesto por cinco módulos que son:

- Diseñador de modelos.
- Diseñador de reportes.
- Diseñador de consultas.
- Visor de reportes.
- Administrador de reportes.

Las instituciones requieren de una manera centralizada de crear, administrar y entregar reportes en tiempo real. El GDR permite a las organizaciones encargarse de este proceso, pero no todos los usuarios son capaces de diseñar reportes de manera rápida y con la calidad requerida, debido al nivel de complejidad que la aplicación puede presentar para la creación de un reporte, por lo que es necesario crear un mecanismo que permita guiar a cualquier tipo de usuario durante todo el proceso de diseño, facilitando un ahorro significativo de tiempo y esfuerzo mediante operaciones sencillas y básicas. Según lo planteado anteriormente, surge el siguiente **Problema a resolver:** ¿Cómo proveer un nuevo mecanismo de diseño de reportes con una baja participación del usuario que se integre a la

arquitectura base del sistema Generador Dinámico de Reportes?

Como **Objeto de estudio** se define: El proceso de diseño predeterminado de reportes de los sistemas de generación dinámica de reportes, enmarcado en el **Campo de acción:** Asistentes de diseño predeterminado de reportes.

Para darle solución al problema se plantea el siguiente **Objetivo general:** Desarrollar el Asistente de diseño predeterminado de reportes para el Sistema Generador Dinámico de Reportes, versión 2.0.

Derivando el Objetivo general se plantean los siguientes **Objetivos específicos:**

1. Analizar los diferentes tipos de reportes predeterminados que proveen los principales sistemas de generación de reportes.
2. Documentar el análisis y diseño del Asistente de diseño predeterminado de reportes del sistema Generador Dinámico de Reportes, versión 2.0.
3. Validar el diseño, implementando el Asistente de diseño predeterminado de reportes, con un reporte predeterminado.

Para lograr el cumplimiento del objetivo general se plantearon las siguientes **Tareas de la investigación:**

1. Identificación de los diferentes tipos de reportes predeterminados que proveen los principales sistemas de generación dinámica de reportes.
2. Selección de los tipos de reportes predeterminados que se incluirán en el sistema Generador Dinámico de Reportes, versión 2.0.
3. Descripción del XML del motor de generación de reportes, al que se debe ajustar el Asistente de diseño predeterminado de reportes, del sistema Generador Dinámico de Reportes, versión 2.0.

4. Realización del análisis con vista a documentar el modelo de casos de uso del sistema.
5. Identificación del diseño de clases de la máquina de estados que dará soporte al Asistente de diseño predeterminado de reportes.
6. Implementación de la Máquina de estados.
7. Selección de un reporte predeterminado para asociarlo a la Máquina de estados.
8. Documentación de las pruebas.

Métodos científicos de la investigación:

Analítico – Sintético: Se utiliza para analizar la teoría y los documentos relacionados con el proceso de diseño de los principales sistemas de generación dinámica de reportes, identificar los diferentes tipos de reportes predeterminados que estos proveen, así como también seleccionar los reportes que se incluirán en el sistema Generador Dinámico de Reportes, versión 2.0, mediante la extracción de las características generales, relaciones esenciales y los elementos más importantes.

Inductivo – Deductivo: Se utiliza para determinar las características generales que reflejan lo común en los diferentes tipos de reportes predeterminados y aplicarlas en la implementación del Asistente de Reportes.

Modelación: Permite la creación de los modelos que pertenecen al análisis, diseño e implementación del Asistente de reportes, el modelado del diagrama de casos de uso del sistema, diagramas de clases del diseño, diagramas de interacción, diagramas de componentes, entre otros.

Observación: Se utiliza para el estudio del proceso de diseño predeterminado de reportes de los Asistentes de los principales sistemas de generación dinámica de reportes.

Estructura de la Tesis:

Fundamentación teórica

Estudio del estado del arte de las herramientas de diseño de reportes existentes, centrando el análisis en los Asistentes de diseño predeterminados de reportes.

Análisis de la solución

Análisis de la solución planteada: Modelo de dominio, Definición de requisitos (funcionales y no funcionales), Modelo del Sistema.

Diseño de la solución

Diseño de la solución planteada: Descripción de los componentes, Estilo arquitectónico y patrones de diseño utilizados, Modelo del Diseño.

Implementación y Pruebas de la solución

Implementación y Pruebas de la solución planteada: Implementación del Asistente de Reportes reutilizando componentes e implementando el nuevo componente genérico Máquina de estados, Modelo de Implementación. Validación de la solución mediante las pruebas a Nivel de Desarrollador.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Introducción

En este capítulo se estudian los elementos relacionados con el estado del arte de los principales sistemas de generación de reportes existentes, centrando el análisis en los Asistentes de diseño predeterminado de reportes, herramientas complementarias de estos sistemas, que permiten a los usuarios agilizar enormemente el proceso de creación de reportes, además, se presentan las tecnologías a utilizar para dar solución al problema.

Reportes en el ámbito de la informática

Figura 1: Reportes Informáticos

Un reporte es un informe o una noticia. Este tipo de documento puede ser impreso, digital o audiovisual, pretende transmitir una información, aunque puede tener diversos objetivos. Existen reportes divulgativos, persuasivos y de otros tipos, específicamente en el ámbito de la informática los reportes son informes que organizan y exhiben la información contenida en una base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. Los reportes tienen diversos niveles de complejidad, desde una lista o enumeración hasta gráficos mucho más desarrollados. Según la herramienta

informática y la base de datos en cuestión, los reportes permiten la creación de etiquetas y la elaboración de facturas, entre otras tareas. *(Daily WP. 2008)*

1.1 Sistemas de generación de reportes

Los generadores de reportes son sistemas prácticos de alto nivel, incluidos en la mayoría de los productos de software empresariales de los sistemas de información. Se encargan de diseñar reportes para brindar información necesaria a los usuarios, la cual se obtiene a través de la realización de consultas a las bases de datos. A continuación se realiza un estudio de los principales Sistemas de generación de reportes existentes.

1.1.1 Crystal Reports

Crystal Reports es un producto de inteligencia empresarial, con alta tecnología para la creación e integración de reportes, con datos provenientes de múltiples SGBD (Database Management System (Tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan)), entre los que se encuentran: PostgreSQL, MySQL, Oracle, DB2, MS-SQL, Informix, InterBase, Sybase y Frontbase. Permite crear contenido interactivo con calidad de presentación en la plataforma .NET (Framework (Marco de trabajo) de Microsoft que permite un rápido desarrollo de aplicaciones), debido a que es la herramienta de elaboración de informes estándar para Visual Studio .NET, lo que ha supuesto una ventaja fundamental para Crystal Reports durante años. *(SAP Business Objects 2008a)*

Es un líder comprobado en el diseño de reportes, que cumple los desafíos que día a día enfrentan los analistas de negocio y los desarrolladores, permite transformar de manera rápida cualquier fuente de datos en contenido interactivo, integrar estrechamente capacidades de diseño, modificación y

visualización en aplicaciones .NET o Java (Lenguaje de programación orientado a objetos), permitiendo a los usuarios finales acceder e interactuar con los reportes a través de portales Web, dispositivos móviles y documentos de Microsoft Office. *(SAP Business Objects 2008a)*

Herramienta Complementaria: Asistente de Diseño Predeterminado de Reportes.

Sistema Operativo: Microsoft Windows (Serie de sistemas operativos desarrollados por Microsoft desde 1981 como: Windows XP, Windows Server 2003, Windows NT, Windows 98, Windows 2000, Windows Vista, entre otros) y UNIX (Sistema operativo portable, multitarea y multiusuario como: Linux, Solaris, AIX, HP-UX, Mac OS X, entre otros).

Licencia: Crystal Reports dispone de licencias por usuario designado para el diseño de reportes, independientemente de la edición de Crystal Reports adquirida, se distribuye bajo los términos de la EULA ((End User Licensing Agreement (Licencia por la cual el uso de un producto sólo está permitido para un único usuario (el comprador))), por lo que es software privativo y de uso restringido mediante el pago de patente. *(SAP Business Objects 2008b)*

1.1.2 iReport

iReport es una herramienta visual de diseño de informes para JasperReports (Herramienta libre de Java para la creación de informes que tiene la habilidad de entregar contenido enriquecido al monitor, a la impresora o a ficheros PDF (Portable Document Format (Formato de Documento Portátil)), HTML (HyperText Markup Language (Lenguaje de Marcado de Hipertexto)), XLS (Acrónimo de Microsoft Excel Spreadsheet perteneciente a la categoría Extensión de archivo), CSV (Comma-Separated Values (Tipo de documento en formato abierto)) y XML (Extensible Markup Language (Lenguaje de Marcas Extensible)), potente, con una interfaz gráfica intuitiva y fácil de usar. Permite a los usuarios editar visualmente cualquier tipo de informe complejo con gráficas, imágenes y subinformes de manera

sencilla y rápida, tanto para usuarios que no están familiarizados con esta tecnología y que desconocen la sintaxis del XML de JasperReports (Librería de Java que combinada con herramientas para el diseño, facilita y agiliza la generación, la previsualización y la impresión de los reportes), como a usuarios expertos que ya conocían este lenguaje, ahorrándoles tiempo durante el desarrollo de informes muy elaborados. Los datos a visualizar pueden ser recogidos de diferentes maneras, incluyendo conexiones a múltiples JDBC (Java Database Connectivity (API (Interfaz de Programación de Aplicaciones) que permite la ejecución de operaciones sobre bases de datos desde el lenguaje de programación Java)), Modelos de Entidades (Herramienta para el modelado de datos de un sistema de información), JavaBeans (Componentes de software reutilizables que se puedan manipular visualmente en una herramienta de construcción), XML, Hibernate (Herramienta de Mapeo objeto-relacional (ORM)), además, la previsualización de los informes en varios formatos, como PDF, XLS o RTF (Rich Text Format (Formato de Texto Enriquecido)). *(Motion Control Engineering 2008)*

Las novedades más significativas con respecto a la última versión homologada de iReport 2.0.5 son: *(Ignacio López Vellon 2010)*

- Soporte completo para JasperReports 2.0.5.
- Soporte completo para la internacionalización (i18n).
- Conexión a Hibernate mediante Spring (Framework de código abierto de desarrollo de aplicaciones para la plataforma Java).
- Exportador para formatos de OpenOffice (Suite ofimática libre (Recopilación de programas, los cuales son utilizados en oficinas) de código abierto y distribución gratuita que incluye herramientas como procesador de textos, hoja de cálculo, presentaciones, herramientas para el dibujo vectorial y base de datos).
- Soporte completo para archivos jrtx (plantillas de estilo).

- Actualizaciones de varias librerías que incorpora iReport.
- Soporte completo para la exportación de ficheros swf (flash).
- Numerosos bugs (Defecto de software) arreglados.

Herramienta Complementaria: Asistente de Diseño Predeterminado de Reportes.

Sistema Operativo: Windows 2000, NT y XP.

Licencia: Un programa OpenSource (Software distribuido y desarrollado libremente), con licencia GPL (Licencia Pública General), se licencia como Freeware (Tipo de software que se distribuye sin coste, disponible para su uso y por tiempo ilimitado) para el sistema operativo Windows. iReport se ofrece como una descarga gratuita a todos los usuarios de software Freeware. *(iReport - Dev - Confluence 2008)*

1.1.3 Pentaho Reporting

La unidad de reportes de Pentaho (Pentaho Reporting) permite a las organizaciones acceder, dar formato y distribuir fácilmente la información a empleados, clientes y asociados. Pentaho provee acceso a fuentes de datos relacionales, OLAP (On-Line Analytical Processing (Procesamiento Analítico en Línea cuyo objetivo es agilizar la consulta de grandes cantidades de datos)) o basadas en XML, además de ofrecer varios formatos de salida como PDF, HTML, Excel o hasta texto plano. Lleva la información a los usuarios finales vía web, e-mail, portales corporativos o aplicaciones propias. Pentaho Reporting puede ir incrementando la plataforma de reportes a medida que las necesidades crecen. *Vladimir Díaz Ordaz 2011*

Pentaho Reporting abarca tres productos con diferentes enfoques dirigidos a distintos tipos de usuarios. *(Josep Curto Díaz 2010)*

- **Pentaho Report Designer:** Editor basado en Eclipse (Entorno de desarrollo integrado de código abierto multiplataforma) con prestaciones profesionales y de calidad, con capacidad de personalización de informes a las necesidades de negocio destinado a desarrolladores. Incluye Asistentes para facilitar la configuración de propiedades. Está estructurado de forma que los desarrolladores puedan acceder a sus prestaciones de forma rápida. Incluye un editor de consultas para facilitar la confección de los datos que serán utilizados en un informe. **Pentaho Report Designer** incorpora un Asistente de reportes muy útil, incluido en la última versión (1.6.0) que agiliza enormemente la creación de reportes a los usuarios, proporcionando una guía con siete pasos para el diseño.
- **Pentaho Report Design Wizard:** Herramienta de diseño de informes, que facilita el trabajo y permite a los usuarios obtener resultados de forma inmediata. Está destinada a usuarios con menos conocimientos técnicos.
- **Web ad-hoc reporting:** Es el similar a la herramienta anterior, pero vía web. Extiende la capacidad de los usuarios finales para la creación de informes a partir de plantillas preconfiguradas y siguiendo un Asistente de creación.

Sistema Operativo: No existe dependencia de ningún sistema operativo.

Licencia: El software es gratuito y puede ser apoyado y desarrollado.

1.1.4 Report Manager

El Administrador de informes (Report manager) es un acceso a los informes basados en Web y una herramienta de gestión que se utiliza para administrar una instancia del servidor de informes desde una ubicación remota a través de una conexión HTTP (Hypertext Transfer Protocol (Protocolo de transferencia de hipertexto)). Aplicación de generación de informes que incluye un Servidor de

informes TCP (Transmission Control Protocol (Protocolo de Control de Transmisión fundamentales en Internet)), a través del cual los clientes obtienen informes procesados en el servidor. También se incluye un completo servidor web de informes, generando archivos Adobe PDF, potente servidor de informes de red sin pago de licencias y con soporte para máquinas multiprocesador. *(Página Oficial de Report Manager 2008)*

Report Manager dispone de múltiples características, incluyendo algunas exclusivas, como:

Bibliotecas de informes, metaarchivos, fuentes de impresora, secciones externas y subinformes hijos (subinformes en cascada). Si utiliza el IDE (Integrated Development Environment (Entorno de Desarrollo Integrado, programa informático compuesto por un conjunto de herramientas de programación)) Delphi, Kylix, o Builder, puede incluir el motor de informes en sus ejecutables, presentación preliminar, diálogo de impresión, y opciones del informe.

Report manager es un conjunto de componentes para Delphi, Builder y Kylix. También puede utilizarse desde otros entornos de desarrollo con el componente ActiveX incluido (Visual Basic,Visual FoxPro, cualquier lenguaje de Visual Studio.Net). *(Página Oficial de Report Manager 2008)*

Sistema Operativo: Funciona en Windows y Linux.

Licencia: Producto Open Source bajo el modelo MPL (Mozilla Public License (Licencia Pública de Mozilla de código abierto y de software libre)) se incluye permiso de uso en aplicaciones GPL, por lo que se puede usar en aplicaciones comerciales, pero cualquier mejora introducida en el motor de impresión debe ser publicada bajo esta licencia.

1.1.5 ActiveReports

ActiveReports cuenta con las premiadas capacidades mundialmente significativas que proveen la habilidad de diseñar, crear y desplegar aplicaciones de reportes de forma fácil y rápida.

- Está diseñado teniendo en cuenta los desarrolladores y sus diversas necesidades. Soporta el uso de componentes .NET en tiempo de diseño.
- Escrito completamente en Visual C# y provee una completa integración con Visual Studio .NET.
- Asistente para la conversión de reportes importados desde Microsoft Access.
- Incluye filtros para exportar a los formatos más populares, tales como Adobe PDF, Microsoft Excel, RTF (Rich Text Format (formato de texto enriquecido)), HTML, texto plano e imágenes TIFF (Tagged Image File Format (Formato de fichero para imágenes)), tanto para aplicaciones de escritorio como para las basadas en la web.
- Diseñador de reportes orientado a usuarios finales, permitiendo su inclusión en las aplicaciones con el fin de que los propios usuarios diseñen y modifiquen sus reportes. *(GrapeCity 2011)*

Sistema Operativo: Para utilizarlo se requiere tener instalado Windows NT 4.0, Windows 2000, Windows XP, Windows Vista o Windows 2003 Server.

Licencia: Se distribuye bajo licencia privativa.

1.1.6 Generador Dinámico de Reportes

El Generador Dinámico de Reportes es una herramienta de generación de reportes desarrollada en la Universidad de las Ciencias Informáticas, en el centro de desarrollo de software DATEC, específicamente en el departamento de Integración de Soluciones, dicha herramienta es una aplicación multiplataforma, con tecnologías web, que permite la creación y edición de reportes utilizando una amplia gama de fuentes de datos. Los reportes son creados teniendo en cuenta el siguiente estándar: Encabezado del documento, Encabezado de página, Cuerpo, Pie del documento y Pie de página. El GDR da la posibilidad de cargar en una vista estándar los modelos de bases de datos con los que se desea trabajar a través de una interfaz amigable, creada con el objetivo de

brindar la mayor cantidad de opciones posibles para el diseño de los reportes, los que pueden ser personalizados dependiendo de las necesidades del informe deseado, dando la posibilidad de agregar gráficas e imágenes que muestren una representación más clara de los datos a valorar.

De los cinco sistemas de generación de reportes estudiados solo Crystal Reports, iReport y Pentaho Reporting cuentan con Asistente de Reportes. A continuación se analiza el proceso de diseño predeterminado de reportes que realizan dichos Asistentes con el objetivo de determinar los tipos de reportes a incluir en el GDR V2.0.

1.2 Asistentes de diseño predeterminado de reportes

Los Asistentes de diseño predeterminado de reportes son herramientas complementarias de los sistemas generadores de reportes que les permiten a los usuarios agilizar enormemente el proceso de creación de reportes, facilitando la creación de informes de manera rápida y eficiente. Tanto los usuarios sin experiencia, como los programadores, prefieren crear la mayoría de sus informes usando estos modelos. Generalmente, el trabajo con estos consiste en elegir el reporte predeterminado que más se aproxime al tipo de informe que se desea crear y de esta forma el Asistente guía a los usuarios a través de pasos durante todo el proceso de creación de informes.

1.2.1 Asistente de diseño predeterminado de Crystal Reports

Crystal Reports permite generar con el Asistente los siguientes reportes: Estándar, Tablas cruzadas, Etiqueta y Olap. (Business Objects SA 2005)

El Asistente para la creación de informes estándar es el Asistente más genérico, guía al usuario en la elección de una fuente de datos y en el establecimiento de vínculos entre las tablas de una base de

datos, también permite añadir campos y especificar los criterios de agrupamiento, resumen (totales) y ordenamiento que se desee utilizar, además de orientar el proceso de creación de gráficos y selección de registros.

El Asistente para la creación de informes de tablas cruzadas guía al usuario en la creación de un informe en el que los datos se muestran como un objeto de tabla cruzada. Existen dos pantallas especiales (Tablas cruzadas y Tipo de cuadrícula) que crean y dan formato a la tabla cruzada.

El Asistente para la creación de informes de etiquetas de correo puede crear informes en un formato tal que permita su impresión en etiquetas de cualquier tamaño. Puede usar la pantalla Etiqueta para seleccionar un tipo de etiqueta comercial o puede definir su propia presentación de filas y columnas para cualquier informe con diseño de varias columnas.

El Asistente para la creación de informes de Crystal Reports consta de varias pantallas que guían al usuario mediante instrucciones, paso a paso, en la creación del informe especificado.

Las pantallas comunes son:

- Pantalla Datos
- Pantalla Vínculo
- Pantalla Campos
- Pantalla Agrupamiento
- Pantalla Resúmenes
- Pantalla Ordenación de grupos
- Pantalla Estilo de informe

La Pantalla Datos se utiliza para seleccionar el origen de datos y tablas que se deseen para el informe.

La Pantalla Vínculo se usa para vincular las tablas del nuevo informe. La pantalla aparece en el

Asistente para la creación de informes cuando se han seleccionado dos o más tablas en la Pantalla Datos.

La Pantalla Campos se usa para seleccionar los campos que se deseen incluir en el informe.

La Pantalla Agrupamiento se utiliza para especificar cómo se agruparán los campos en el informe. La creación de grupos es un paso opcional en el Asistente.

La Pantalla Resúmenes se usa para elegir campos, de los que se va a calcular el subtotal, contar, etc. Esta pantalla aparece únicamente si se ha especificado un grupo en la Pantalla Agrupamiento. La creación de un campo de resumen es un paso opcional en el Asistente.

La Pantalla Ordenación de grupos se utiliza para ordenar los grupos creados en la Pantalla Agrupamiento. Se pueden ordenar todos los grupos o se pueden elegir los cinco grupos superiores o inferiores. Cuando se ordena por los cinco grupos superiores o inferiores, también se puede elegir el campo resumido en el que se base el ordenamiento. Esta pantalla aparece únicamente si se ha especificado un grupo en la Pantalla Agrupamiento y un resumen en la Pantalla Resúmenes. La creación de un ordenamiento de grupos es un paso opcional en el Asistente.

La Pantalla Estilo de informe se usa para elegir entre las plantillas de formato predefinido y usarlas en el informe. La adición de un estilo es un paso opcional en el Asistente.

Los Asistentes para la creación de informes de tablas cruzadas y de etiquetas de correo tienen pantallas exclusivas para sus tipos de informe específicos, las cuales son:

Pantallas exclusivas para la creación de informes de tablas cruzadas:

- Pantalla Tablas cruzadas
- Pantalla Diagrama
- Pantalla Selección de registro

- Pantalla Tipo de cuadrícula

La Pantalla Tablas cruzadas se utiliza para crear y dar formato a un objeto de tabla cruzada para el informe. Esta pantalla aparece únicamente en el Asistente para la creación de informes de tablas cruzadas.

La Pantalla Diagrama inserta un gráfico o un diagrama en el informe, y aparece con ligeras variaciones en los Asistentes, exceptuando el Asistente para la creación de informes de etiquetas de correo. En el Asistente para la creación de informes estándar, la pantalla Diagrama aparece solo si se ha especificado un grupo en la Pantalla Agrupamiento y un resumen en la Pantalla Resúmenes. La creación de gráficos es un paso opcional en todos los Asistentes.

La Pantalla Selección de registro se usa para elegir campos, seleccionar (o filtrar) registros en un informe. Al seleccionar los registros de esta forma, se puede limitar el ámbito del informe y aumentar la velocidad de procesamiento. La creación de una selección de registros es un paso opcional en el Asistente.

Pantalla Tipo de cuadrícula se usa para elegir un estilo predefinido para la tabla cruzada.

Pantallas exclusivas para la creación de informes de etiquetas de correo:

- Pantalla Etiqueta

La Pantalla Etiqueta se usa para configurar las opciones de las etiquetas de correo. Permite imprimir datos en la práctica, totalidad de etiquetas comercialmente disponibles para impresoras de líneas o impresoras de páginas. *(Business Objects documentation 2005)*

1.2.2 Asistente de diseño predeterminado de iReport

iReport contiene un Asistente de diseño predeterminado de reportes para crear rápidamente informes de bases de datos de forma automática, el mismo se encuentra en el menú Fichero Mago de Informes

(Report Wizard). Permite generar los siguientes reportes: Estándar y Tablas cruzadas. *(Business Objects SA 2005)*

La secuencia de pasos a seguir para la elaboración de los reportes es la siguiente: *(MCE Technical Publications 2008)*

1. Select the datasource type: Se selecciona la conexión de origen de datos o base de datos a utilizar para el informe.
2. Name and locaction: Se introduce el nombre y la ubicación adecuada para el informe.
3. Query: Se entra la consulta para obtener los campos que se utilizarán para crear el informe.
4. Fields: Se seleccionan los campos que se visualizarán en el informe.
5. Group by: Se selecciona la forma en que se desea agrupar los datos.
6. Layout: Se selecciona la plantilla que define el estilo visual del informe (en columnas o tabular).
7. Finish: Se da por concluido el diseño y se procede a generar la plantilla del informe para su visualización en el editor.

1.2.3 Asistente de diseño predeterminado de Pentaho Report Designer

Pentaho Report Designer incorpora un Asistente de reportes llamado Report Design Wizard, herramienta de diseño de informes muy útil, incluido en la última versión (1.6.0) destinada a usuarios con menos conocimientos técnicos, facilita y agiliza enormemente la creación de reportes a los usuarios, el mismo se encuentra en el menú Archivo, opción Asistente de reporte. Permite generar los siguientes reportes: Estándar, Tablas cruzadas y Olap proporcionando una guía de siete pasos para el diseño. *(Adobe PDF Library 8.0 2008)*

La secuencia de pasos a seguir para la elaboración de los reportes es la siguiente:

1. Comienzo: El primer paso del Asistente de reportes permite definir el título y una breve descripción, además de seleccionar plantillas predeterminadas para la visualización general del reporte final.
2. Consulta: Seleccionar el tipo de conjunto de datos que se desea utilizar en el informe. Para esto se debe agregar o establecer una conexión a la lista de conexiones que se encuentra del lado izquierdo. Posteriormente, en el área de Query Details, se encuentra un cuadro de texto donde se especifica la sentencia de la consulta a realizar. Para el caso de conexiones JNDI (Java Naming and Directory Interface (Interfaz de Programación de Aplicaciones que puede hacer uso de un servidor, un fichero, o una base de datos)) (excepto con el uso de MDX (Acrónimo de MultiDimensional eXpressions (Lenguaje de consulta para bases de datos multidimensionales))) y MQL (Lenguaje de programación) estará disponible el Query Designer, el cual ayuda a realizar las consultas con herramientas gráficas y modelado de la BD (Base de Datos).
3. Mapeo de plantilla: Este paso es relevante si se está utilizando una plantilla de estilo seleccionada en el primer paso, se le indica al Asistente algunos valores para etiquetas y elementos que puedan estar incluidos en la plantilla.
4. Distribución: En el siguiente paso se definen los niveles de agrupación de los datos, es recomendable organizar los reportes utilizando grupos que contengan algún parámetro.
5. Formato: Se define y personaliza la apariencia del informe mediante plantillas. Para cada campo de la consulta que estemos mostrando se puede configurar un conjunto de propiedades asociadas a este.
6. Página: El Asistente permitirá configurar algunos aspectos de más alto nivel, tales como el formato de la página y la marca de agua.

7. Opciones Avanzadas: En este último paso que nos ofrece el Asistente de reportes de Pentaho Report Designer se configura un conjunto de opciones para la personalización del reporte *(Emilio Arias 2007)*

Según el análisis realizado a los Asistentes y la comparación establecida entre estos en cuanto a los tipos de reportes predeterminados que permiten diseñar, se determinó incluir los reportes comunes siguientes: Reportes Columnares (Estándares) y Reportes de Tablas Cruzadas. Teniendo en cuenta otros tipos de reportes, que estos Asistentes no permiten generar, pero que podrían resultar de gran importancia incluir en el sistema GDR V2.0, se incluirá también los Reportes de Tablas Pivote. Que, además, fueron solicitados por el cliente.

1.3 Ambiente de desarrollo

Por investigaciones realizadas anteriormente a nivel de proyecto ya se encontraban definidas las tecnologías que formarán parte del ambiente de desarrollo a utilizar para dar solución al problema en cuanto a Metodología, Herramientas, Lenguajes y Framework.

1.3.1 Metodología de desarrollo OpenUP

Roger S. Pressman en “Ingeniería del Software. Un Enfoque práctico” caracteriza el proceso de desarrollo de software como un marco de trabajo de las tareas que se requieren para construir software de alta calidad. *(Roger S. Pressman 2005)* En un proyecto de desarrollo de software, la metodología es el proceso que define Quién debe hacer, Qué, Cuándo y Cómo debe hacerlo. Guía a los desarrolladores en la realización de las actividades, durante todo el proceso de creación de un producto. No existe una metodología de software universal, las características de cada proyecto o

equipo de desarrollo exigen que el proceso sea configurable. *(latecladeescape.com 2010)*

La metodología que se utilizará para desarrollar el Asistente de diseño predeterminado de reportes es OpenUP por decisión del proyecto, la cual se define como un proceso de desarrollo de software unificado, dentro de un ciclo de vida estructurado, basado en RUP, que mantiene las características de estar dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental. Puede dirigir cualquier tipo de proyecto, el tiempo de desarrollo es corto, el equipo de trabajo es pequeño, y cuenta con menos de veinte artefactos. Es un método y un proceso de desarrollo de software propuesto por un conjunto de empresas de tecnología, que lo donaron en el año 2007 a la Fundación Eclipse, publicado bajo una licencia libre, actualmente se mantiene como método de ejemplo dentro del proyecto Eclipse Process Framework, desarrollado por IBM (International Business Machines (Empresa multinacional estadounidense que fabrica y comercializa herramientas, programas y servicios relacionados con la informática) y reconocido mundialmente como uno de los procesos de desarrollo de software de mayor calidad basándose en los principios de Adaptación, Importancia a los involucrados e interesados en los resultados del proyecto; Colaboración, Valor a la iteración; y Calidad Continua. *(Ricardo Balduino 2008*)

Características fundamentales de OpenUP

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

Capas de OpenUP

- Microincrementos
- Ciclo de vida de la iteración
- Ciclo de vida del proyecto

OpenUP estructura el ciclo de vida del proyecto en cuatro fases: Concepción, Elaboración, Construcción y Transición. El ciclo de vida del proyecto provee a los interesados de un mecanismo de supervisión y dirección para controlar los fundamentos del proyecto, su ámbito, la exposición a los riesgos, el aumento de valor y otros aspectos.

1.3.2 Herramienta de modelado Visual Paradigm 6.1

La herramienta de modelado Visual Paradigm soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML (Unified Modeling Language (Lenguaje Unificado de Modelado de sistemas de software)) ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación, es fácil de usar, soporta la última notación UML 2.1, ingeniería inversa, generación de código, importación desde Rational Rose (Herramienta CASE (Computer Aided Software Engineering (Ingeniería de Software Asistida por Ordenador, diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software) que comercializan los desarrolladores de UML), exportación/importación XMI (Nombre estándar para el intercambio de metamodelos usando XML con el objetivo de permitir un intercambio de metainformación), generador de informes, editor de figuras, integración con MS Visio, plug-in, integración IDE con Visual Studio, Eclipse, NetBeans. Entre sus nuevas características se incluyen el modelado colaborativo con CVS y

Subversion, interoperabilidad con modelos UML2 a través de XMI. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. *(Evelyn Menéndez Alonso 2009)*

1.3.3 Lenguaje de modelado UML 2.0

El lenguaje de modelado UML es el lenguaje estándar de propósito general más utilizado para especificar y documentar cualquier sistema de forma precisa. Proporciona una gran flexibilidad y expresividad a la hora de modelar sistemas. UML es una consolidación de muchas de las notaciones y conceptos más usados orientados a objetos. Empezó como una consolidación del trabajo de Grade Booch, James Rumbaugh, e Ivar Jacobson, creadores de tres de las metodologías orientadas a objetos más populares. UML se puede usar para modelar distintos tipos de sistemas: de software, hardware y organizaciones del mundo real. *(James Rumbaugh e Ivar Jacobson 2010)*

1.3.4 Lenguaje de programación para la capa de presentación JavaScript

Al igual que VisualBasic y Perl, JavaScript es un lenguaje interpretado, característica que lo hace especialmente idóneo para trabajar en la Web, son los navegadores que se utilizan para viajar por ella los que interpretan y, por tanto, ejecutan los programas escritos en JavaScript. De esta forma, se envían documentos a través de la Web que llevan incorporados el código fuente de programas, convirtiéndose de esta forma en documentos dinámicos, dejando de ser simples fuentes de información estática. JavaScript comparte muchos elementos con otros lenguajes de alto nivel. *(Juan J. Merelo 2009)*

Permite incluir macros en páginas Web, estas macros se ejecutan en el ordenador del visitante de las páginas, y no en el servidor (algo muy interesante, porque los servidores Web suelen estar

sobrecargados, mientras que las PC's de los usuarios no suelen estarlo). Hay que tener en cuenta que este lenguaje es muy semejante a otros como C, Java o PHP (Lenguaje de programación interpretado, diseñado originalmente para la creación de páginas web dinámicas) tanto en su formato como en su sintaxis, aunque por supuesto tiene sus propias características definitorias. *(Marcos Legido Hernández 2009)*

Lenguaje basado en objetos: es decir, el paradigma de programación es básicamente el de la programación dirigida a objetos, pero con menos restricciones).

Lenguaje orientado a eventos: debido al tipo de entornos en los que se utiliza (Windows y sistemas X-Windows). Gran parte de la programación se centra en describir objetos (con sus variables de instancia y métodos de "clase") y escribir funciones que respondan a movimientos del ratón, pulsación de teclas, apertura y cerrado de ventanas o carga de una página, entre otros eventos.

JavaScript proporciona los medios para:

- Controlar las ventanas del navegador y el contenido que muestran.
- Evitar depender del servidor Web para cálculos sencillos.
- Capturar los eventos generados por el usuario y responder a ellos sin salir a Internet.
- Simular el comportamiento de las macros CGI (Common Gateway Interface (Interfaz de entrada común, importante tecnología de la World Wide Web (Sistema de distribución de información basado en hipertexto o hipermedios enlazados y accesibles a través de Internet) que permite a un cliente (navegador web) solicitar datos de un programa ejecutado en un servidor web)) cuando no es posible usarlas.
- Comprobar los datos que el usuario introduce en un formulario antes de enviarlos.
- Comunicarse con el usuario mediante diversos métodos.

1.3.5 Lenguaje de programación para representar la estructura del reporte XML

El lenguaje de marcas XML representa el conjunto de reglas para definir etiquetas semánticas que organizan un documento en diferentes partes. XML es un metalenguaje que define la sintaxis utilizada para definir otros lenguajes de etiquetas estructurados. En teoría HTML es un subconjunto de XML especializado en presentación de documentos para la Web, mientras que XML es un subconjunto de SGML especializado en la gestión de información para la Web. En la práctica, XML contiene a HTML, aunque no en su totalidad. XML fue desarrollado por un grupo de trabajo bajo los auspicios de la World Wide Web (W3C (Consorcio internacional que produce recomendaciones para la World Wide Web, organismo que vela por el desarrollo de esta partiendo de las amplias especificaciones de SGML (Standard Generalized Markup Language (Estándar de Lenguaje de Marcado Generalizado)). *(María Isabel García Arenas 2009)* Este fue constituido en 1994 con el objetivo de desarrollar protocolos comunes para la evolución de Internet.

Características que ofrece XML: *(JoséManuel López Franco 2001)*

- Aunque hoy día XML aún no está tan extendido como HTML, su uso futuro en la web mejorará la eficiencia de las búsquedas, al proporcionar cada documento XML metadatos sobre sí mismo.
- Permite proporcionar diferentes vistas sobre los datos (HTML, PDF, voz, entre otros), dependiendo de quién sea el cliente.
- Facilita la integración desde fuentes de datos heterogéneas, por ejemplo, páginas Web, distintas bases de datos.
- Los documentos tienen una estructura que los hace legibles e inteligibles, no solo para los ordenadores, sino, también, para los humanos.

- Las aplicaciones de XML son fácilmente extensibles mediante definiciones de nuevos tipos de documentos DTD (Document Type Definition (Descripción de una estructura y sintaxis de un documento XML o SGML)).
- XML proporciona una representación estructural de los datos que ha probado ser ampliamente implementable y fácil de distribuir.
- Dentro de XML se puede definir un conjunto ilimitado de etiquetas.
- El poder y la belleza del XML es que mantiene la separación entre la interface de usuario y los datos estructurados.
- Los datos codificados en XML pueden ser transmitidos sobre la Web hasta el escritorio.

1.3.6 Lenguaje de programación para la transferencia de información JSON

El formato ligero para el intercambio de datos JSON (JavaScript Object Notation), fácil de leer y escribir, pero, también, fácil de analizar, usa lenguaje C (incluyendo C, C + +, C #, Java, JavaScript, Perl, Python). Estas características hacen del formato JSON el intercambio de datos ideal. Nada más puede tener una estructura jerárquica, lo único que tiene son objetos, vectores, variables y valores. La simplicidad de JSON ha dado lugar a la generalización de su uso, especialmente como alternativa a XML en AJAX (Asynchronous JavaScript And XML (Técnica de desarrollo web para crear aplicaciones interactivas o RIA (Rich Internet Applications))). Una de las supuestas ventajas de JSON sobre XML como formato de intercambio de datos en este contexto es que es mucho más sencillo escribir un analizador semántico de JSON *(Jimmy Wales 2010a)*

Características esenciales:

- JSON es un protocolo de intercambio de datos ligero, fácil de leer para humanos y máquinas.

- Existen parsers de JSON para la mayoría de lenguajes de programación y para todos los más populares.
- Los datos en JSON ocupan mucho menos que XML.
- Su procesamiento por parte de los ordenadores es rápido; se necesitan librerías muy pequeñas para trabajar con él (siendo posible, incluso, procesarlo sin librería); dada su naturaleza es ideal para entornos Ajax, y ya existen parsers (Analizador sintáctico) para este formato en varios lenguajes de programación.

1.3.7 Framework Ext JS 3.3

El marco de trabajo Ext JS fue creado inicialmente por Jack Slocum. Empezó siendo un conjunto de librerías y extensiones para YUI (Yahoo! User Inteface), librería desarrollada en JavaScript que explota las potencialidades de AJAX para el desarrollo de RIA. Escrito en JavaScript con la finalidad de asistir el desarrollo de aplicaciones enriquecidas para Internet. Con el tiempo se convirtió en un Framework independiente y a principios de 2007 se creó una compañía para comercializar y dar soporte del Framework Ext. De esta forma Ext tiene dos tipos de licencias, GPL y comercial, basado en componentes soportados por recursos para la programación orientada a objetos en JavaScript, los que facilitan la implementación de extensiones y aplicaciones de gran complejidad. Ext JS es uno de los Frameworks que, además de flexibilizar el manejo de componentes de la página como el DOM (Document Object Model (Modelo de objetos para la representación de documentos, esencialmente una API que proporciona un conjunto estándar de objetos para representar documentos HTML y XML) y peticiones AJAX, tiene la gran funcionalidad de crear interfaces de usuario bastante funcionales. *(Shea Frederick, Colin Ramsay, y Steve 'Cutter' Blades 2008)*

1.3.8 Herramienta de desarrollo NetBeans 7.0

La herramienta de desarrollo NetBeans provee de una estructura para los proyectos que se puede crear junto a este IDE, a partir de un conjunto de componentes de software llamado módulo. Un módulo es un archivo Java que contiene clases de Java escritas para interactuar con las APIs de NetBeans. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que estos pueden ser desarrollados de forma independiente, las aplicaciones basadas en la plataforma NetBeans pueden ser extendidas fácilmente por otros desarrolladores de software. Es un proyecto de código abierto de gran éxito con una gran base de usuarios, una comunidad en constante crecimiento. Sun MicroSystems (Empresa informática) fundó el proyecto de código abierto NetBeans en junio de 2000 y continúa siendo el patrocinador principal de los proyectos. *(Jimmy Wales 2010b)*

Características generales: *(Ezequiel Montes 2010)*

Integración con Symfony y ZenFramework: Una de las ventajas de utilizar NetBeans es justamente su integración con estos populares Framework de PHP, lo que hace posible dejar a un lado la consola de comandos de Symfony y centrarse en desarrollar en el IDE, además, se encuentran cargadas todas las clases, ayuda en línea.

Editor de Código Fuente: El editor de PHP, es mucho más ágil y a la vez robusto, contiene más ayuda en línea, reconocimiento de sintaxis y todo lo que provee la última versión de PHP, la 5.3.

Integración con PHP Unit Testing: Es posible crear test con PHPUnit para diferentes funciones, luego realizar la comprobación y ver todos los resultados. En las propiedades PHPUnit puede definir una configuración personalizada de archivos XML, un archivo de arranque para las opciones de línea de comandos, o una serie de pruebas a medida, o puede que el IDE genere el código.

Depuración de PHP: NetBeans integra muy bien la utilización Xdebug, gracias a esto se puede

inspeccionar y examinar cada variable local, establecer puntos de interrupción y evaluar el código en nuestra lógica. El IDE de NetBeans para PHP también ofrece la línea de comandos de depuración: La salida del programa PHP aparece en una pantalla de línea de comandos en el IDE de sí mismo y se puede inspeccionar el código HTML generado sin tener que cambiar a un navegador.

Integración con MySQL: MySQL y NetBeans poseen una integración completa en términos de administración básica y avanzada de MySQL, y todo desde el mismo entorno.

Integración con Sistemas de Control de Versiones: Esta es una de las condiciones necesarias para los proyectos y es la posibilidad de contar con la integración de sistemas de control de versiones, tales como SVN, CVS, Mercurial y Git. Desde el editor es posible realizar la administración de estos sistemas versionados, sus commit, branch, importar, exportar, revert, clonar, entre otros.

1.4 Roles involucrados

Las personas involucradas en un proyecto de software pueden ser los clientes, usuarios finales, administrativos y desarrolladores. El término rol es el papel o los papeles que puede desempeñar determinado trabajador. RUP define determinados roles por flujo de trabajo dependiendo de los objetivos, actividades y artefactos que se construyan en el mismo. A continuación se especifican los roles que se desempeñarán durante el proceso de desarrollo de software.

Analista del sistema: Dirige y coordina la adquisición de requisitos esquematizando la funcionalidad del sistema y delimitándolo. Especifica y detalla los requisitos del sistema.

Diseñador: Dirige el diseño de una parte del sistema, dentro de las restricciones de los requisitos, arquitectura y proceso de desarrollo para el proyecto.

Diseñador de interfaz de usuario: Coordina el diseño de la interfaz de usuario. Esto incluye recopilar los requisitos de utilización y los diseños de interfaz de usuario candidata a la creación de prototipos

para cumplir estos requisitos.

Implementador: Desarrolla los componentes de software y efectúa las pruebas de desarrollador para la integración en subsistemas más grandes, de acuerdo con los estándares adoptados de proyecto.

Probador: Responsable de las actividades básicas de ejecución de pruebas, que implica el conducir las pruebas necesarias y el registrar los resultados de aquello que prueba.

1.5 Conclusiones Parciales

Después de analizar los diferentes tipos de reportes predeterminados que proveen los principales sistemas de generación de reportes se concluye lo siguiente:

- Se determinaron los tipos de reportes a incluir en el sistema Generador Dinámico de Reportes para la versión 2.0.
- Se conformó el ambiente de desarrollo en cuanto a las tecnologías a utilizar.
- Se definieron los roles a desempeñar para desarrollar el Asistente de Reportes.

CAPÍTULO 2: ANÁLISIS DE LA SOLUCIÓN

Introducción

En este capítulo se realiza el análisis del sistema a desarrollar. Profundizando en las actividades específicas del proceso, que permiten asegurar una organización del software para alcanzar resultados satisfactorios. El capítulo incluye: Modelo de dominio, Definición de requisitos (funcionales y no funcionales), Modelo del Sistema que contiene Actores y Casos de uso representados en el Diagrama de casos de uso del sistema, y las Descripciones textuales correspondientes a los Casos de uso.

2.1 Propuesta del sistema

Se propone la implementación de un Asistente de reportes para el módulo Diseñador de Reportes, que sea capaz de generar los tres tipos de reportes seleccionados, y permita guiar a cualquier tipo de usuario durante todo el proceso de diseño de reporte, facilitando un ahorro significativo de tiempo y esfuerzo mediante operaciones sencillas y básicas.

2.1.1 Modelo de Dominio

El Modelo de Dominio es un artefacto de la disciplina de análisis, construido con las reglas de UML durante la fase de concepción, en la tarea construcción del modelo de dominio, presentado como uno o más diagramas de clases que contiene, no conceptos propios de un sistema de software, sino de la propia realidad física. *(Iván Garcerant 2008)* El modelo de dominio es una representación visual estática del entorno real objeto del proyecto. Es un diagrama con los objetos reales que existen, relacionados con el sistema que se va a desarrollar y las relaciones que existe entre ellos.

El modelo de dominio del sistema Asistente de Reportes es el siguiente:

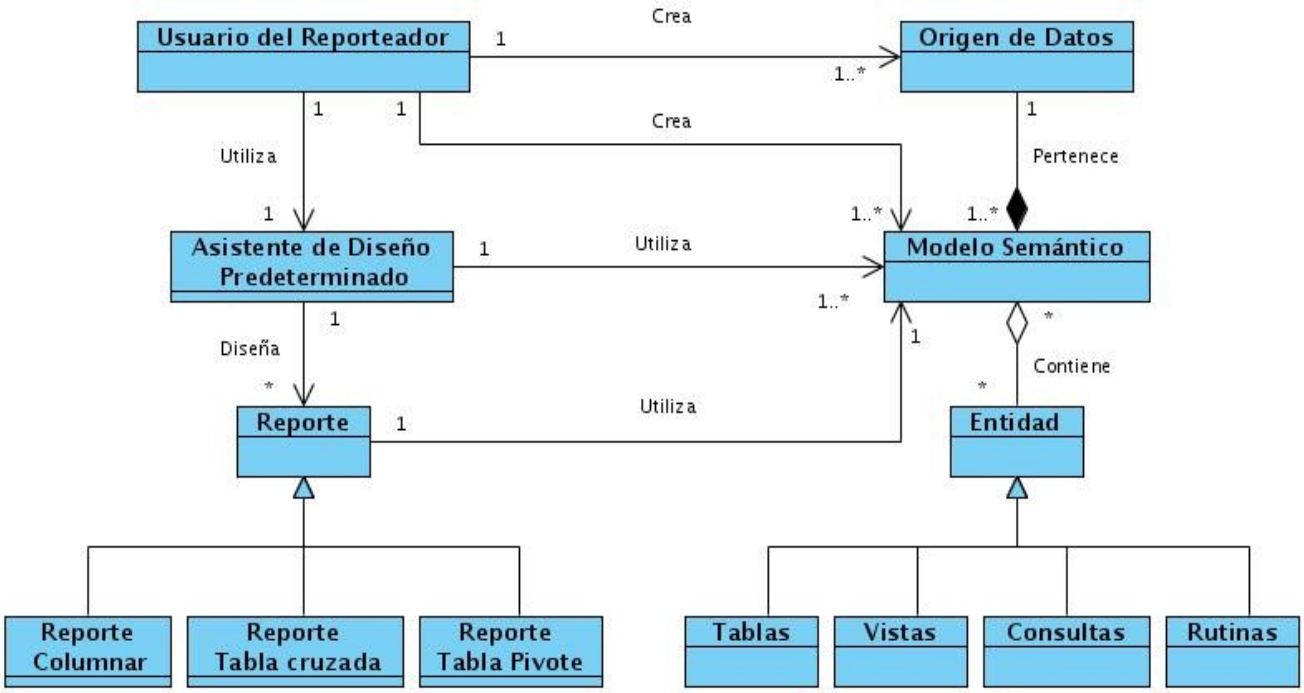

Figura 2: Modelo de dominio

Descripción de las clases:

Usuario del Reporteador: Persona encargada de diseñar los reportes.

Asistente de Diseño Predeterminado: Mecanismo de diseño de reportes, con una baja participación del Usuario del Reporteador, que permite generar reportes a través de una secuencia de pasos.

Reporte: Objeto que entrega información en un formato particular y que permite realizar ciertas operaciones como imprimirlo, enviarlo por email, guardarlo, entre otras, a partir de los datos almacenados en una base de datos. El reporte puede ser Columnar, de Tabla Cruzada, o Tabla Pivote.

Reporte Columnar: Es un tipo de reporte en forma de listado de columnas.

Reporte de Tabla Cruzada: Es un tipo de reporte matricial en forma de tabla donde se proporciona información resumida en un formato de estilo de hoja de cálculo, que genera los datos de resumen en una cuadrícula donde las filas y las columnas representan grupos de datos.

Reporte de Tabla Pivote: Es un tipo de reporte matricial en forma de tabla donde se proporciona información resumida al igual que en los reportes de Tabla Cruzada, cuenta con un elemento pivote al cual se le aplican operaciones y de esta forma obtener la información necesaria.

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

Modelo Semántico: Es un modelo de datos usado para almacenar en forma de fichero XML toda la información de los metadatos de los objetos de la base de datos, formado por entidades tales como: tablas, vistas, consultas y rutinas.

Entidad: Objeto de la base de datos que puede ser una Tabla, Vista, Consulta o Rutina.

Tablas: Objeto de la base de datos donde se almacena la información en forma de tabla.

Vistas: Objeto de la base de datos que almacena el resultado de una consulta realizada a la base de datos.

Consultas: Objeto de la base de datos donde se definen las preguntas que se realizan a la base de datos con el fin de extraer y presentar la información resultante de diferentes formas.

Rutinas: Objeto de la base de datos que posee acceso directo a los datos que necesita manipular.

Reglas del negocio

"Las reglas del negocio describen políticas que deben cumplirse o condiciones que deben satisfacerse, por lo que regulan algún aspecto del negocio". Son las restricciones explícitas de comportamiento y/o proporcionan soporte para la dirección de las actividades del negocio, se aplican a lo largo de los procesos y procedimientos. *(Antonio C 2003)*

Algunas de las reglas del negocio son:

- Debe existir, creado en el sistema, al menos un "Origen de datos".
- Debe existir, creado en el sistema, al menos un "Modelo".
- Debe ser especificado al menos un campo de la fuente de datos para la fila de la tabla cruzada.
- Debe ser especificado al menos un campo de la fuente de datos para la columna de la tabla cruzada.
- Debe ser especificado al menos un campo de la fuente de datos para el elemento pivote de la tabla pivote.
- Debe ser especificada una operación al campo pivote.

2.2 Requisitos funcionales y no funcionales

Los requisitos son: condiciones o capacidades que necesita un usuario para darle solución a un problema y lograr un objetivo, o que tienen que ser alcanzadas por un sistema o componente de un sistema para satisfacer un contrato, estándar, u otro documento impuesto formalmente. Los requisitos se pueden clasificar en: funcionales y no funcionales.

Los requisitos funcionales son: capacidades o condiciones que el sistema debe cumplir.

Los requisitos no funcionales son: propiedades o cualidades que el producto debe tener, las

características que hacen al producto atractivo, usable, rápido o confiable.

Requisitos funcionales que debe cumplir el Asistente de Reportes

RF 1 Diseñar tipo de reporte columnar con Asistente.

RF 2 Diseñar tipo de reporte de tabla cruzada con Asistente.

RF 3 Diseñar tipo de reporte de tabla pivote con Asistente.

RF 4 Seleccionar fuente de datos.

RF 5 Seleccionar campos.

RF 6 Configurar formato de celda.

RF 7 Configurar parámetros.

RF 8 Generar vista previa del reporte en html**.**

Especificación de Requisitos Funcionales

Los requisitos funcionales se encuentran registrados en el documento *Especificación_requisitos_software_Asistente_Reportes_v2.0*, el cual contiene una lista detallada y completa de dichos requisitos donde el nivel de detalle de estos se encuentra especificado de forma clara, concreta, completa y consistente. En los Anexos se encuentra la especificación de tres de estos requisitos (v*er Anexo 8*)

Requisitos No Funcionales que debe cumplir el sistema a desarrollar

El Asistente de Reportes se integra a la arquitectura base del Generador Dinámico de Reportes y cuenta con los mismos requisitos no funcionales del GDR, los cuales se encuentran detallados en el documento *Especificación_requisitos_software_GDR.* A continuación se mencionan algunos de los

requisitos no funcionales que aparecen en dicho documento: Usabilidad, Fiabilidad, Eficiencia, Restricciones de diseño e implementación, Interfaz y Estándares Aplicables, entre otros.

2.3 Modelo del sistema

Requisitos es un flujo de trabajo de OpenUP que se desarrolla en la fase de Inicio, a partir de la especificación de los requisitos y la modelación gráfica de estos, agrupados en casos de uso. El artefacto fundamental de este flujo de trabajo es el Modelo del sistema que contiene actores y casos de uso representados en el diagrama de casos de uso del sistema, además de las descripciones textuales de dichos casos de uso. Este artefacto es un modelo de las funciones deseadas para el sistema y su entorno, que brinda una visión de lo que se debe implementar.

Actor del sistema

Un actor es un rol que un usuario desempeña con respecto al sistema. Es importante destacar el uso de la palabra rol, pues con esto se especifica que un actor no necesariamente representa a una persona en particular, sino, más bien, la labor que realiza frente al sistema.

Actor del sistema Asistente de Reportes

Diseñador de reporte

Caso de uso del sistema

Un caso de uso del sistema es una operación o tarea específica que se realiza tras una orden de algún agente externo, sea desde una petición de un actor o desde la invocación de otro caso de uso. Es la descripción de la secuencia de interacciones que se producen entre un actor y el sistema, cuando el actor usa el sistema para llevar a cabo una tarea específica.

Casos de uso del sistema Asistente de Reportes

CU 1 Diseñar reporte.

CU 2 Diseñar tipo de reporte Columnar con Asistente.

CU 3 Diseñar tipo de reporte de Tabla Cruzada con Asistente.

CU 4 Diseñar tipo de reporte de Tabla Pivote con Asistente.

CU 5 Seleccionar fuente de datos.

CU 6 Seleccionar campos.

CU 7 Configurar formato de celda.

CU 8 Configurar parámetros.

CU 9 Generar vista previa de reporte en html.

Diagrama de casos de uso del sistema

Un diagrama de casos de uso muestra la relación entre los actores y los casos de uso del sistema. Representa la funcionalidad que ofrece el sistema en lo que se refiere a su interacción externa. Los elementos que pueden aparecer en los diagramas de casos de uso son: actores, casos de uso y relaciones entre ellos.

Patrones de Casos de Uso

Los patrones de casos de uso son comportamientos que deben existir en el sistema, describen el uso del mismo y cómo este interactúa con los usuarios. Estos patrones capturan mejores prácticas para modelar casos de uso. A continuación se presentan los patrones de Casos de Uso utilizados para describir como están estructurados y organizados los casos de uso del Asistente de Reportes.

Concordancia (Especialización): El caso de uso Diseñar Reporte es un caso de uso base del cual heredan los casos de uso Diseñar Reporte Columnar, Diseñar Reporte Tabla Cruzada y Diseñar Reporte Tabla Pivote, donde todas la acciones del caso de uso base serán especializadas por los casos de usos hijos, así como también otras acciones que serán adicionadas.

Concordancia (Reuso): Los casos de uso pertenecientes al paquete Tipos de reportes representan la subsecuencia común, que modela una secuencia de acciones que aparecerán en los casos de uso pertenecientes al paquete Comunes, los cuales modelan el uso del sistema que comparte la subsecuencia común de acciones.

Concordancia (Adición): Los casos de uso pertenecientes al paquete Tipos de reportes representan la subsecuencia común, que modela una secuencia de acciones, la cual extiende los casos de uso compartiendo la subsecuencia de acciones. El caso de uso Generar vista previa html modelan el flujo que será expandido con la subsecuencia.

El diagrama de casos de uso del sistema Asistente de Reportes es el siguiente:

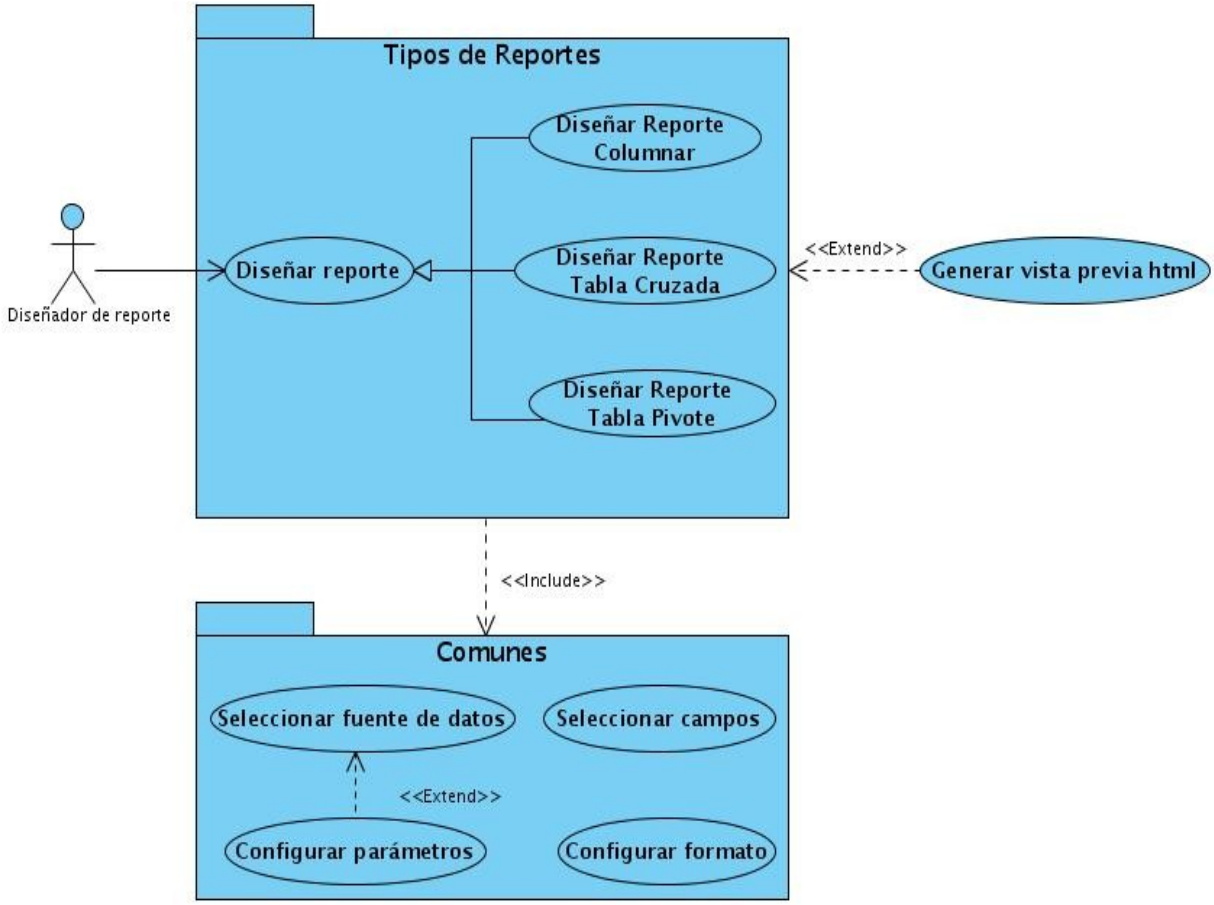

Figura 3: DCU del Asistente de Reportes

Descripciones textuales de los casos de uso del sistema

Las descripciones textuales se encuentran registradas en el documento Modelo_sistema_Asistente_Reportes_v2.0, el cual describe cada uno de los casos de uso explicando la forma en que interactúa el sistema con el usuario. A continuación se presenta un resumen de los principales Casos de uso del Asistente de Reportes.

Caso de uso Diseñar tipo de reporte columnar con Asistente

El presente caso de uso es la especialización del caso de uso genérico Diseñar Reporte, el cual permite el diseño de un reporte columnar. Para la realización de este debe ser especificado al menos un campo de la fuente de datos para la columna del reporte. El caso de uso se relaciona con los siguientes requisitos funcionales: Seleccionar fuente de datos, Configurar formato de celda y Generar vista previa del reporte en html.

Casos de uso Diseñar tipo de reporte de tabla cruzada con Asistente

El presente caso de uso es la especialización del caso de uso genérico Diseñar Reporte, el cual permite el diseño de un reporte de tabla cruzada. Para la realización de este debe ser especificado al menos un campo de la fuente de datos para la fila y uno para la columna de la tabla cruzada. El caso de uso se relaciona con los siguientes requisitos funcionales: Seleccionar fuente de datos, Seleccionar campos, Configurar formato de celda y Generar vista previa del reporte en html.

Caso de uso Diseñar tipo de reporte de tabla pivote con Asistente

El presente caso de uso es la especialización del caso de uso genérico Diseñar Reporte, el cual permite el diseño de un reporte de tabla pivote, para la realización de este debe ser especificado al menos un campo de la fuente de datos para la fila, uno para la columna y uno para el elemento pivote. El caso de uso se relaciona con los siguientes requisitos funcionales: Seleccionar fuente de datos, Seleccionar campos, Configurar formato de celda y Generar vista previa del reporte en html.

2.4 Conclusiones Parciales

Según las actividades realizadas en el capítulo para documentar el análisis del Asistente de diseño predeterminado de reportes se concluye lo siguiente:

- El modelo de dominio quedó conformado por cinco clases con sus respectivas descripciones.
- Se especificaron los ocho requisitos funcionales encontrados según el estudio realizado a los diferentes tipos de reportes predeterminados que proveen los principales sistemas de generación de reportes.
- El Asistente de Reportes quedó conformado por nueve casos de uso, representados y descritos textualmente en el Modelo del Sistema.
- La documentación del análisis del Asistente de Reportes se logró cumpliendo los objetivos necesarios con la realización de las tareas correspondientes para alcanzar el avance del proceso.

CAPÍTULO 3: DISEÑO DE LA SOLUCIÓN

Introducción

El diseño del software se encuentra en el núcleo técnico de la ingeniería del software y se aplica independientemente del modelo de diseño que se utilice. Una vez que se analizan y especifican los requisitos, el diseño es la primera de las tres actividades técnicas (diseño, generación de código y pruebas) que se requieren para construir y verificar el software.

En este capítulo se realiza el diseño del sistema a desarrollar, que incluye la descripción de los componentes que conforman el Asistente de Reportes y la descripción del XML de persistencia para los reportes de Tabla Pivote y Tabla Cruzada. Se presentan los Estilos arquitectónicos y Patrones de diseño utilizados. Se muestra el diagrama de clases y los diagramas de interacción de la Máquina de estados que dará soporte al Asistente de diseño predeterminado de reportes.

3.1 Descripción de los componentes del Asistente de Reportes

Un sistema está conformado por una estructura jerárquica de componentes, los cuales se pueden generalizar para representar los elementos principales del sistema y sus interacciones. *(Carlos Billy Reynoso 2004)*

El Asistente de Reportes está formado por los siguientes componentes:

- **Seleccionar fuente de datos:** permite seleccionar la fuente de datos que se desea utilizar para la elaboración del reporte, tendrá relación con el componente de seleccionar campos y configurar parámetros.
- **Configurar parámetros:** permite configurar los parámetros de una función en caso de que ésta

sea la fuente de datos seleccionada, tendrá relación con el componente Seleccionar fuente de datos.

- **Seleccionar campos:** permite seleccionar, de una fuente de datos, los campos que conformarán el reporte.
- **Configurar formato de celda:** permite configurar el formato que se desea que tenga las filas, columnas y datos del reporte.
- **Generar vista previa html:** permite visualizar en html el reporte diseñado.
- **Máquina de estados:** permite realizar la secuencia de pasos en el Asistente, tendrá relación con todos los componentes anteriormente descritos.

Los componentes Seleccionar fuente de datos, Configurar parámetros, Seleccionar Campos, Configurar formato de celda y Generar vista previa html son componentes que ya se encontraban implementados y son reutilizados por el Asistente de Reportes para la realización de las funcionalidades definidas que permiten el diseño de los tres tipos de reportes seleccionados anteriormente.

El componente Máquina de estados no se encontraba implementado y se implementará para lograr realizar la secuencia de pasos necesarios para construir el reporte.

3.2 Descripción del XML de persistencia del reporte mediante el Asistente

La salida principal del módulo Diseñador de Reportes es un fichero en formato XML que contiene las especificaciones del diseño del reporte y organiza la estructura del mismo para facilitar su procesamiento. A continuación se describen los elementos que conforman la estructura del XML utilizado para los reportes de Tabla Cruzada (ver Anexo 9) y Tabla Pivote (ver Anexo 10).

3.2.1 Descripción del XML para los reportes de Tabla Cruzada

REPORT: Representa la configuración general del reporte y está compuesto por los elementos HEADER, FOOTER, SOURCE REPORT, STYLE y ST.

- HEADER: Representa la configuración de la sección correspondiente a la cabecera del reporte.
- FOOTER: Representa la configuración de la sección correspondiente al pie del reporte.
- SOURCE_REPORT: Representa información de la fuente de datos utilizada para la creación del reporte y está compuesto por los elementos AVAILABLES y PARAMETERS.
  - AVAILABLES: Representa los campos disponibles de la fuente de datos y está compuesto por un conjunto de elementos SOURCE.
  - PARAMETERS: Representa información de los parámetros en caso de que la fuente de datos seleccionada sea una función y está compuesto por un conjunto de elementos SOURCE.
    - SOURCE: Representa información del campo especificado en la fuente de datos.
- STYLE: Representa la configuración del formato de la tabla y está compuesto por los elementos ROWS, COLUMNS y CUERPO.
  - ROWS: Representa la configuración del formato para el encabezado de las filas de la tabla.
  - COLUMNS: Representa la configuración del formato para el encabezado de las columnas de la tabla.
  - CUERPO: Representa la configuración del formato para los datos de la tabla.
- ST: Representa los campos que conforman la tabla y está compuesto por los elementos Rows, COLS y CELLS.

- Rows: Representa la configuración de datos de las filas y está compuesto por elementos FIELD.
- COLS: Representa la configuración de datos de las columnas y está compuesto por elementos FIELD.
- FIELD: Representa la configuración del campo de una relación de la base de datos.
- CELLS: Representa las operaciones que se aplican a los datos del reporte.

3.2.2 Descripción del XML para los reportes de Tabla Pivote

REPORT: Representa la configuración general del reporte y está compuesto por los elementos HEADER, FOOTER, SOURCE REPORT, STYLE y ST.

- HEADER: Representa la configuración de la sección correspondiente a la cabecera del reporte.
- FOOTER: Representa la configuración de la sección correspondiente al pie del reporte.
- SOURCE_REPORT: Representa información de la fuente de datos utilizada para la creación del reporte y está compuesto por los elementos AVAILABLES y PARAMETERS.
  - AVAILABLES: Representa los campos disponibles de la fuente de datos y está compuesto por un conjunto de elementos SOURCE.
  - PARAMETERS: Representa información de los parámetros en caso de que la fuente de datos seleccionada sea una función y está compuesto por un conjunto de elementos SOURCE.
    - SOURCE: Representa información del campo especificado en la fuente de datos.
- STYLE: Representa la configuración del formato de la tabla y está compuesto por los elementos ROWS, COLUMNS y CUERPO.

- ROWS: Representa la configuración del formato para el encabezado de las filas de la tabla.
- COLUMNS: Representa la configuración del formato para el encabezado de las columnas de la tabla.
- CUERPO: Representa la configuración del formato para los datos de la tabla.

- ST: Representa los campos que conforman la tabla y está compuesto por los elementos ROWS, COLS y PIVOTE.
  - ROWS: Representa los campos que conforman las filas del reporte y está compuesto por un conjunto de elementos FIELD.
  - COLS: Representa los campos que conforman las columnas del reporte y está compuesto por un conjunto de elementos FIELD.
  - PIVOTE: Representa la configuración de datos del pivote y está compuesto por un elemento FIELD.
  - FIELD: Representa la configuración del campo de una relación de la base de datos.

3.3 Estilo arquitectónico y Patrones de diseño utilizados

Un conjunto de estilos arquitectónicos permite que el ingeniero del software reutilice los conceptos a nivel de diseño aplicando patrones. Los estilos arquitectónicos definen las reglas generales de la organización en términos de patrones, las restricciones en la forma y la estructura de un grupo numeroso y variado de sistemas de software, además, determinan el vocabulario de componentes y conectores que pueden ser utilizados. *(Graig Larman 2004)*

3.3.1 Estilo arquitectónico utilizado

A nivel de proyecto se utiliza el estilo arquitectónico llamado “Estilo de llamada y retorno”, y dentro de él, el patrón de arquitectura Modelo-Vista Controlador (MVC), utilizado frecuentemente en aplicaciones Web. A continuación se describe cómo se aprecia esto en el Asistente de Reportes, el cual hereda este patrón.

El Modelo: Gestiona los datos solicitados por el controlador. Esto puede ser apreciado cuando se necesita mostrar la fuente de datos.

La Vista: Contiene los elementos asociados a la presentación de los datos y la información visual de los elementos que conformarán el Reporte, encontrándose el código asociado a las interfaces, que se encargan de visualizar la información que el controlador devuelve.

El Controlador: Se encarga de la trasferencia de información entre la vista y el modelo. Esto puede ser apreciado cuando se envía desde la vista la configuración del reporte al controlador, este lo captura y se lo manda al modelo para que lo almacene.

3.3.2 Patrones de diseño utilizados

Los patrones de diseño solucionan problemas que existen en varios niveles de abstracción. Con el uso de estos se evita la reiteración en la búsqueda de soluciones a problemas ya conocidos y solucionados anteriormente. Permiten formalizar un vocabulario común entre diseñadores y estandarizar el modo en que se realiza el diseño. *(Graig Larman 2004)* A nivel de proyecto se utilizan diversos patrones: GRASP, GOF (Gang Of Four) y otros particulares del marco de trabajo utilizado Ext JS.

A continuación se muestran los patrones GRASP utilizados en el Asistente de Reportes:

Alta Cohesión: En la solución las clases se caracterizan por no estar sobrecargadas y tener

responsabilidades estrechamente relacionadas y enfocadas. Las clases que conforman la Máquina de estados contienen varias funcionalidades, las que poseen un propósito único, no desempeñado por el resto de las clases, siendo estas funcionalidades las encargadas de controlar la navegación del Asistente de Reportes. Esto hace posible que las clases no se vean afectadas constantemente debido a los cambios y que resulten fáciles de comprender, reutilizar y conservar.

Bajo Acoplamiento: En el Asistente de Reportes las clases se encuentran estructuradas de forma tal que cuentan con la cantidad mínima de asociaciones. La dependencia que existe entre la Maquina de estados y las cartas es para la navegación entre estas.

A continuación se muestra el patrón GOF utilizado en el Asistente de Reportes:

Observer (Observador): El patrón Observador se encuentra implementado en la clase Ext.util.Observable del marco de trabajo Ext JS, clase que tiene asociado los métodos que facilitan el trabajo con eventos como el fireEvent método encargado de notificar que ha ocurrido un evento. En la solución se implementó un mecanismo para validar cada una de las cartas. Dicho mecanismo se encuentra en la clase CardBase siendo heredado por sus clases hijas, las cuales constituyen el elemento Observado, las cuales disparan el evento *valid* con su validación propia, el cual es registrado por la calse CardMapping (*ver Anexo 16*) que representa el elemento Observador, quedando de esta forma evidenciada la utilización del método fireEvent, el trabajo con eventos y poniéndose de manifiesto dicho patrón. A continuación aparece el fragmento de código que registra el evento que valida cada una de las cartas.

A continuación se muestran los patrones del marco de trabajo Ext JS utilizados en el Asistente de Reportes:

Saving resources with lazy component instantiation (Instanciación tardía de componente): Los

componentes visuales de Ext JS utilizados en la solución se construyen cuando se utiliza el Asistente de Reportes, debido a que no es necesaria la creación de estos hasta que se vayan a utilizar. De esta manera disminuye la cantidad de recursos en memoria. Lo anteriormente se evidencia en la clase CardMapping (*ver Anexo 16*) donde en vez de crear los botones se les pasa la configuración de los mismos, garantizando su creación cuando se inicializa el Asistente de Reportes. A continuación se muestra la imagen correspondiente a dicho ejemplo:

Building preconfigured classes (Construcción de clases preconfiguradas):

En el Asistente de Reportes se hace uso de las clases preconfiguradas, que son las extensiones de las clases de Ext JS con opciones de configuración implícitas. Ejemplo de esto se encuentra en un fragmento de la clase Card 2 _ PT (*ver Anexo 17*) donde al componente FieldSet que hereda de una clase de Ext JS se le pasa la configuración predeterminada con la cual va a ser posteriormente creado. A continuación se muestra la imagen correspondiente a dicho ejemplo:

3.4 Modelo del Diseño

El Modelo de Diseño se utiliza para documentar el diseño de un sistema. Es un modelo de objeto que describe la realización de los casos de uso, y sirve como una abstracción del Modelo de Implementación y del código fuente. Se utiliza como entrada esencial para las actividades en el flujo de trabajo Implementación y en el flujo de trabajo Prueba. Es un artefacto integral que abarca todas las clases del diseño y sus relaciones, e incluye los diagramas de clases y de interacción del diseño. *(Roger S. Pressman 2005)*

Diagramas de clases del diseño

Las clases definen los objetos, con los cuales se implementan los casos de uso. Los diagramas de

clases del diseño representan una abstracción de las clases de la implementación del sistema, dependiendo del lenguaje de programación.

Diagramas de Interacción

Un diagrama de interacción explica gráficamente las interacciones que existentes entre las instancias y las clases. El UML define dos tipos de estos diagramas; ambos sirven para expresar interacciones semejantes o idénticas de mensaje. Los diagramas de colaboración y Los diagramas de secuencia. Para documentar el diseño de un sistema se recomienda utilizar los diagramas de secuencia. En los Anexos se muestran los diagramas de secuencia pertenecientes a los CU para los reportes de Tabla Cruzada, Tabla Pivote (*ver Anexo 12*) y la integración del Asistente con el GDR (*ver Anexo 13*).

A continuación se muestran tres diagramas pertenecientes a las clases que conforman el Asistente de Reportes.

El primer diagrama representa las clases que corresponden al componente Máquina de estados. En el segundo diagrama está representado el caso de uso Diseñar tipo de reporte de Tabla Pivote con Asistente. El tercer y último muestra la integración de los diagramas anteriores para representar en uno solo todas las clases utilizadas en la implementación del Asistente de Reportes. En los Anexos se encuentra el diagrama de clases que representa el caso de uso Diseñar tipo de reporte de Tabla Cruzada con Asistente (*ver Anexo 11*).

Diagrama de clases de la Máquina de estados:

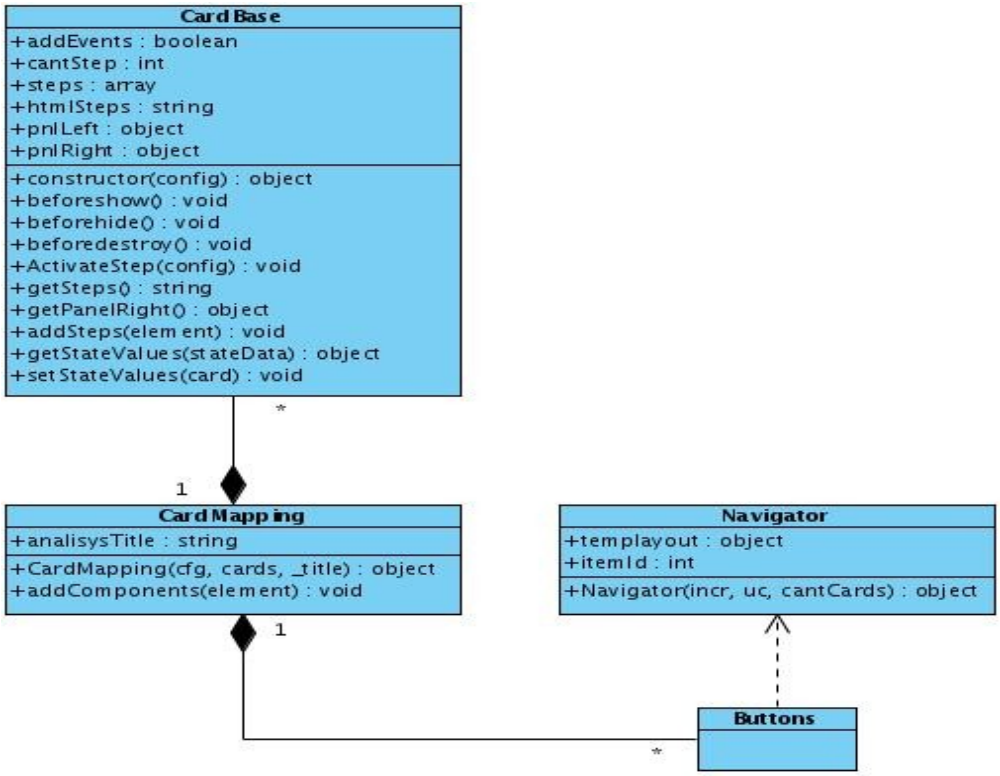

Figura 4: DCD de la Máquina de estados

Descripción de las clases:

- **CardBase**: Representa la definición básica de todas las cartas que se utilizan en el Asistente.
- **CardMapping**: Crea los botones necesarios para navegar en el Asistente de Reportes y registra el evento que valida cada una de las cartas.
- **Navigator**: Habilita y controla los estados de los botones y de las cartas.
- **Buttons:** Clase de Ext JS contenida en la clase CardMapping encargada de ejecutar las acciones de la clase Navigator.

Diagrama de clases del caso de uso Diseñar tipo de reporte de Tabla Pivote:

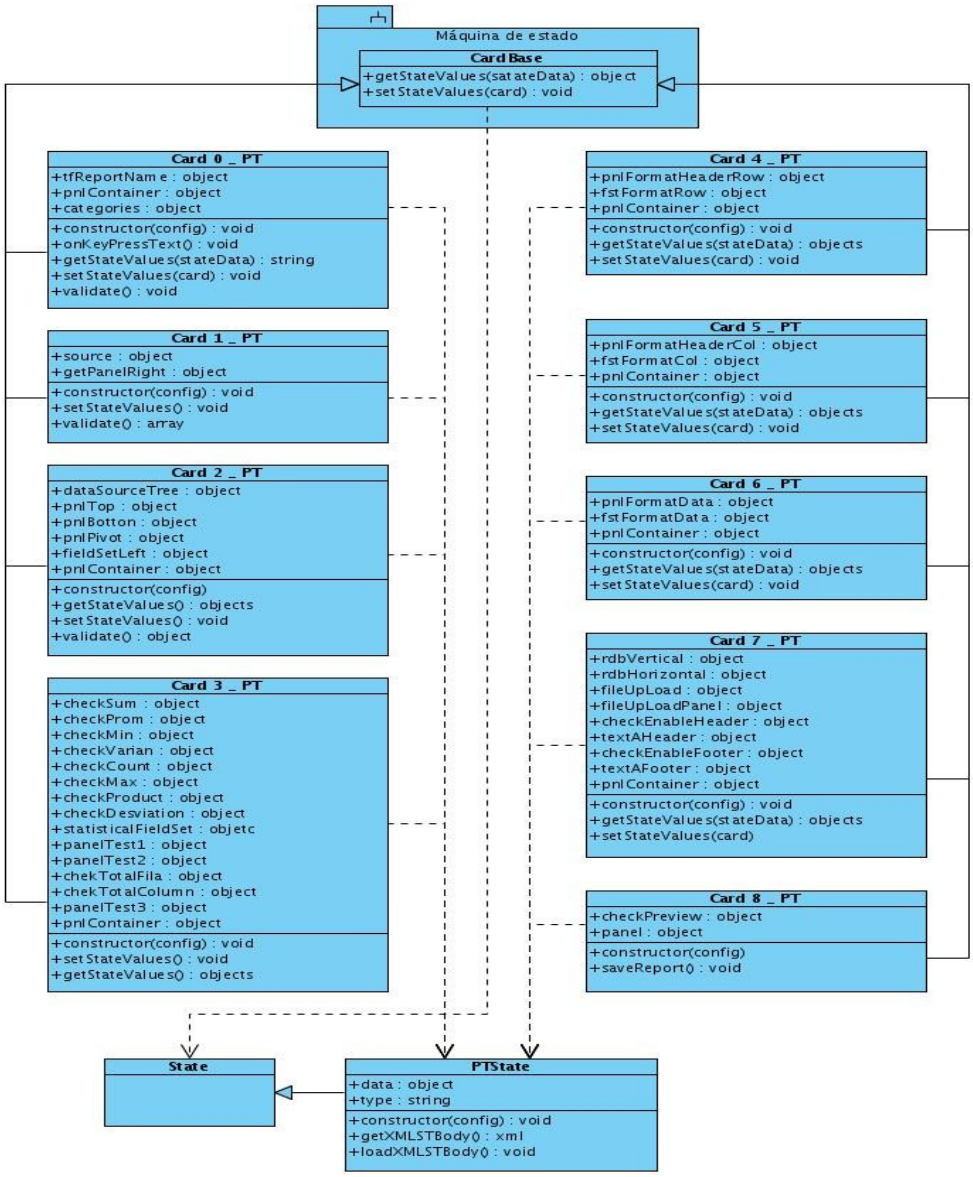

Figura 5: DCD del caso de uso Diseñar tipo de reporte de Tabla Pivote

Descripción de las clases:

- **Card 0 _ PT:** Representa la información correspondiente al nombre y la categoría del reporte.
- **Card 1 _ PT:** Representa la reutilización del componente de Selección de la fuente de datos donde se seleccionan a partir de una Consulta, Función, Tabla o Vista los campos disponibles para incluir en el reporte.
- **Card 2 _ PT:** Representa la reutilización del componente de Selección de los campos donde se seleccionan de los campos disponibles los campos que van a formar parte de las filas y las columnas y el elemento pivote del reporte.
- **Card 3 _ PT:** Representa la información correspondiente a las operaciones que se le pueden aplicar al elemento pivote del reporte tales como: Suma, Promedio, Mínimo, Varianza, Desviación Estándar, Contar, Producto y Máximo.
- **Card 4 _ PT:** Representa la reutilización del componente Configurar formato de celda donde se define el formato de las filas del reporte en cuanto a la Fuente, Fondo y Alineación.
- **Card 5 _ PT:** Representa la reutilización del componente Configurar formato de celda donde se define el formato de las columnas del reporte en cuanto a la Fuente, Fondo y Alineación.
- **Card 6 _ PT:** Representa la reutilización del componente Configurar formato de celda donde se define el formato del elemento pivote del reporte en cuanto a la Fuente, Fondo, Alineación, y Opciones de Datos.
- **Card 7 _ PT:** Representa la información correspondiente a la Configuración de la página del reporte donde se especifica la orientación del papel, cargar o no un logotipo para el encabezado y habilitar o no un texto para el encabezado y/o pie del reporte.
- **Card 8 _ PT:** Brinda la opción para visualizar o no el reporte después de guardado.

- **PTState:** Representa los datos que se manejan en el Asistente para construir o modificar los reportes de tabla pivote.

Diagrama de clases del Asistente de Reportes:

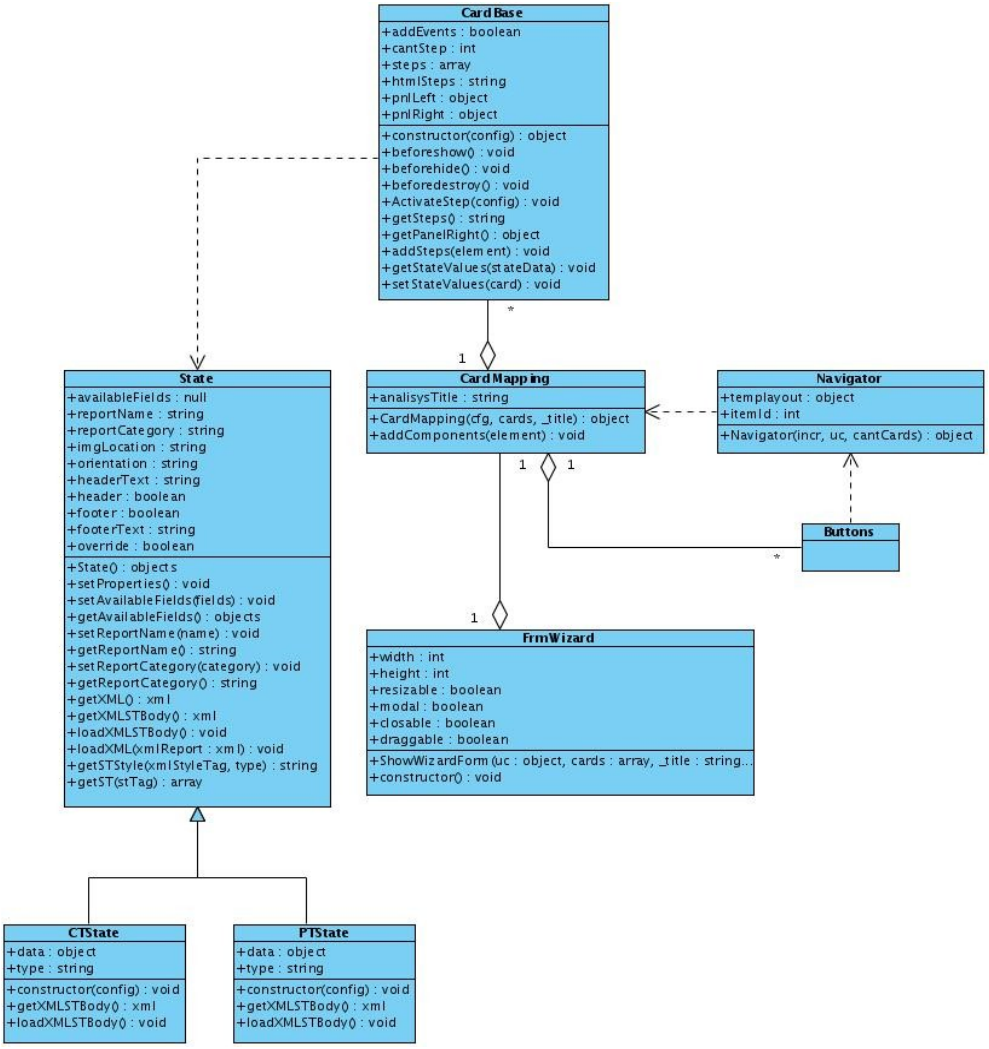

Figura 6: DCD del Asistente de Reportes

Descripción de las clases:

- **CardBase**: Representa la definición básica de todas las cartas que se utilizan en el Asistente.
- **State**: Clase base que representa la definición básica de los datos que se manejan en el Asistente para construir o modificar el reporte.
- **PTState:** Especificación de la clase State que representa los datos que se manejan en el Asistente para construir o modificar los reportes de tabla pivote.
- **CTState:** Especificación de la clase State que representa los datos que se manejan en el Asistente para construir o modificar los reportes de tabla cruzada.
- **CardMapping**: Crea los botones necesarios para navegar en el Asistente de Reportes y registra el evento que valida cada una de las cartas.
- **FrmWizard**: Representa la configuración visual del Asistente de Reportes con sus correspondientes propiedades.
- **Navigator**: Habilita y controla los estados de los botones y de las cartas.
- **Buttons:** Clase de Ext JS contenida en la clase CardMapping encargada de ejecutar las acciones de la clase Navigator.

3.5 Conclusiones Parciales

Según las actividades realizadas en este capítulo para documentar el diseño del Asistente de diseño predeterminado de reportes se concluye lo siguiente:

- Se describieron los componentes que conformarán el Asistente de Reportes.
- Se realizó la descripción de los elementos que conforman el XML de persistencia para los reportes de tabla pivote y tabla cruzada mediante el Asistente.
- Se especificó el estilo arquitectónico y los patrones de diseño utilizados.
- Se realizaron los diagramas de clases y de interacción del diseño.
- La documentación del diseño del Asistente de Reportes se logró cumpliendo los objetivos necesarios con la realización de las actividades correspondientes para alcanzar el avance del proceso

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN

Introducción

En este capítulo se describe la implementación del Asistente de Reportes en términos de componentes, se presenta el modelo matemático en el cual se basó la implementación del componente Máquina de estados, se realiza el Modelo de implementación que incluye los diagramas de componentes, se muestra el Estándar de codificación utilizado, y se valida la implementación mediante la ejecución de las pruebas a Nivel de Desarrollador aplicadas al caso de uso Diseñar tipo de reporte de Tabla Pivote con Asistente.

4.1 Implementación del Asistente de Reportes

Para la implementación del Asistente de Reportes se reutilizaron los componentes Seleccionar fuente de datos, Configurar parámetros**,** Seleccionar campos**,** Configurar formato de celda, y Generar vista previa en html, los cuales permiten la realización de los requisitos necesarios para la elaboración de los reportes. Además de estos componentes se utilizó la Máquina de estados, que es el mecanismo de implementación utilizado para realizar la secuencia de pasos definida, estableciendo el control de la navegación en el Asistente de Reportes.

La Máquina de estados es un componente que se implementó con el objetivo de dar soporte al Asistente de Reportes representando su comportamiento con entradas y salidas, en donde las salidas dependen no solo de las señales de entradas actuales, sino, también, de las anteriores. Este componente básicamente se encarga de representar la definición básica de todas las cartas que se utilizan, crea los botones necesarios para navegar en el Asistente y habilita y controla el estado de las cartas y los botones.

La implementación de la Máquina de estados se basó en el siguiente Modelo Matemático:

La Máquina de estados es el mecanismo de implementación que representa el comportamiento del Asistente de Reportes con entradas y salidas. Dicho mecanismo está formado por un conjunto de estados que sirven de intermediarios en esta relación de entradas y salidas. La bibliografía suele emplear el término autómata como sinónimo de Máquina de estados. De manera específica la Máquina de estados implementada es un autómata finito determinista (AFD) cuyo estado de llegada está unívocamente determinado por el estado de salida y el carácter leído por el autómata, lo que significa que para cada estado en que se encuentre el autómata, y con cualquier símbolo del alfabeto leído, existe siempre a lo más una transición posible desde ese estado y con ese símbolo.

Definición Formal de un autómata:

Un autómata finito está definido por una 5-tupla $N = (S, \Sigma, \S, S_0, F)$.

Donde:

S: Es un conjunto finito de estados.

Σ: Alfabeto del lenguaje. Conjunto de símbolos de entrada del autómata.

§: Función de transición definida.

S_0: Es el estado inicial del autómata.

F: Es el conjunto de estados finales o de aceptación del autómata.

A continuación se muestra el modelo matemático realizado para representar el AFD en el cual se basó dicha implementación:

Definición formal del autómata modelado:

N = ({ Q_0, Q_1, Q_2, Q_3, Q_4 }, { anterior, siguiente, cancelar, finalizar}, §, Q_0, { Q_3, Q_4})

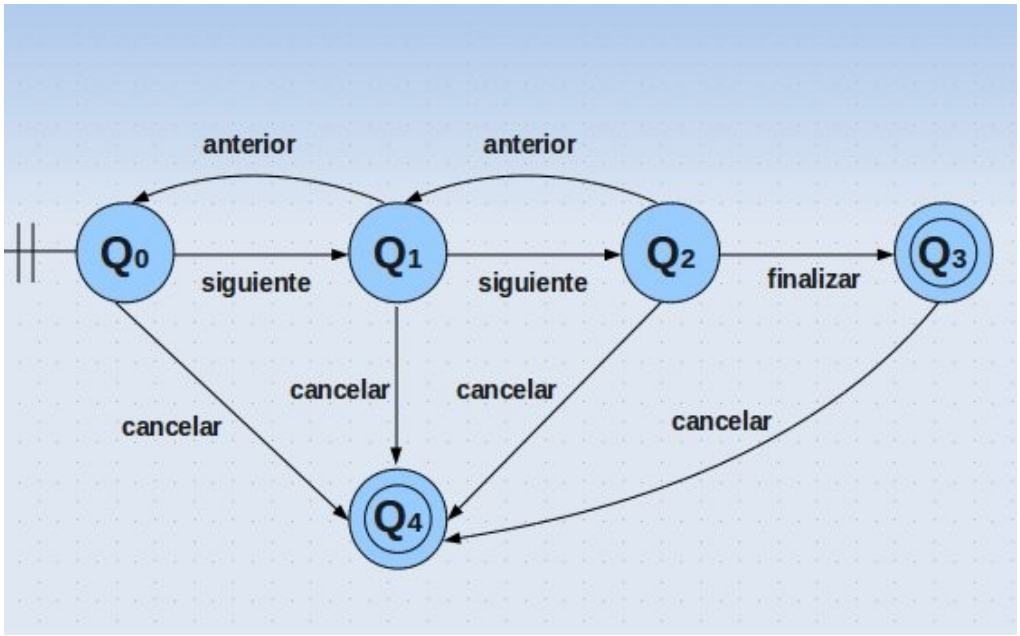

Figura 7: Modelo matemático de la Máquina de estados (AFD)

En los anexos se encuentra la leyenda del autómata definido para el caso de uso Diseñar tipo de reporte de Tabla Pivote con Asistente (ver Anexo 14).

4.2 Modelo de implementación

El Modelo de implementación es de gran utilidad para la implementación del sistema. Organiza el trabajo para los desarrolladores y describe cómo se implementan en términos de componentes los elementos del modelo de diseño. *(Ivar Jacobson 2000)*

Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones, se utiliza para mostrar la estructura de alto nivel del modelo de implementación en términos de subsistemas de implementación y las relaciones entre los componentes. *(Ivar Jacobson 2000)* A continuación se muestran los diagramas de componentes pertenecientes a la Máquina de estados y al caso de uso para los reportes de tabla pivote. En los Anexos se encuentra el diagrama de componentes para los reportes de Tabla Cruzada (*ver Anexo 15*).

Diagrama de componentes de la Máquina de estados:

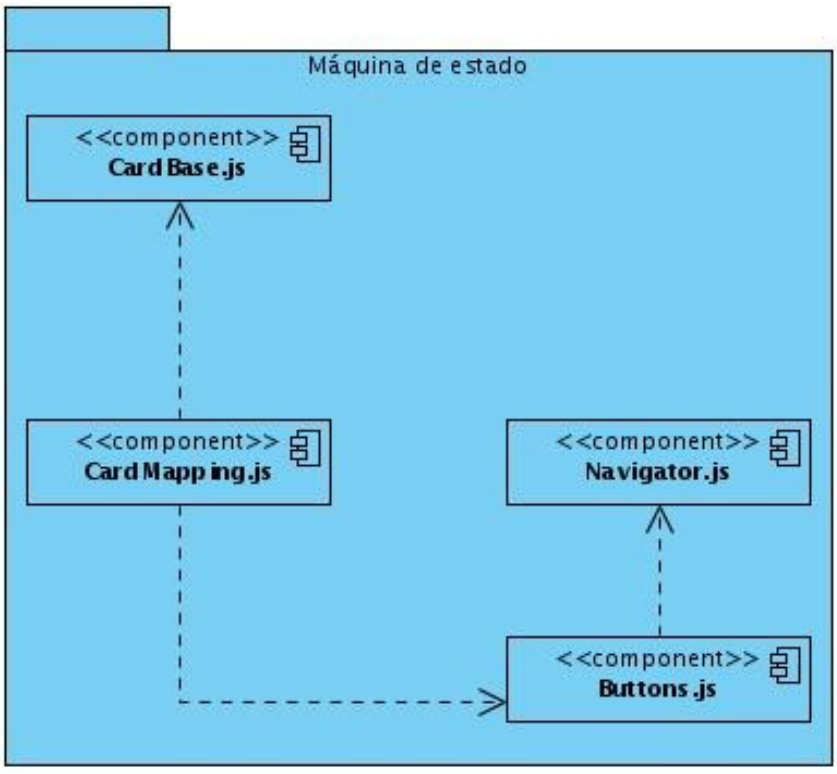

Figura 8: DC de la Máquina de estados

4.3 Diagrama de componentes para los reportes de Tabla Pivote:

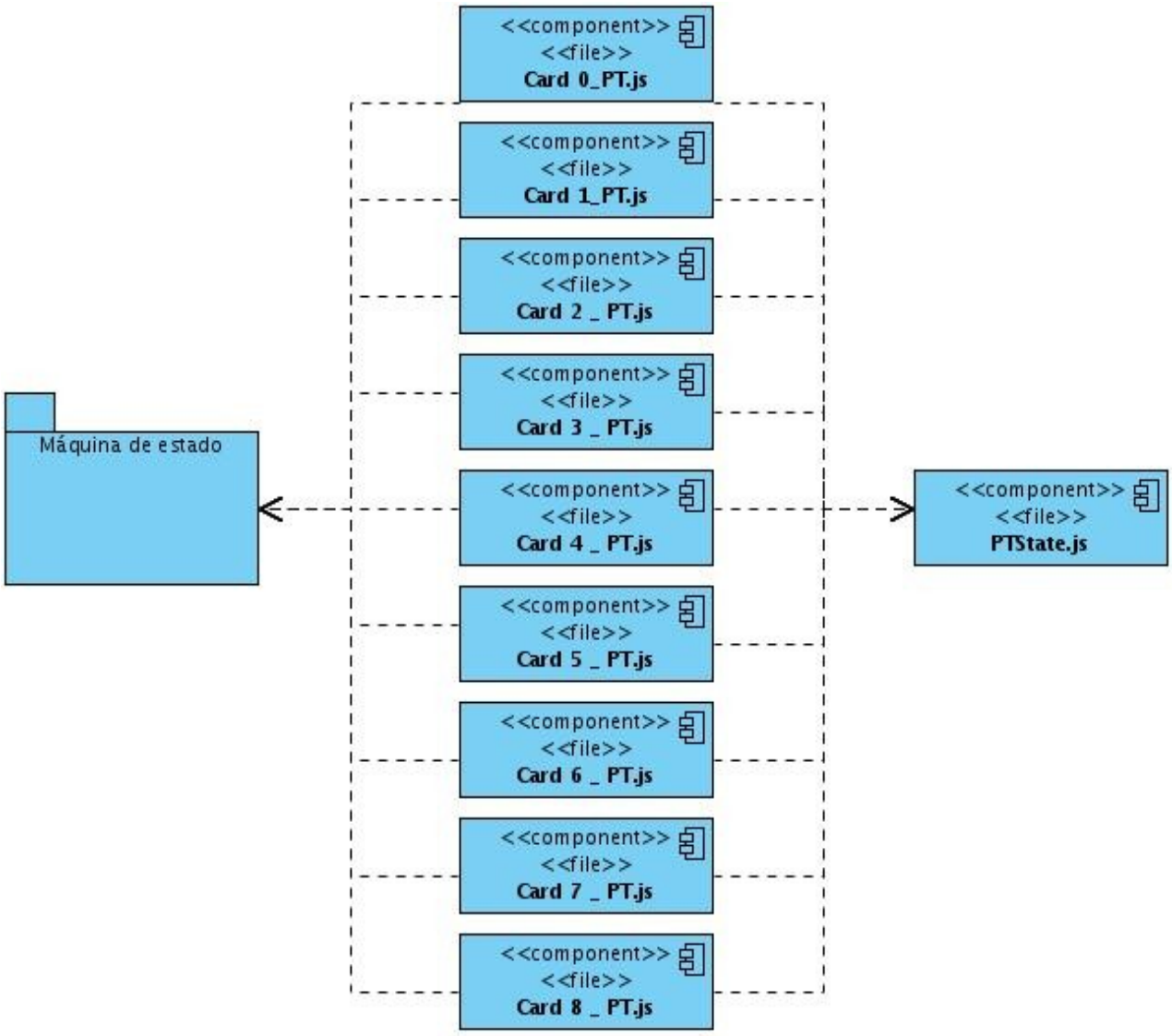

Figura 9: DC para los reportes de Tabla Pivote

4.4 Estándar de codificación

Se definen estándares de codificación porque un estilo de programación homogéneo en un proyecto permite que todos los participantes lo puedan entender en menos tiempo y que el código en consecuencia sea mantenible. *(Manuel Arias Calleja 2008)*

Un estándar de codificación comprende todos los aspectos de la generación de código. Usar técnicas de codificación sólidas y realizar buenas prácticas de programación con vistas a generar un código de alta calidad es de gran importancia para la calidad del software y para obtener un buen rendimiento. *(Maidelys Machado Díaz y Yader Luis Coca Ribas 2009)*

Estilo de codificación utilizado

- Todas las clases tienen que tener el mismo nombre que el fichero al cual pertenecen. Los nombres de las clases estarán antecedidos por el espacio de nombres al cual pertenece.
- Los bloques de código siempre deben estar encerrados por llaves (incluso, si solo constan de una línea).
- Indentación: tamaño = 4 (espacios) para:
  - Declaraciones dentro de las clases.
  - Enunciado dentro de métodos y funciones.
  - Enunciados dentro de bloques de comandos.
- Espacios en blanco:
  - Antes de abrir paréntesis.
  - Luego de abierto el paréntesis.

- Antes de abrir corchetes.
- Luego de la coma en declaraciones.

- Líneas en blanco:
  - Antes de la declaración de una constante.
  - Luego de la declaración de un campo al comienzo del cuerpo de una declaración/método.
- Nueva línea:
  - En enunciados vacíos.
- Líneas de código:

Máximo tamaño de línea (ancho) 300 caracteres.

4.5 Documentación de la ejecución de las pruebas mediante la validación a nivel de desarrollador

Las pruebas se realizan con el objetivo de encontrar y documentar los defectos que pueden afectar la calidad del software, verificar que el software trabaje como fue diseñado, validar y probar los requisitos que debe cumplir el software y la implementación correcta de estos requisitos.

Existen varios niveles de prueba, donde cada nivel contiene una técnica de prueba específica según los atributos de calidad que se deseen verificar. Las técnicas a su vez pueden derivarse en distintos tipos de pruebas los cuales utilizan métodos para llevar a cabo la ejecución de las pruebas al software.

Nivel de Prueba

Los programadores siempre prueban el código durante todo el ciclo de desarrollo, realizando de esta forma las llamadas pruebas a Nivel de Desarrollador, que es el primer nivel de pruebas, diseñadas e

implementadas por el equipo de desarrollo.

Técnica de Prueba

La técnica de prueba que se determinó emplear para implementar las pruebas, es la de Pruebas de Funcionalidad, encargadas de analizar cada funcionalidad implementada para verificar que se cumplan todos los requisitos establecidos y de esta forma satisfacer las necesidades existentes.

Tipo de Prueba

El tipo de prueba que se utilizará es el de Pruebas Funcionales, que tienen como objetivo asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados.

Método de Prueba

Las Pruebas Funcionales utilizan el método de Caja Negra, que permite demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto. Este método consiste en ejecutar cada caso de uso, flujo de caso de uso, o función, usando datos válidos e inválidos, para verificar lo siguiente:

- Que se aplique apropiadamente cada regla de negocio.
- Que los resultados esperados ocurran cuando se usen datos válidos.
- Que sean desplegados los mensajes apropiados de error y precaución cuando se usan datos inválidos.

Caso de Prueba

La aplicación del método de Caja Negra se realizará mediante un Caso de Prueba, diseñado para el caso de uso seleccionado, Diseñar tipo de reporte de Tabla Pivote con Asistente. El Caso de Prueba constituye un conjunto de entradas, condiciones de ejecución y resultados esperados para cumplir un

objetivo en particular o una función esperada. Para diseñar el Caso de Prueba se utilizó la técnica de Partición de Equivalencia, que divide el campo de entrada de un programa en variables de equivalencia con juegos de datos de entrada y salida. Las variables de equivalencia representan un conjunto de estados válidos y no válidos para las condiciones de entrada de un programa.

El caso de prueba realizado cuenta con la sección a probar Diseñar reporte tabla pivote, formada por los siguientes escenarios:

- EC 1: Construir Asistente.
- EC 2: Cancelar Asistente.
- EC 3: Nombrar reporte.
- EC 4: Seleccionar fuente de datos.
- EC 5: Seleccionar campos.
- EC 6: Configuración general.
- EC 7: Formato de fila.
- EC 8: Formato de columna.
- EC 9: Formato de datos.
- EC 10: Configuración de la página.
- EC 11: Guardar reporte.
- EC 12: Reporte duplicado.
- EC 13: Visualizar reporte.

La documentación detallada de la ejecución de las pruebas se encuentra en el documento CP_RD_Disenar_reporte_tabla_pivote.

Resultados de las Pruebas

Después de realizar las pruebas se encontraron tres dificultades, las cuales quedaron registradas en el documento CP_RD_Disenar_reporte_tabla_pivote y solucionadas posteriormente, quedando listo el Asistente de Reportes para pasar a otro nivel de prueba.

En los Anexos se encuentran las imágenes perteneciente a las No Conformidades encontradas (*ver Anexo 19*), las cuales se muestran a continuación:

NC 1: En el EC 7 Formato de Fila se detectó la primera No conformidad no significativa, la cual no permitía cambiar manualmente las opciones de color.

NC 2: En el EC 8 Formato de columna se detectó la segunda No conformidad no significativa, la cual no permitía cambiar manualmente las opciones de color.

NC 3: En el EC 5 Seleccionar campos se detectó la tercera No conformidad significativa, la cual permitía seleccionar más de un campo para las filas y columnas.

4.6 Conclusiones Parciales

Según las actividades realizadas en este capítulo para validar el diseño, implementando el Asistente de diseño predeterminado de reportes, con un reporte predeterminado y validar la implementación mediante la documentación de las pruebas realizadas a dicho reporte predeterminado implementado se concluye lo siguiente:

- Se realizaron los diagramas de componentes pertenecientes a la Máquina de estados y a los casos de uso para los reportes de tabla cruzada y tabla pivote.
- El Asistente de Reportes cuenta con cuatro componentes, tres de ellos son reutilizados en la implementación del mismo y el restante es el mecanismo de diseño (Máquina de estados) específicamente implementado para el Asistente en término de componente, facilitando de esta forma su reutilización en otros módulos del sistema Generador Dinámico de Reportes.
- Se realizaron las pruebas donde se encontraron dos no conformidades, las cuales quedaron registradas y solucionadas posteriormente.
- La implementación del Asistente de Reportes se logró cumpliendo las funcionalidades necesarias que se definieron para obtener el diseño predeterminado de los reportes de tabla cruzada y tabla pivote.
- El sistema quedó validado verificando que cumpliera con las funcionalidades definidas, mediante las pruebas realizadas donde se detectaron y solucionaron los errores identificados.

CONCLUSIONES

Una vez concluida la presente investigación se arribó a las siguientes conclusiones:

- A partir del estudio realizado a los procesos de diseño predeterminado de reportes que realizan los diferentes Asistente de Reportes se determinaron los tipos de reportes a incluir en el sistema Generador Dinámico de Reportes para la versión 2.0.
- La documentación del Análisis y Diseño del Asistente de Reportes se obtuvo a través de la realización de las actividades correspondientes para alcanzar el avance del proceso.
- Se describió el mecanismo de diseño de la Máquina de estados implementado para el Asistente en término de componente, facilitando de esta forma su reutilización en otros módulos del sistema Generador Dinámico de Reportes.
- La implementación del Asistente de Reportes se logró cumpliendo las funcionalidades necesarias que se definieron para obtener el diseño predeterminado de los reportes de Tabla Pivote.
- El sistema quedó validado verificando que cumpliera con las funcionalidades definidas para obtener los reportes de Tabla Pivote, mediante las pruebas realizadas donde se detectaron y solucionaron los errores identificados.

RECOMENDACIONES

- Implementar el caso de uso Diseñar tipo de reporte Columnar con Asistente.
- Reutilizar el componente genérico implementado Máquina de estados en los módulos Diseñador de Modelo y Diseñador de Consulta del sistema Generador Dinámico de Reportes V2.0.

BIBLIOGRAFÍA

1. Adobe PDF Library 8.0. Manual Generador de Reportes de Pentaho. Julio 2008. [cited 18 Octubre 2010].
2. Antonio C. Manifiesto de Reglas de Negocio. Noviembre 2003. [cited 15 Enero 2011].
3. Business Objects documentation. Manual del usuario de Crystal Reports XI. Febrero 2005. [cited 2 Noviembre 2010]. Available from world wide web: <http://www.wiener.edu.pe/manuales2/5to-ciclo/PROGRAMACION-VISUAL-4/Manual-Crystal-Reports-11-XI.pdf>.
4. Business Objects SA. Asistentes disponibles en Crystal Reports. 2005. [cited 23 Octubre 2010]. Available from world wide web: <http://msdn.microsoft.com/es-es/library/ms227804%28v=VS.80%29.aspx>.
5. Carlos Billy Reynoso. Introducción a la Arquitectura de Software. Marzo 2004. [cited 27 Febrero 2011]. Available from world wide web:
6. Daily WP. Definición de reporte. *Definición.de* 2008. [cited 22 Octubre 2010]. Available from world wide web: <definicion.de/reporte/>.
7. Emilio Arias. Microsoft Word - Comparativa _2_.doc. Julio 2007. [cited 17 Octubre 2010]. Available from world wide web:
8. Evelyn Menéndez Alonso. Herramientas CASE para el proceso de desarrollo de Software. *Monografias.com S.A.* Diciembre 2009. [cited 20 Noviembre 2010]. Available from world wide web: <http://www.monografias.com/trabajos73/herramientas-case-proceso-desarrollo-software/herramientas-case-proceso-desarrollo-software2.shtml>.
9. Ezequiel Montes. NetBeans 6.9. Junio 2010. [cited 20 Noviembre 2010]. Available from world

wide web: <http://netbeans.org/community/releases/69/>.

10. Graig Larman. UML y PATRONES Introducción al análisis y diseño orientado a objetos. Febrero 2004. [cited 1 Marzo 2011].
11. GrapeCity. ASP.NET - ActiveReports | GrapeCity. *.NET Reporting Tool for Silverlight, WinForms* 2011. [cited 25 Octubre 2010]. Available from world wide web: <http://www.datadynamics.com/Products/ActiveReports/Overview.aspx>.
12. Ignacio López Vellon. iReport AME4Java Endesa. *iReport AME4Java Endesa* Abril 2010. [cited 15 Octubre 2010]. Available from world wide web: <https://ame.endesa.es/confluenceame/display/PP4JENDESA/iReport>.
13. iReport - Dev - Confluence. iReport - Dev - Confluence. Septiembre 2008. [cited 15 Octubre 2010]. Available from world wide web: <http://amap.cantabria.es/confluence/display/DEV/iReport>.
14. Iván Garcerant. Tecnología y SynergixVisión de Synergix de los Sistemas de Información y la Ingeniería del Software. Agosto 2008. [cited 10 Enero 2011]. Available from world wide web: <http://synergix.wordpress.com/2008/07/10/modelo-de-dominio/>.
15. Ivar Jacobson. *El Proceso Unificado De Desarrollo De Software*. Pearson Addison Wesley, 2000 [cited 22 Noviembre 2010]. Available from world wide web:
16. James Rumbaugh, y Grady Booch Ivar Jacobson. *The Unified Modeling Language Reference Manual*. Addison-wesley, 2010 [cited 24 Noviembre 2010].
17. Jimmy Wales. JSON. Noviembre 2010a. [cited 5 Diciembre 2010]. Available from world wide web:
18. Jimmy Wales. NetBeans. Noviembre 2010b. [cited 27 Noviembre 2010]. Available from world wide web: <http://netbeans.org/community/releases/roadmap.html>.

19. José Manuel López Franco. Características de XML. Octubre 2001. [cited 13 Octubre 2010]. Available from world wide web: <http://trevinca.ei.uvigo.es/~txapi/espanol/proyecto/superior/memoria/node156.html>.
20. Josep Curto Díaz. Review Pentaho Reporting 3.5 for Java Developers. Abril 2010. [cited 20 Octubre 2011]. Available from world wide web: <http://informationmanagement.wordpress.com/2010/03/16/review-pentaho-reporting-3-5-for-java-developers/>.
21. Juan J. Merelo. Introducción al lenguaje de programación JavaScript. 2009. [cited 3 Diciembre 2010]. Available from world wide web: latecladeescape.com. Metodologías de desarrollo del software. Octubre 2010. [cited 20 Octubre 2010]. Available from world wide web: <http://latecladeescape.com/ingenieria-del-software/metodologias-de-desarrollo-del-software.html>.
22. Maidelys Machado Díaz, y Yader Luis Coca Ribas. SACCEM: Módulo de gestión de la información del Departamento de Supervisión del CCEEM versión 1.0. Junio 2009. [cited 25 Abril 2011].
23. Manuel Arias Calleja. Estándares de codificación. Enero 2008. [cited 4 Mayo 2011].
24. Marcos Legido Hernández. Manual JavaScript. Características. Octubre 2009. [cited 3 Octubre 2010].
25. María Isabel García Arenas. Curso XML Introducción. 2009. [cited 23 Octubre 2010].
26. MCE Technical Publications. iReport. Septiembre 2008. [cited 4 Noviembre 2010]. Available from world wide web: <http://www.mceinc.com/products/Manuals/WebManuals/iReport%2042-02-S026.pdf>.
27. Motion Control Engineering. User Guide, iReport. Septiembre 2008. [cited 15 Octubre 2010].

Available from world wide web: <http://www.mceinc.com/products/Manuals/WebManuals/iReport%2042-02-S026.pdf>.

28. Página oficial de Report Manager. 2008. [cited 16 Octubre 2010]. Available from world wide web: <http://reportman.sourceforge.net/indexes.html>.
29. Ricardo Balduino. Introduction to OpenUP (Open Unified Process) - OpenUP.doc. 2008. [cited 22 Noviembre 2010]. Available from world wide web: <http://www.numbersix.com/news/n6articles/OpenUP.html>.
30. Roger S. Pressman. *Ingeniería del Software. Un Enfoque práctico*. Quinta Edición Febrero 2005 [cited 15 Noviembre 2010].
31. SAP Business Objects. Chile - SAP Business Objects l Crystal Reports l Crystal Reports Visual Advantage. 2008a. [cited 15 Octubre 2010]. Available from world wide web: <http://www.sap.com/chile/solutions/sapbusinessobjects/sme/reporting/crystalreports/index.epx>.
32. SAP Business Objects. Crystal Solutions | Crystal Reports. 2008b. [cited 15 Octubre 2010]. Available from world wide web: <http://www.crystalsolutions.com.ar/productos/crystalreports.html>.
33. Shea Frederick, Colin Ramsay, y Steve 'Cutter' Blades. *Learning Ext JS*. Akshara Aware, Noviembre 2008.
34. Vladimir Díaz Ordaz. Gravitar Información sin límites. 2011. [cited 18 Octubre 2010]. Available from world wide web: <http://www.gravitar.biz/index.php/herramientas-bi/pentaho/caracteristicas-pentaho/>.