UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
CENTRO DE TECNOLOGÍAS DE GESTIÓN DE DATOS

Implementación del módulo Diseñador de Reportes para el Generador Dinámico de Reportes incluido en la Plataforma de Ayuda para la Toma de Decisiones y Soluciones Integrales

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

AUTOR

Aurelio Rodriguez Durán

TUTOR

Ing. Miguel Lezcano Ramos

Ciudad de La Habana, Cuba

Junio, 2010

DECLARACIÓN DE AUDITORÍA

Declaro que soy el único autor 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 _____ del año _____.

Nombre Autor

_____

Nombre del Tutor

_____

AGRADECIMIENTOS

Agradezco a:

Todos los profesores y estudiantes que de alguna forma influyeron en mi formación profesional durante mi estancia en la universidad.

Miguel Lezcano mi tutor, por sus revisiones y consejos oportunos durante la conformación del documento de tesis.

Mi novia Arianna por llenarme la vida de felicidad.

DEDICATORIA

Dedico el presente trabajo a mis padres Aurelio Rodriguez Prades y Dora Durán Borges y a mis hermanos Reynerio, Emilio y Yoel.

RESUMEN

Los diseñadores de reportes son herramientas complementarias de los sistemas generadores de informes. Permiten la construcción de reportes que se ajustan a las necesidades y gustos del usuario, siendo posible sintetizar la información esencial de sus negocios.

El presente trabajo de diploma describe la implementación del módulo Diseñador de Reportes correspondiente al Generador Dinámico de Reportes basado en tecnologías web. La herramienta implementada constituye una aplicación rica en Internet, ya que explota al máximo la experiencia del usuario haciendo uso de JavaScript, AJAX y otras tecnologías subyacentes.

El estudio basó su fundamentación teórica en el análisis valorativo de diseñadores de reportes existentes, facilitando la selección de técnicas apropiadas para su desarrollo. La validación del Diseñador de Reportes incluyó la realización de pruebas a varios niveles que permitieron detectar y corregir ágilmente los defectos.

Este módulo es reconocido por su flexibilidad e integración. Con él se potenció la personalización y la calidad de presentación de los reportes. Fue desplegado en varios entornos reales, mostrando una evolución guiada por la colaboración de sus clientes. Esto se ha concretado en un incremento de funcionalidades y oportunidades de uso en nuevos negocios.

TABLA DE CONTENIDOS

INTRODUCCIÓN

El desarrollo de las TIC (Tecnologías de la Información y las Comunicaciones) abona al perfeccionamiento del almacenamiento y manejo de datos. En Cuba se considera necesario y útil, dominar e introducir en la práctica social dichas tecnologías. Además, se fomenta el plan y accionar por alcanzar una cultura digital como una de las características imprescindibles del hombre, para aproximarlo más hacia el desarrollo sostenible.

El avance tecnológico influye positivamente en la determinación con exactitud de cuándo, cómo, dónde y con qué objetivo emplear conocimientos adquiridos. El mismo posibilita concretar dicha utopía para los directivos en cualquier empresa y les ofrece un pasaje seguro hacia el éxito.

“Las empresas de hoy en día obtienen, crean y almacenan más datos que nunca. Y, esa información proviene de una gran variedad de fuentes: clientes, proveedores, asociados, consultores, empresas de investigación de mercados, etc. Manejar toda esa información y poder convertirla en información útil para la toma de decisiones se está convirtiendo en una tarea cada vez más complicada. Sin embargo, las potenciales ventajas de utilizar de forma adecuada toda esa información son incluso más grandes, por lo que merece la pena el esfuerzo: mejorar la productividad de los empleados, aumentar el servicio de atención a los clientes, conseguir mayores ingresos, mayor cuota de mercado, etc.” (Arias, 2006)

Como medio primario para sintetizar la información esencial en apoyo a la toma de decisiones se aprovechan los reportes, formados mediante la combinación lógica de datos generalmente dispersos. Seleccionando, organizando y estableciendo relaciones entre dichos datos de forma adecuada, se obtienen resultados precisos para las personas apropiadas dentro de empresas u organizaciones.

Los reportes facilitan a los usuarios observar la marcha del negocio y que de esta forma logren identificar nuevas oportunidades de negocio o servicios. El carácter determinante de los mismos, estimula a un mayor esfuerzo para garantizar su eficiente obtención.

Generar reportes ante el gran volumen de datos y la naturaleza dispersa de los mismos, muestra gran complejidad cuando se realiza de forma manual. Sin embargo, la informatización de dicho proceso ya es un hecho. El desarrollo de las tecnologías de

información y las comunicaciones converge en generadores de reportes como: ActiveReports, Pentaho Reporting, Crystal Reports y Report Manager.

En este mundo por naturaleza cambiante, la práctica ha demostrado que el software sigue una evolución continua y adaptativa, guiada por las demandas constantes de sus usuarios. Simultáneamente, su caudal de oportunidades conduce a su inclusión en las diversas esferas de la vida y su proceso de desarrollo se está convirtiendo en un escenario vital.

Aprovechando la desafiante ola que figura el desarrollo de software en el mundo, para Cuba la creación de la UCI (Universidad de las Ciencias Informáticas) es quizás el más puntero y prometedor de los pasos dados en ese sentido. Este centro de altos estudios existe con el objetivo de producir software y servicios informáticos a partir de la vinculación estudio – trabajo como modelo de formación.

El curso 2008-2009 transcurrido en dicha institución, destaca entre sus resultados la creación del inicialmente nombrado CENTALAD (Centro de Tecnologías de Almacenamiento y Análisis de Datos) y hoy conocido por DATEC (Centro de Tecnologías de Gestión de Datos). Conformemente, anuncia el inicio de un conjunto de proyectos que tributan al desarrollo de herramientas y servicios informáticos, especializados en el almacenamiento y análisis de datos. Entre las tecnologías en construcción relacionadas al análisis de datos, despunta la suite de herramientas PATDSI (Plataforma de Ayuda para la Toma de Decisiones y Soluciones Integrales).

El sistema GDR (Generador Dinámico de Reportes) incluido en dicha plataforma de inteligencia de negocio, presenta las siguientes deficiencias:

- Enlaza los datos con plantillas codificadas manualmente por el usuario que definen el estilo del reporte.
- Carece de un mecanismo lo suficientemente flexible para diseñar reportes personalizados.

En consecuencia, se dilata el tiempo de construcción del reporte y no se logra una temprana retroalimentación necesaria para el usuario final. Además, se reducen las oportunidades a la inteligencia de este último para efectuar actividades de análisis, así como su espíritu creativo durante el proceso de diseño.

Teniendo en cuenta los elementos antes mencionados queda definido el siguiente **problema científico**: ¿Cómo diseñar de forma dinámica y orientada al usuario final los reportes que serán utilizados por el Generador Dinámico de Reportes?

Como **objetivo general** de la investigación para darle solución al problema antes definido se tiene: Implementar partiendo de los artefactos obtenidos durante el análisis, diseño y arquitectura una aplicación Web que facilite el diseño dinámico de reportes.

Por consiguiente el **objeto de la investigación** sería, el proceso de desarrollo de software, enfocando el **campo de acción** en la implementación del módulo Diseñador de Reportes para el sistema informático Generador Dinámico de Reportes.

Lo que conlleva a la siguiente **Idea a Defender**: Con la implementación del módulo Diseñador de Reportes basado en Tecnologías Web, se proveerá mayor usabilidad y operatividad para los usuarios finales durante el diseño de reportes personalizados.

Para dar cumplimiento al objetivo trazado, se definen las siguientes **Tareas de Investigación**:

1. Estudio de sistemas existentes para el diseño dinámico de reportes.
2. Análisis de tecnologías y herramientas de desarrollo de software utilizadas en la actualidad.
3. Estudio de estándares de codificación y patrones de diseño.
4. Selección de las herramientas y lenguajes de implementación a utilizar.
5. Implementación del módulo Diseñador de Reportes.
6. Validación de la solución propuesta.

Para el correcto cumplimiento de las tareas anteriormente enunciadas se tuvieron en cuenta los siguientes **métodos científicos de investigación**:

Métodos teóricos

Histórico-Lógico: Utilizado para explorar los historiales de herramientas y tecnologías, empleadas para implementar el diseño de reportes en sistemas con características similares. Permitió distinguir sus directrices, ventajas y desventajas, así como la forma apropiada de diseñar flexiblemente los reportes.

Análisis-Síntesis*:* Este método fue empleado para hacer un estudio del objeto de investigación mediante la determinación de sus componentes o piezas claves que lo

conforman, analizar particularmente cada elemento y encontrar los nexos que lo hacen funcionar como un todo.

Hipotético-Deductivo: Permitió a partir del problema formular el objetivo y establecer la idea a defender verificando elementos que se puedan inferir para establecer conclusiones previas.

Métodos empíricos

Observación: Se realizó para a partir de la percepción, obtener una valoración en la etapa exploratoria del objeto estudiado. Se enfocó en determinar cómo ocurre realmente el proceso de diseño de reportes y cuáles son los principales elementos en los que se debe trabajar.

Aporte práctico esperado

Con esta investigación se espera poder brindar un módulo para el diseño de reportes personalizados que cumpla con los requisitos pactados con los clientes, mejorando así el análisis y la presentación de datos en los proyectos de la UCI y otras entidades que así lo requieran.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA

1.1 Introducción

Durante el presente capítulo se realiza un análisis valorativo de algunas de las herramientas existentes en el mercado para el diseño de reportes, dando lugar a una visión general del estado del arte de las mismas. Se efectúa un estudio de las metodologías de desarrollo, lenguajes de programación, frameworks (marcos de trabajo) y tecnologías que se utilizan para el desarrollo de software basado en la web. Además, se seleccionan cuales de los elementos estudiados serán aplicados durante la solución del problema, argumentando el por qué de su elección.

1.2 Análisis de herramientas existentes para el diseño de reportes

En la actualidad existen numerosos sistemas en el mercado para diseñar reportes, entre los que se destacan: el Crystal Report Designer, el ActiveReport Designer, el Pentaho Report Designer y el Report Manager Designer.

1.2.1 El diseñador de reportes de Crystal

Crystal Reports para Visual Studio .NET es la herramienta de elaboración de informes estándar para Visual Studio .NET. Permite crear contenido interactivo con calidad de presentación en la plataforma .NET, lo que ha supuesto una ventaja fundamental para Crystal Reports durante años. (Microsoft Corporation, 2009)

El diseñador de Reportes de Crystal permite diseñar y modificar los informes del entorno de programación integrado de Visual Studio .NET. Dicha herramienta puede programarse directamente desde Visual Studio .NET y además no es necesario distribuir el diseñador de reportes con el informe.

Utiliza una funcionalidad de arrastrar y colocar parecida a la que se utiliza en Visual Studio .NET, se arrastra un objeto de informe hasta el diseñador, puede ser un campo de base de datos o un objeto de texto y se utiliza la ventana Propiedades o el menú contextual para dar formato al objeto.

Al crear un informe, la mayor parte del trabajo de diseño del informe se realiza en el diseñador de Reportes. El mismo se divide en dos secciones de informes etiquetadas. Siendo posible colocar objetos de informe, como campos de base de datos, de fórmula, de parámetro y de totales acumulados, en la sección en que se desee que aparezcan.

El diseñador de reportes proporciona un entorno eficaz para el diseño de informes ya que, aunque se trabaja con los datos de una base de datos, se trabaja en un entorno de diseño en el que no es necesario actualizar los datos al modificar el informe. Al incluir un campo en el informe, el diseñador de reportes de Crystal utiliza un marco para identificarlo en la ficha en lugar de recuperar los datos. De este modo, puede añadir y eliminar campos y otros objetos, moverlos y definir fórmulas complejas sin tener que esperar a que se actualicen los datos. (Microsoft Corporation, 2009)

1.2.2 El diseñador de reportes de Pentaho

Pentaho Reporting es la solución proporcionada por Pentaho e integrada en su suite para el desarrollo de informes.

El diseñador de reportes de Pentaho es una herramienta independiente que forma parte de la unidad de reportes de Pentaho, que simplifica el proceso de generación de reportes, permitiendo a los diseñadores de reportes crear rápidamente informes sofisticados y ricos visualmente basados en el proyecto de reportes de Pentaho JfreeReport. (Pentaho Corporation, 2007)

Ofrece un entorno gráfico familiar, con herramientas intuitivas y fáciles de utilizar, y una estructura de reporte bastante acertada y flexible para darle libertad al diseñador de generar reportes que se adapten totalmente a su gusto y necesidad.

Se caracteriza por permitir un diseño de informes flexible:

- Entorno de diseño gráfico.
- Capacidad de uso de plantillas.
- Acceso a datos relacionales, OLAP y XML.

Además está desarrollado para:

- Ser embebible.
- Ser fácil de extender.
- No consumir muchos recursos.
- 100% Java: portabilidad, escalabilidad e integración.

1.2.3 El diseñador de reportes de ActiveReports

ActiveReports es un componente de informes .NET para aplicaciones Windows Forms y de formularios Web. Entre las principales características de ActiveReports figuran la

personalización, un rendimiento rápido, alta calidad y prestaciones multilingües: todas ellas contrastadas mediante su uso en decenas de miles de aplicaciones en todo el mundo. ActiveReports admite exportaciones de datos a todos los formatos de archivo habituales, como PDF, Excel, RTF y TIFF. Además, incluye un diseñador de informes Visual Studio .NET fácil de usar y una potente API (Application Programming Interface). También ofrece un despliegue de tiempo de ejecución armonizado y libre de derechos.

El diseñador de informes de ActiveReports se integra perfectamente en los entornos de desarrollo Visual Studio .NET. Una vez instalado el producto en el equipo del desarrollador, la adición de un informe a un proyecto es tan fácil como añadir una clase o un formulario. (ComponentSource, 1996-2010)

Se considera orientado a usuarios finales, ya que permite su inclusión en las aplicaciones con el fin de que los propios usuarios diseñen y modifiquen sus reportes.

1.2.4 El diseñador de reportes de Report Manager

Report Manager es una aplicación de generación de informes y un conjunto de componentes para Delphi, Builder y Kylix. También puede utilizarse desde otros entornos de desarrollo con el componente ActiveX incluido como son: Visual Basic, Visual FoxPro y cualquier lenguaje de Visual Studio.Net. Se proporciona una librería dinámica estándar con funciones para su uso con el lenguaje GNU C.

Es un producto código abierto bajo el modelo MPL (Mozilla Public License), que incluye permiso de uso en aplicaciones GPL (General Public License), por lo que se puede usar en aplicaciones comerciales, pero cualquier mejora introducida en el motor de impresión debe ser publicada bajo esta licencia.

Funciona en Windows y Linux. Se puede distribuir el diseñador de informes y de esta manera se consigue la modificación de informes sin modificar (recompilar) la aplicación. El resultado puede ser guardado como un archivo con formato Adobe PDF.

Algunos aspectos que se destacan durante el diseño y definición del reporte con el diseñador de reportes de Report Manager son:

- Facilidades para el manejo de subreportes.
- La agrupación permite operaciones de suma, mínimo, máximo, media y desviación estándar.

- Se pueden utilizar grupos como los encabezados y pies de página del subreporte.
- Soporte de multi-idioma, las etiquetas del informe pueden ser definidas en varios idiomas.
- Configuración de página: márgenes, tamaño del papel, orientación de la página.
- Se puede integrar el Diseñador de informes de Report Manager en una aplicación, con solo soltar un componente TRpDesigner y usarlo.
- Consultas parametrizadas que permiten obtener resultados de varias consultas.
- Opciones de edición avanzadas, como selección múltiple, ocultar elementos y las alineaciones.

1.3 Comparación de las herramientas analizadas con la aplicación a desarrollar

Los diseñadores de reportes estudiados aunque admirables en muchos sentidos, muestran algunas desventajas para su empleo:

- Excepto el diseñador de reportes del Report Manager, presentan un elevado coste adquisitivo.
- Son aplicaciones de escritorio, lo que implica la instalación de los mismos en cada máquina cliente.
- Con excepción de los diseñadores de reportes: Pentaho Report Designer y Report Manager Designer, no son multiplataforma ni están completamente orientados al usuario final.

Por otra parte, la aplicación a desarrollar es multiplataforma y posee gran flexibilidad permitiendo al usuario final el diseño de reportes personalizados desde la Web. Además no representa costo de adquisición. Como lo aprovecharán directamente los usuarios finales, se eliminan de una vez los costes de intermediarios, y se mejoran de esta manera los márgenes de beneficios.

La propuesta de este trabajo constituye el resultado de asimilar características admirables, encontradas en los sistemas de su tipo más utilizados en el mundo y combinarlas con las ventajas de la web. Además, al tratarse de una tecnología propia se facilitan los siguientes aspectos: mantenimiento continuo, personalización e integración de la misma con otros sistemas.

1.4 Metodologías de desarrollo

Un proceso de desarrollo de software es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. (Jacobson, y otros, 2000)

Ante la existencia de diversas metodologías de desarrollo de software y de las diferentes características que acompañan a los proyectos, se hace necesario estudiar las primeras con el propósito de poder valorar cual es la que más se ajusta al proyecto en desarrollo.

1.4.1 El Proceso Unificado de Rational

El Proceso Unificado de Rational, es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas software, para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyecto.

El Proceso Unificado utiliza el Lenguaje Unificado de Modelado (Unified Modeling Language, UML) para preparar todos los esquemas de un sistema software. De hecho, UML es una parte esencial del Proceso Unificado – sus desarrollos fueron paralelos. (Jacobson, y otros, 2000)

No obstante, los aspectos definitorios del Proceso Unificado se resumen en tres frases clave –dirigido por casos de uso, centrado en la arquitectura, e iterativo e incremental. Esto es lo que hace único al Proceso Unificado. (Jacobson, y otros, 2000)

El proceso RUP combina un conjunto básico de mejores prácticas aprobadas por el sector con una serie de complementos opcionales del proceso a fin de dar cabida y soporte a proyectos de cualquier envergadura o alcance. A continuación se señalan dichas prácticas:

- Desarrollo de software en forma iterativa.
- Manejo de requerimientos.
- Utiliza arquitectura basada en componentes.
- Modela el software visualmente.
- Verifica la calidad del software.
- Controla los cambios.

1.4.2 El Proceso Unificado Abierto

El Proceso Unificado Abierto (OpenUP) es un proceso unificado que incorpora técnicas ágiles probadas. El resultado es un proceso estructurado, robusto, eficiente y liviano.

Está diseñado para pequeños equipos organizados quienes quieren tomar una aproximación ágil del desarrollo. OpenUP es un proceso iterativo que es mínimo, completo, y extensible. Con esta metodología se valora la colaboración y el aporte de los stakeholders (interesados) por encima de los entregables y la formalidad innecesarios.

El proceso se considera mínimo ya que solamente el contenido fundamental es incluido; completo porque puede ser manifestado como todo el proceso para construir un sistema y extensible dado que al contenido del proceso se le puede extender o adaptar según lo necesitado.

Se caracteriza por cuatro principios básicos que se soportan mutuamente:

- Colaborar para alinear los intereses y un entendimiento compartido.
- Balancear las prioridades en contradicción para maximizar el valor de los stakeholders.
- Enfocarse en articular la arquitectura para facilitar la colaboración técnica, reducir los riesgos y minimizar excesos y trabajo extra.
- Evolución progresiva para reducir riesgos, demostrar resultados y obtener retroalimentación de los clientes.

OpenUP es un proceso iterativo cuyas iteraciones se distribuyen a través de cuatro fases: Concepción, Elaboración, Construcción, y Transición. Cada fase podrá tener tantas iteraciones como se requiera dependiendo del grado de novedad del dominio de negocio, de la tecnología a ser utilizada, de la complejidad de la arquitectura de la solución y del tamaño del proyecto, entre otros factores. (Eclipse, 2008)

1.5 Lenguajes de programación

Un lenguaje de programación es un idioma artificial concebido para crear programas de computador, permitiendo incorporar algoritmos con precisión. Está formado de un conjunto de símbolos y reglas tanto sintácticas como semánticas que especifican su estructura y el significado de sus elementos y expresiones.

Cada lenguaje de programación tiene sus propias características, las cuales lo hacen más potente o más débil para determinada finalidad. El estudio de los mismos garantiza contar con los recursos apropiados para codificar la solución a desarrollar.

1.5.1 Lenguajes de programación en el cliente

1.5.1.1 JavaScript

JavaScript es un lenguaje de programación interpretado que se utiliza principalmente para crear páginas web dinámicas. Legalmente, JavaScript es una marca registrada de la empresa Sun Microsystems, como se puede ver en http://www.sun.com/suntrademarks/. (Eguíluz Pérez, 2009)

La integración de JavaScript y XHTML es muy flexible, ya que existen al menos tres formas para incluir código JavaScript en las páginas web:

- Incluir JavaScript en el mismo documento XHTML.
- Definir JavaScript en un archivo externo.
- Incluir JavaScript en los elementos XHTML.

Entre sus características se distinguen las siguientes:

- No tipado.
- Basado en objetos.
- JavaScript no es Java.
- En la actualidad, pocos navegadores no disponen de soporte para JavaScript.

También se destacan las habilidades a continuación señaladas:

- Control de la visualización y contenido de los documentos HTML.
- Control de objetos del navegador.
- Interactividad mediante formularios HTML.
- Interactividad con el usuario.
- Lectura y escritura de cookies.
- Temporizar acciones.

En cuanto a sus limitaciones, JavaScript fue diseñado de forma que se ejecutara en un entorno muy limitado que permitiera a los usuarios confiar en la ejecución de los scripts. De esta forma, los scripts de JavaScript no pueden comunicarse con recursos que no pertenezcan al mismo dominio desde el que se descargó el script. Los scripts tampoco pueden cerrar ventanas que no hayan abierto esos mismos scripts.

La aparición de las aplicaciones AJAX (Asynchronous JavaScript And XML) programadas con JavaScript le provee una popularidad sin igual dentro de los lenguajes de programación web.

1.5.1.2 Visual Basic Script

VBScript es un lenguaje de programación de scripts del lado del cliente y se comunica con las aplicaciones host mediante Windows Script. Windows Script se utiliza en Microsoft® Internet Explorer y en los Servicios de Microsoft® Internet Information Server. (Microsoft Corporation, 2001)

Microsoft proporciona implementaciones binarias de VBScript para API de Windows® de 32 bits, API de Windows® de 16 bits de Windows y Macintosh®. VBScript está integrado en los exploradores de World Wide Web. VBScript y Windows Script también se pueden utilizar como un lenguaje de ejecución de secuencias de comandos general en otras aplicaciones. (Microsoft Corporation, 2001)

VBScript está basado en Visual Basic, un popular lenguaje para crear aplicaciones Windows. Sin embargo, no todo lo que se puede hacer en Visual Basic se puede hacer en Visual Basic Script. El modo de funcionamiento de Visual Basic Script para construir efectos especiales en páginas web es similar al utilizado en JavaScript y los recursos a los que se puede acceder también son los mismos: el navegador.

1.5.1.3 Applets de Java

Constituyen una manera de incluir código a ejecutar en los clientes que visualizan una página web. Son pequeños programas hechos en Java, que se ejecutan dentro de un cliente web para ampliar las capacidades de interacción del navegador.

Al estar programados en Java y precompilados, la manera de trabajar de éstos varía un poco con respecto a los lenguajes de script como JavaScript.

La principal ventaja de utilizar applets consiste en que son mucho menos dependientes del navegador que los scripts en JavaScript, incluso independientes del sistema operativo del ordenador donde se ejecutan. Java es más potente que JavaScript, por lo que el número de aplicaciones de los applets podrá ser mayor. Además, no hay necesidad de instalar la aplicación en la máquina local y aumenta la seguridad de los sistemas.

Como desventajas en relación con JavaScript cabe señalar que los applets son más lentos de procesar y que tienen entorno de ejecución restringido en la página donde se ejecutan, es decir, no se mezclan con todos los componentes de la página ni tienen acceso a ellos.

1.5.2 Lenguajes de programación en el servidor

1.5.2.1 Tecnología Java Server Pages

JSP, acrónimo de Java Server Pages y especificación de Sun Microsystems, es una interfaz de programación de aplicaciones de servidores Web. En una página jsp se entremezclan bloques de HTML estáticos, y HTML dinámico generados con Java que se ejecutan en el servidor. Su objetivo final es separar la interfaz (presentación visual) de la implementación (lógica de ejecución).

Con JSP pueden crearse aplicaciones web que se ejecuten en variados servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Una página jsp puede procesar formularios Web, acceder a bases de datos y redireccionar a otras páginas. Las páginas jsp son transformadas a un servlet y después compiladas.

El contenedor JSP proporciona un motor que interpreta y procesa una página JSP como un servlet. Al estar basadas en los servlets, las distintas peticiones a una misma página jsp son atendidas por una única instancia del servlet.

Para utilizar JSP, aparte de conocer HTML, es necesario comprender y tener experiencia en la programación con Java, que es un lenguaje de programación Orientado a Objetos por completo.

1.5.2.2 PHP

PHP (acrónimo de "PHP: Hypertext Preprocessor") es un lenguaje de "código abierto" interpretado, de alto nivel, embebido en páginas HTML y ejecutado en el servidor. Existen tres campos en los que se usan scripts escritos en PHP:

- Scripts del lado del servidor.
- Scripts en la línea de comandos.
- Escribir aplicaciones de interfaz gráfica. (utilizando PHP-GTK)

Se puede utilizar en cualquiera de los principales sistemas operativos del mercado, incluyendo Linux, Microsoft Windows, Mac OS X y RISC OS. También soporta la mayoría de los servidores web de hoy en día, entre ellos: Apache, Microsoft Internet Information Server (IIS), Personal Web Server, Netscape e iPlanet, Oreilly Website Pro server, Caudium, Xitami y OmniHTTPd. De modo que, con PHP se tiene la libertad de elegir el sistema operativo y el servidor que se prefiera.

Entre las capacidades de PHP se incluyen: creación de imágenes, archivos PDF y películas Flash (usando libswf y Ming) sobre la marcha. Además puede presentar otros resultados, como XHTM y archivos XML. Quizás la característica más potente y admirable de PHP es su soporte para una gran cantidad de bases de datos: Adabas D, dBase, Empress, FilePro (solo lectura), Hyperwave, IBM DB2, Informix, Ingres, InterBase, FrontBase, mSQL, Direct MS-SQL, MySQL, ODBC, Oracle (OCI7 y OCI8), Ovrimos, PostgreSQL, Solid, Sybase, Velocis, Unix dbm. (the PHP Documentation Group, 1997 - 2008)

1.5.2.3 PHP 5

PHP está dirigido por su núcleo, el motor Zend. PHP 5 incluye el nuevo motor Zend 2.0. En consecuencia PHP 5 se beneficia de un nuevo modelo de objetos que ha sido reescrito por completo, permitiendo un mejor desempeño y más características. Las mismas son enunciadas a continuación:

- Soporte sólido para la POO (Programación Orientada a Objetos).
- Data Objects.
- Mejoras de rendimiento.
- Mejor soporte para MySQL con extensión de reescritura completa.
- Mejor soporte a XML: XPath y DOM (Document Object Model).
- Soporte nativo para SQLite.
- Soporte integrado para SOAP.
- Iteradores de datos.
- Excepciones de errores.

En la actualidad, el paradigma de la Programación Orientada a Objetos es el más utilizado y ello conlleva a destacar los elementos que se incorporan en PHP 5:

- Métodos constructores y destructores.
- Métodos setter y getter.
- Métodos mágicos.
- Clases, objetos y variables estáticas, privadas y protegidas.
- Clases abstractas.
- Interfaces.
- Abstracción de datos.
- Standard PHP Library (SPL).
- Clases extendidas, excepciones, iteradores.

1.5.2.4 Tecnología ASP

ASP (Active Server Pages) es la tecnología desarrollada por Microsoft para la creación de páginas dinámicas del servidor, combinando HTML con un lenguaje de secuencia de comandos, o lenguaje script. Los lenguajes más usados actualmente para programar el código script dentro de las páginas ASP son VBScript y Jscript (JavaScript de Microsoft), aunque también se pueden utilizar otros como Rexx, Perl y Python.

Esta tecnología permite integrar componentes ActiveX en las páginas, para de esta forma acceder a recursos residentes en el propio servidor. El hecho de que las páginas ASP no funcionen sobre un servidor que no sea IIS y que éste, a su vez, no se ejecute bajo un sistema operativo que no sea de Microsoft, les resta competitividad con respecto a otras tecnologías actuales que funcionan bajo diferentes sistemas operativos. (González, y otros, 2000)

1.5.2.5 Tecnología ASP.NET

Versión para la plataforma .NET de la tecnología ASP. Presenta tres partes muy diferenciadas:

- Web Forms.
- Server Controls.
- Web Services.

ASP.NET se ha construido bajo los siguientes principios:

- Facilidad de desarrollo: ASP.NET introduce un nuevo concepto, los "server controls", que permiten a modo de etiquetas HTML tener controles manejados por el servidor que identifican el navegador usado adaptándose para cada navegador. Permite la elección del lenguaje de programación, así como independencia de la herramienta de desarrollo e incorpora una rica biblioteca de clases que evita obtener componentes de otras empresas.
- Alto rendimiento y escalabilidad: El código es compilado para ser ejecutado en el CLR (Common Language Runtime). El resultado es de 3 a 5 veces superior en velocidad que las antiguas páginas ASP y cuenta con un rico sistema de cache.
- Mejora de la fiabilidad: ASP.NET es capaz de detectar pérdidas de memoria, problemas con bloqueos y protección ante caídas.

- Fácil distribución e instalación: Una aplicación ASP.NET se instala tan fácilmente como copiando los ficheros que la componen. No es necesario registrar ningún componente, tan solo copiar los ficheros al web.

1.6 XML y XSLT

XML son las siglas de Extensible Markup Language, un lenguaje de marcas desarrollado por el W3C (World Wide Web Consortium). XML es una versión de SGML (Standard Generalized Markup Language), diseñado especialmente para los documentos de la web.

XML es uno de los formatos más utilizados para el intercambio de información estructurada hoy en día: entre programas, entre las personas, entre ordenadores y personas, tanto a nivel local como a través de redes. (W3C, 2010)

Esta tecnología está dotada de capacidades profundamente necesarias para el desarrollo de la web en el presente:

- XML puede ser leído tanto por máquinas como por personas.
- Los documentos XML siguen unas reglas de generación sencillas pero bien definidas que les hace fácilmente procesables.
- XML utiliza el estándar de codificación de caracteres UNICODE, lo que facilita la internacionalización.
- Facilita encontrar lo que se está buscando: exactitud y agilidad.
- Permite el intercambio de información sobre Internet.
- XML es fácil de aprender y de usar.

XML es una tecnología sencilla que tiene a su alrededor otras tecnologías que la complementan y la hacen mucho más grande y potente. Con todas las tecnologías relacionadas, representa una manera más avanzada de hacer las cosas. Su novedad más admirable está en permitir compartir los datos con los que se trabaja de forma segura, fiable y fácil a todos los niveles, por todas las aplicaciones y soportes.

Entre las tecnologías que a menudo acompañan a XML se distingue XSLT. De manera general XSL (eXtensible Style Language) presenta las siguientes características:

- XSL consiste en tres partes:
    - XSLT – lenguaje para la transformación de documentos XML.
    - XPath – lenguaje para la navegación en documentos XML.
    - XSL-FO – lenguaje para el formateo de documentos XML.

- XSLT es un lenguaje de programación de hoja de estilos para la transformación de documentos XML en otros documentos XML, HTML, XHTML, WML o incluso PDF.
- XPath es la especificación que desarrolla el lenguaje para acceder a los elementos de un documento XML. Ha sido desarrollada para ser utilizada desde la especificación XSLT y XPointer.
- XSLT, XSL, XSL-FO y XPath están definidos en XML
- XSLT, XSL, XPath y XSL-FO son recomendaciones del W3C.

XSLT comprende entre sus posibilidades:

- Formatear los elementos fuente basados en relaciones de ancestro/descendiente, posición y unicidad.
- La creación de construcciones de formato sofisticadas incluyendo texto generado e imágenes.
- La definición de macros de formateo reutilizables.
- Estilos independientes de la dirección en que se escriba el lenguaje.
- Conjunto de objetos de formato extensible.

Actualmente XSLT es muy usado en la edición web, generando páginas HTML o XHTML. La unión de XML y XSLT permite separar contenido y presentación, aumentando así la productividad.

1.7 Frameworks

Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Los frameworks constituyen un mecanismo para la reutilización y el desarrollo rápido de aplicaciones, y el empleo de ellos se impone en el desarrollo de software actual.

1.7.1 Frameworks del lado cliente

1.7.1.1 Ext JS

Ext JS es un completo framework JavaScript para el desarrollo de aplicaciones ricas de Internet. Las admirables aplicaciones RIA (Rich Internet Application) combinan las ventajas que ofrecen las aplicaciones Web y las aplicaciones tradicionales.

Entre las características principales de Ext JS se incluyen las siguientes:

- Gran desempeño, componentes de interfaz de usuario personalizables.

- Modelo de componentes bien diseñado y extensible.
- Documentación intuitiva y fácil de usar.
- Licencias comerciales y de código abierto disponibles.

Esta biblioteca proporciona abstracciones para manipulación del DOM, AJAX, eventos y eventos personalizados, animaciones, plantillas y mecanismos de programación orientada a objetos.

Viene con una gran cantidad de componentes de interfaz con mayor frecuencia en aplicaciones de escritorio. Hay barras de herramientas, ventanas, pestañas y botones, así como formas sofisticadas y redes de datos.

Ext JS soporta los navegadores web más importantes, incluyendo (Ext, 2006-2010):

- Internet Explorer 6+.
- FireFox 1.5+ (PC, Mac).
- Safari 3+.
- Opera 9+ (PC, Mac).

1.7.1.2 Open-jACOB Draw2D

Open-jACOB Draw2D es una librería JavaScript libre que permite crear gráficos y diagramas. Su interfaz permite dibujarlos directamente desde el navegador, sin necesidad de instalar ningún software adicional.

Open-jACOB Draw2D es el componente de dibujo de Open-jACOB un editor de Workflow al puro estilo de Visio con la finalidad de probar que realmente puede hacerse totalmente online y sin necesidad de instalar extensiones. (Herz, 2010)

Esta librería ofrece un área de edición muy útil para el modelado, acompañada de efectos comúnmente utilizados durante el diseño como son: el redimensionamiento, la alineación y la funcionalidad interactiva de arrastrar y soltar.

1.7.2 Frameworks del lado servidor

1.7.2.1 Symfony

Symfony es un framework para desarrollar aplicaciones web que fue programado con PHP 5 y está enfocado al desarrollo en el mismo lenguaje de programación. Añade una nueva

capa por encima de PHP y proporciona herramientas que simplifican el desarrollo de las aplicaciones web complejas.

Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Es compatible con la mayoría de los gestores de bases de datos, entre ellos MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix y Linux) como en plataformas Windows.

Symfony se ajusta a los siguientes requisitos (Potencier, y otros, 2008):

- Fácil de instalar y configurar en la mayoría de plataformas.
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Basado en la premisa de "convenir en vez de configurar", en la que el desarrollador solo debe configurar aquello que no es convencional.
- Sigue la mayoría de mejores prácticas y patrones de diseño para la web.
- Preparado para aplicaciones empresariales, y adaptable a las políticas y arquitecturas propias de cada empresa, además de ser lo suficientemente estable como para desarrollar aplicaciones a largo plazo.
- Código fácil de leer que incluye comentarios de phpDocumentor y que permite un mantenimiento muy sencillo.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

1.7.2.2 Propel

Propel es un ORM (Object-Relational Mapping) para PHP que facilita la labor de desarrollo de aplicaciones web, gracias a la capa que transforma el tratamiento de la base de datos mediante objetos, con la que se puede recuperar, insertar y modificar datos.

Permite realizar consultas complejas y manipular bases de datos sin escribir una sola cláusula SQL (Structured Query Language). Hace más fácil la escritura de aplicaciones, más fácil de desplegar, y mucho más fácil para migrar si alguna vez la situación lo amerita. Con Propel tan solo es necesario definir la base de datos en formato XML u obtener la definición desde una base de datos ya existente.

Está basado en acercamientos probados, desarrollado por el proyecto Torque y optimizado para PHP, Propel proporciona un inteligente y comprensivo servicio de manejo de datos con un mínimo costo de realización para una aplicación en PHP. (Propel project, 2004)

1.7.2.3 PHPReports

La biblioteca software libre PHPReports es un motor de generación de reportes para PHP. Su funcionamiento está basado en transformaciones XSL (siglas de Extensible Stylesheet Language) conocidas como XSLT. Un reporte en PHPReports está dividido en capas que contienen unas a otras.

En general, contiene un encabezado, un pie y está formado por páginas, las que a su vez cuentan con un encabezado, un pie de página y pueden contener grupos anidados en los que se cargan los campos obtenidos de la consulta a la base de datos.

Dicha biblioteca ha sido modificada en la Universidad de las Ciencias Informáticas por un equipo de trabajo del Centro de Tecnologías de Gestión de Datos, agregándole las siguientes funcionalidades:

- Nuevos formatos de salida EXCEL, PDF.
- Modificaciones a la salida de HTML, permitiendo mayor personalización por el usuario mediante el uso del elemento DIV.
- Mejora al soporte de orígenes de datos en SQLite.
- Paginación de reportes.
- Incorporación de funciones de cálculo.

1.8 Entorno integrado de desarrollo

1.8.1 NetBeans 6.8

NetBeans IDE es un entorno integrado de desarrollo código abierto y gratuito para desarrolladores de software. Ofrece todas las herramientas necesarias para crear aplicaciones de sobremesa profesionales, empresariales, web y móviles con el lenguaje Java, JavaFX, C/C ++ y lenguajes dinámicos como PHP, JavaScript, Groovy y Ruby. NetBeans IDE es fácil de instalar y listo para usar y se puede ejecutar tanto en Windows, Linux, Mac OS X como en Solaris. (Sun Microsystems, Inc., 1994-2010)

Dicho entorno integrado de desarrollo se beneficia de las siguientes características:

- **Formateo de código**: muy importante para tener un orden en la programación, y

entender más fácil el código.

- **Soporta JavaScript, HTML, CSS, PHP**: muestra cualquier error que tenga el código, con lo cual es posible corregirlo, antes de ejecutar el proyecto, por lo que se ahorra tiempo.
- **Documentación**: mientras se escribe alguna función, aparecerá abajo una lista de funciones, esto es útil cuando se olvida el nombre de la función, o el tipo de parámetros que recibe, o el tipo de valor que retorna la función. Disponible en lenguajes como HTML, JavaScript, CSS y PHP.

1.9 Estándar de codificación

En general, un estándar de codificación son reglas que se siguen para la escritura del código fuente. El mismo, puede venir definido a nivel de empresa, a nivel de proyecto o incluso a nivel del propio cliente.

La aplicación de un estándar de codificación aporta características siempre deseadas durante la implementación del software:

- Eleva la mantenibilidad del código.
- Sirve como punto de referencia para los programadores.
- Mantiene un estilo de programación.
- Ayuda a mejorar el proceso de codificación, haciéndolo más eficiente.

En particular, el Centro de Tecnologías de Gestión de Datos define un estándar de codificación a nivel de centro, del cual se señalan algunos aspectos:

- Los nombres de funciones se escribirán usando la notación lowerCamelCase.
- Las constantes se escribirán con todas las letras en mayúsculas.
- A las funciones que obtienen o fijan datos se les colocarán los prefijos get y set respectivamente.
- Los identificadores serán escritos de forma tal que al leerlos sea fácil reconocer el propósito de los mismos.

1.10 Patrones de diseño

Actualmente la sociedad requiere sistemas más complejos y más grandes. Por otro lado, los recursos desarrollados son cada vez más escasos y por tanto se hacen imprescindibles los mecanismos de reutilización. Ante la necesidad de asimilar la reutilización en el desarrollo de software se han popularizado mecanismos como: componentes, frameworks, objetos distribuidos y patrones de diseño.

Un patrón de diseño describe una estructura recurrente de componentes que se comunican para resolver un problema general de diseño en un contexto particular. Nomina, abstrae e identifica los aspectos clave de una estructura de diseño común, lo que los hace útiles para crear un diseño orientado a objetos reutilizable. Identifica las clases e instancias participantes, sus roles y colaboraciones y la distribución de responsabilidades.

En esencia, presentan los siguientes elementos:

- **Nombre**: Permite describir, en una o dos palabras, un problema de diseño junto con sus soluciones y consecuencias.
- **Problema**: Describe cuándo aplicar el patrón. Explica el problema y su contexto. A veces incluye condiciones que deben darse para que tenga sentido la aplicación del patrón.
- **Solución**: Describe los elementos que constituyen el diseño, sus relaciones, responsabilidades y colaboraciones. La solución no describe un diseño o implementación en concreto, sino que es más bien una plantilla que puede aplicarse en muchas situaciones diferentes.
- **Consecuencias**: son los resultados, así como ventajas e inconvenientes de aplicar el patrón.

Los patrones de diseño son clasificados según su finalidad en creacionales, estructurales y de comportamiento, mientras que respecto a su ámbito se especifican en clases y objetos.

- **Creacionales**: Resuelven problemas relativos a la creación de objetos.
- **Estructurales**: Resuelven problemas relativos a la composición de objetos.
- **De Comportamiento**: Resuelven problemas relativos a la interacción entre objetos.
- **Clases**: Relaciones estáticas entre clases.
- **Objetos**: Relaciones dinámicas entre objetos.

A continuación se comentan algunos de los patrones, que más se implementan en aplicaciones web.

- **Singleton (Instancia única):** Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia.
- **Observer (Observador)**: Define una dependencia entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él.
- **Command (Orden):** Especifica una forma simple de separar la ejecución de un

comando del entorno que generó dicho comando. Encapsula una operación en un objeto, permitiendo ejecutar dicha operación sin necesidad de conocer el contenido de la misma.

- **Front Controller (Controlador frontal):** Es un patrón muy utilizado por las aplicaciones web. Describe básicamente como construir un sólo punto de acceso para todas las peticiones de una aplicación web. Escucha las peticiones que vienen desde una URL, y se encarga de llamar al controlador específico, posteriormente llama a la acción deseada del controlador.

Los patrones de diseño presentan las siguientes ventajas:

- Facilitan la localización de los objetos que formarán el sistema.
- Facilitan la determinación de la granularidad adecuada.
- Especifican interfaces para las clases.
- Especifican implementaciones (al menos parciales).
- Facilitan el aprendizaje y la comunicación entre programadores y diseñadores.
- Mejoran la calidad del diseño y la implementación.
- Se pueden considerar como “normas de productividad”.

1.11 Selección de la metodología, lenguajes y herramientas para desarrollar la aplicación

Bajo el principio de desarrollar utilizando tecnologías novedosas, libres y de fácil integración se arriba a las siguientes decisiones.

Como el proyecto cuenta con un pequeño equipo de desarrollo y poco tiempo de desarrollo disponible, se selecciona la metodología OpenUP como guía para la documentación del Diseñador de Reportes. De esta forma se evita la elaboración de artefactos de ingeniería innecesarios y se hace un mayor énfasis en la colaboración entre desarrolladores y clientes. Dicha actividad con la aplicación de la herramienta de gestión de proyectos de software RedMine, queda formalizada y se va madurando cada vez más.

Como lenguaje de programación del lado del servidor se prefiere PHP 5 por ser un lenguaje de código abierto, multiplataforma, soportado por los principales servidores web que existen y beneficiado de un sin número de extensiones validadas por su amplia comunidad. Esta versión además está enriquecida de características de Programación Orientada a Objetos, el paradigma de programación más utilizado en el mundo.

Para programar con PHP 5 de forma rápida, se aprovechará el completo framework para el desarrollo de aplicaciones web Symfony, que implementa el estilo arquitectónico MVC (Modelo – Vista – Controlador) el cual se ha vuelto prácticamente un estándar en las aplicaciones web de hoy en día. Dicho framework trae integrado el ORM Propel, el cual asegura el modelo en la arquitectura señalada y soporta los gestores de base de datos más utilizados en la actualidad.

Se utilizará la librería PHPReport mejorada como motor de generación de reportes. Como es una tecnología libre se le pueden hacer cambios al código fuente según el negocio. Además con los cambios realizados por desarrolladores del Centro de Tecnologías de Gestión de Datos, esta biblioteca provee un conjunto de funcionalidades nuevas que son de gran importancia y enriquecen la generación de reportes dinámicos.

Del lado cliente se implementará con el potente lenguaje de programación JavaScript, quien tuvo como catalizador de su popularidad a la tecnología AJAX que en estos momentos se impone en el desarrollo Web. Con el objetivo de aprovechar mecanismos de programación orientada a objetos y reutilizar componentes de interfaz comunes en aplicaciones de escritorio se extenderá el framework EXT JS. A su vez, para conseguir características de edición avanzadas se opta por la librería JavaScript libre Open Jacob Draw2d, restándole así toda la complejidad al proceso de diseño que se desea automatizar.

Como formato para la definición del reporte y para el intercambio con los demás módulos del sistema generador de reportes, se acudirá al estándar XML que acompañado de la tecnología XSLT provee la flexibilidad y escalabilidad deseadas para el sistema. En último lugar, para escribir código de manera organizada y fácil se empleará el entorno integrado de desarrollo Netbeans 6.8 con soporte para los lenguajes de programación a utilizar y se cumplirá el estándar de codificación definido por DATEC.

1.12 Conclusiones del capítulo

A partir de los aspectos abordados durante este capítulo se puede concluir lo siguiente:

- Se realizó el estudio del arte de las herramientas para el diseño de reportes en el mundo, analizando las ventajas y desventajas de cada una de ellas.
- Se seleccionaron los lenguajes, frameworks, entorno de desarrollo y metodología a utilizar para la implementación del módulo Diseñador de Reportes.

CAPÍTULO 2. DESCRIPCIÓN DE LA SOLUCIÓN PROPUESTA

2.1 Introducción

En este capítulo se describe el sistema Generador Dinámico de Reportes, enfocando la atención en el módulo Diseñador de Reportes y su integración con otros componentes. Se especifica la estructura física de la solución que se propone mediante el modelo de despliegue y la vista de implementación. Para un mejor entendimiento se describen las clases más importantes y los principales componentes existentes que reutiliza el Diseñador de Reportes. Además se explica la salida principal del módulo.

2.2 Diseño general del sistema Generador Dinámico de Reportes

El sistema Generador Dinámico de Reportes 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, puesto que aumentan la reutilización y reducen la redundancia de información (Ver Anexo 1).

2.2.1 Descripción general del módulo Diseñador de Reportes

La aplicación es la encargada de diseñar reportes personalizados y almacenarlos en el Servidor. Cuenta con las funciones necesarias para diseñar los reportes de manera interactiva y aprovecha las ventajas de la web. Los reportes son basados en un fichero XML que contiene la definición del reporte. Dicha definición es el núcleo del sistema ya que comunica las diferentes aplicaciones y facilita que las mismas se integren conformando finalmente el reporte final.

Este módulo se considera una parte importante del sistema Generador Dinámico de Reportes y se integra con los siguientes módulos: Diseñador de Modelos y Visor de Reportes.

2.2.2 Integración con el módulo Diseñador de Modelos

Este módulo provee modelos semánticos de base de datos, que facilitan el manejo de la fuente de datos en el Diseñador de Reportes. La principal ventaja de su empleo es que permite reducir el alcance de los datos a reportear.

Los modelos diseñados con este componente pueden ser reutilizados en múltiples reportes y constituyen la única entrada necesaria para el Diseñador de Reportes. Dichos modelos contienen los datos necesarios para conectarse a la fuente de datos y una descripción detallada de los elementos de base de datos que los conforman. Los últimos, generalmente son un subconjunto de la estructura de la base de datos. Puesto que los modelos son almacenados en formato XML se facilita el procesamiento de los mismos y por consiguiente la integración del módulo con el Diseñador de Reportes.

2.2.3 Integración con el módulo Visor de Reportes

Este componente permite visualizar los reportes creados con el Diseñador de Reportes, facilitando además la exportación de los mismos a varios formatos: HTML, PDF, EXCEL y CSV. La unidad básica y necesaria para utilizar dicho componente son los reportes diseñados. Como la definición de los reportes se guarda en formato XML se facilita la visualización de los mismos y por ende la integración del módulo con el Diseñador de Reportes.

2.3 Implementación

En la implementación a partir de los resultados del análisis y diseño se implementa el sistema en términos de componentes, es decir, ficheros de código fuente, scripts, ejecutables y similares. El objetivo fundamental es representar las partes físicas que componen el sistema, organizado de una manera que sea comprensible y manejable.

2.3.1 Modelo de despliegue

El diagrama de despliegue modela la estructura física del sistema, centrándose en la configuración de los nodos de procesamiento en tiempo de ejecución y los componentes de software.

La siguiente figura muestra el diagrama de despliegue del Módulo Diseñador de Reportes del sistema Generador Dinámico de Reportes:

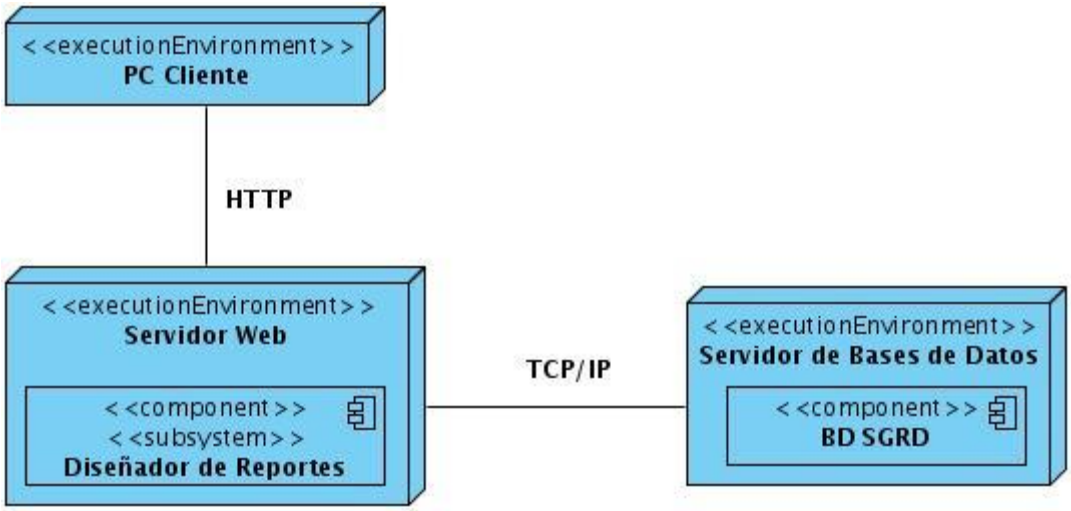


Figura 1. Diagrama de la Vista de Despliegue

En este caso se necesita un servidor web como entorno de ejecución con Apache2, soporte para PHP 5 y sistema operativo basado en GNU/Linux en cualquiera de sus distribuciones, preferentemente Debian o Ubuntu. La base de datos puede estar instalada en el mismo servidor o en uno distinto en dependencia de las posibilidades de la organización, siendo necesario PostgreSQL Server, versión 8.3 o superior. Los usuarios podrán conectarse a la aplicación desde estaciones clientes e incluso desde el mismo servidor a través del navegador web Mozilla Firefox o cualquier otro que utilice el motor Gecko como es el caso de Epiphany Web Browser del escritorio Gnome.

- **Servidor Web**: En este nodo se encuentra la aplicación Diseñador de Reportes.
- **Servidor de Base de Datos**: Es donde se encuentra almacenada la base de datos del Sistema Generador Dinámico de Reportes.
- **PC Cliente**: Desde este nodo se accede al sistema por medio de un navegador.

2.3.2 Vista de implementación

Los diagramas de componentes modelan las organizaciones y dependencias lógicas entre componentes de software. Se considera necesario satisfacer durante su realización los requisitos relacionados con la facilidad de desarrollo, la reutilización, así como las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo.

En la siguiente figura se muestran de forma global los diferentes subsistemas de implementación en los que se divide la aplicación, siendo posible observar el estilo arquitectónico que se implementa.

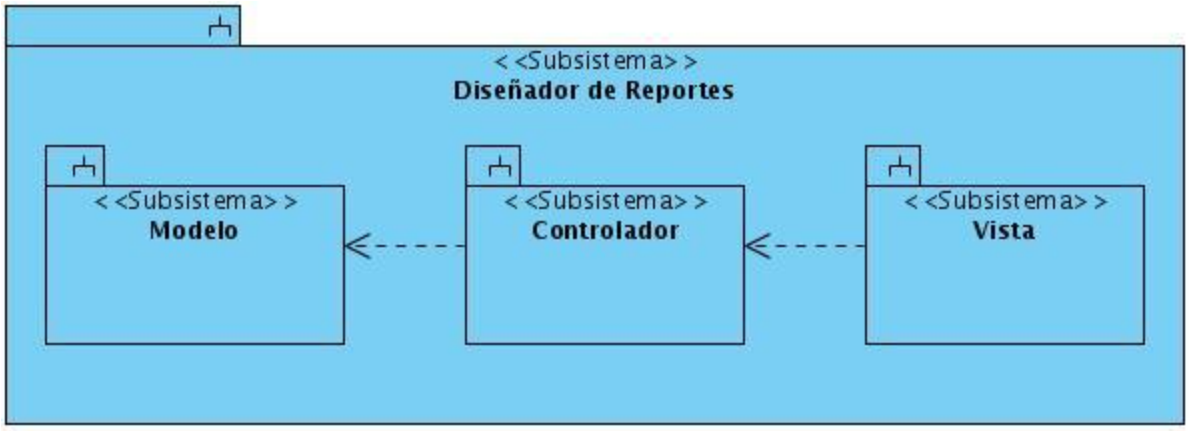


Figura 2. Diagrama de componentes global del Diseñador de Reportes

A continuación se muestra el subsistema de implementación Modelo, el cual está conformado por cinco componentes y otro subsistema de implementación.

Los componentes **DataSource.php**, **Model.php**, **Report.php** y **Template.php** contienen las clases encargadas del tratamiento como objetos de las tablas de la base de datos del sistema que almacenan los orígenes de datos, los modelos, los reportes y las plantillas de reportes respectivamente.

phpreport: subsistema de implementación que contiene las clases encargadas de la generación del reporte a partir de su definición. Recibe el documento XML que representa la definición del reporte y lo transforma mediante XSLT a HTML u otros formatos de salida como el formato estándar PDF.

Reporte.xsl: es el componente que gracias a la tecnología XSLT, permite transformar el diseño preliminar del reporte (XML) en una definición formal de reporte (XML) que está concebida para ser manejada por el motor de reportes **phpreport**.

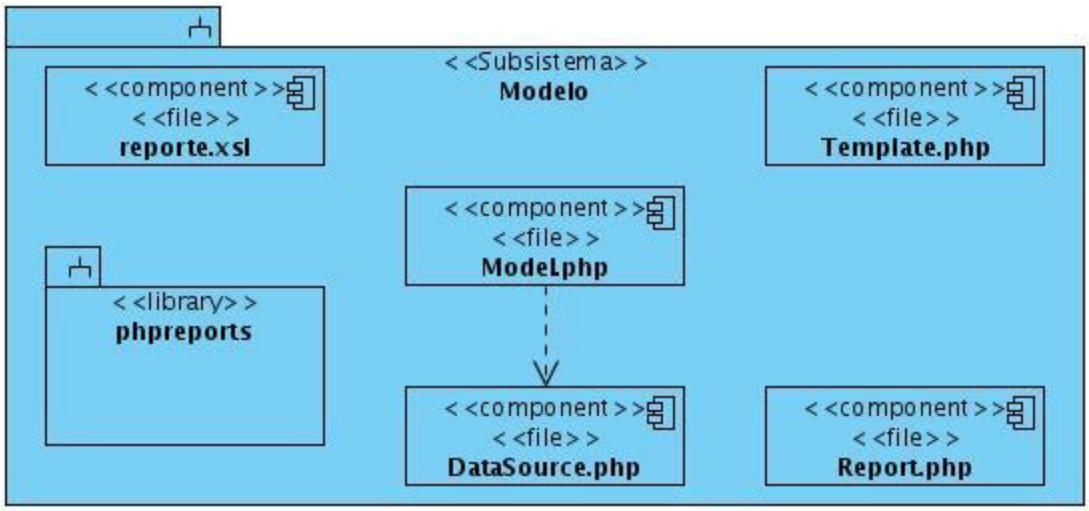


Figura 3. Diagrama de componentes del subsistema Modelo

Esta figura muestra la relación entre los componentes del subsistema de implementación Vista. Particularmente el componente **indexReportDesignerSuccess.php** depende del subsistema **JS** y es definido mediante el framework Symfony como controlador frontal para la aplicación, es decir el único punto de acceso a la misma, resultante de implementar el patrón de diseño Controlador Frontal.

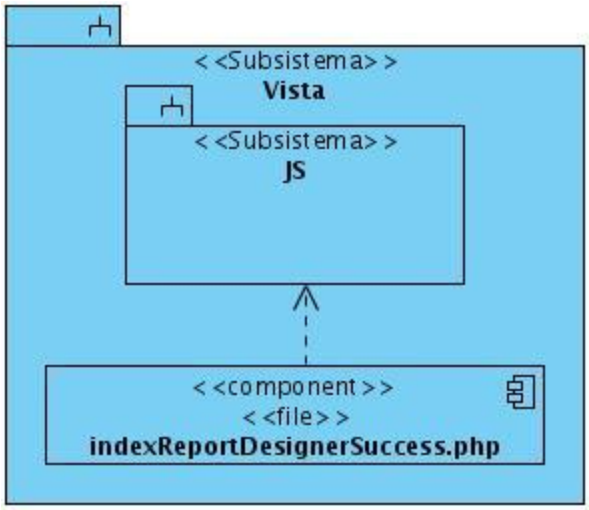


Figura 4. Diagrama de componentes del subsistema Vista

La siguiente figura muestra los componentes del subsistema de implementación Controlador cuyas clases son denominadas **Acciones** en el framework Symfony. Estas a su vez constituyen una aplicación concreta del patrón de diseño Command (Orden).

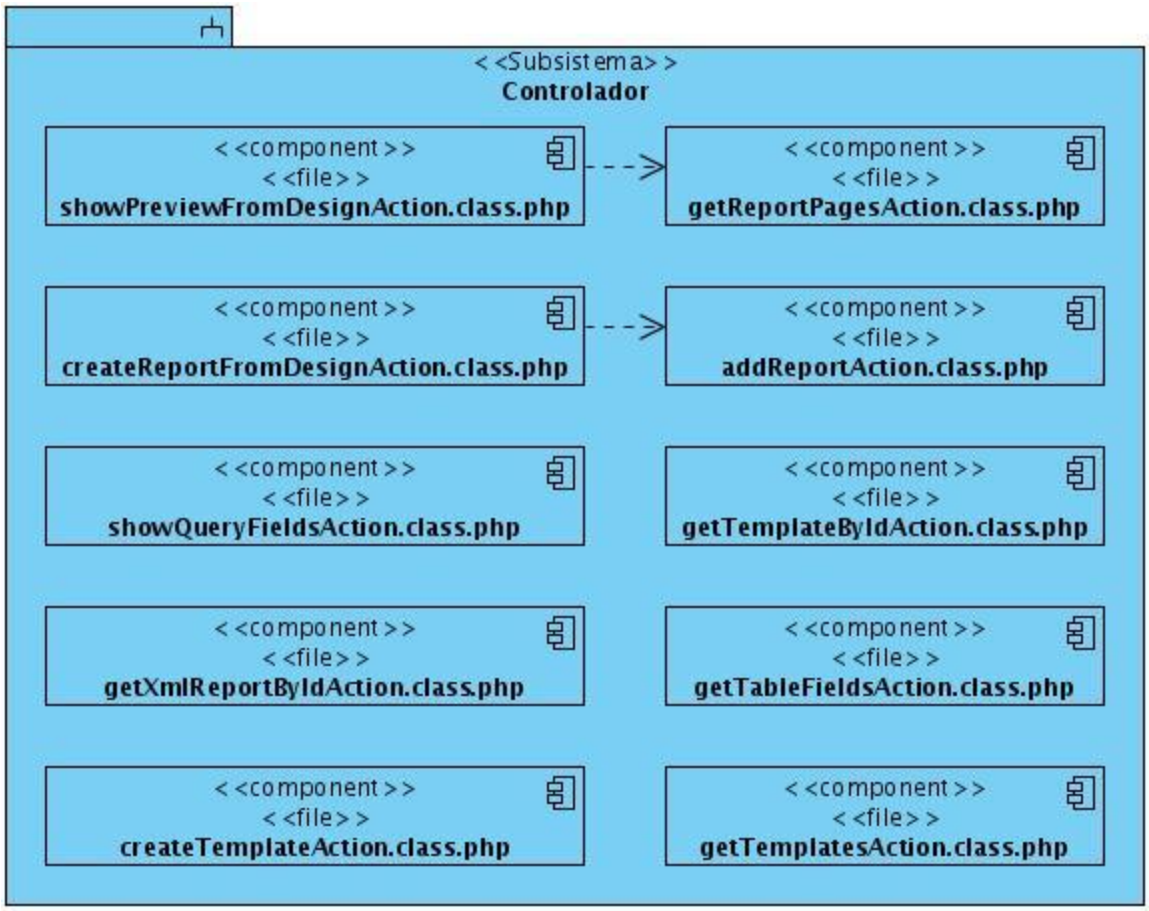


Figura 5. Diagrama de componentes del subsistema Controlador

Los diagramas de componentes referentes a los subsistemas de implementación contenidos en el subsistema Vista son especificados en el Anexo 2.

2.4 Componentes existentes que son reutilizados en la Presentación

La reutilización de componentes en la Presentación, desde el punto de vista del programador aumenta la productividad y la erradicación en lo posible de errores. Comprender este concepto se considera clave para alcanzar en poco tiempo de desarrollo, interfaces de usuario altamente interactivas y agradables.

La siguiente figura muestra la combinación de componentes personalizables de EXT JS y Draw2d, alcanzando una interfaz gráfica agradable y altamente interactiva. En general los componentes de interfaz implementados aplican el patrón de diseño Observer (Observador) facilitado por las librerías mencionadas para realizar un correcto manejo de los eventos disparados por el usuario.

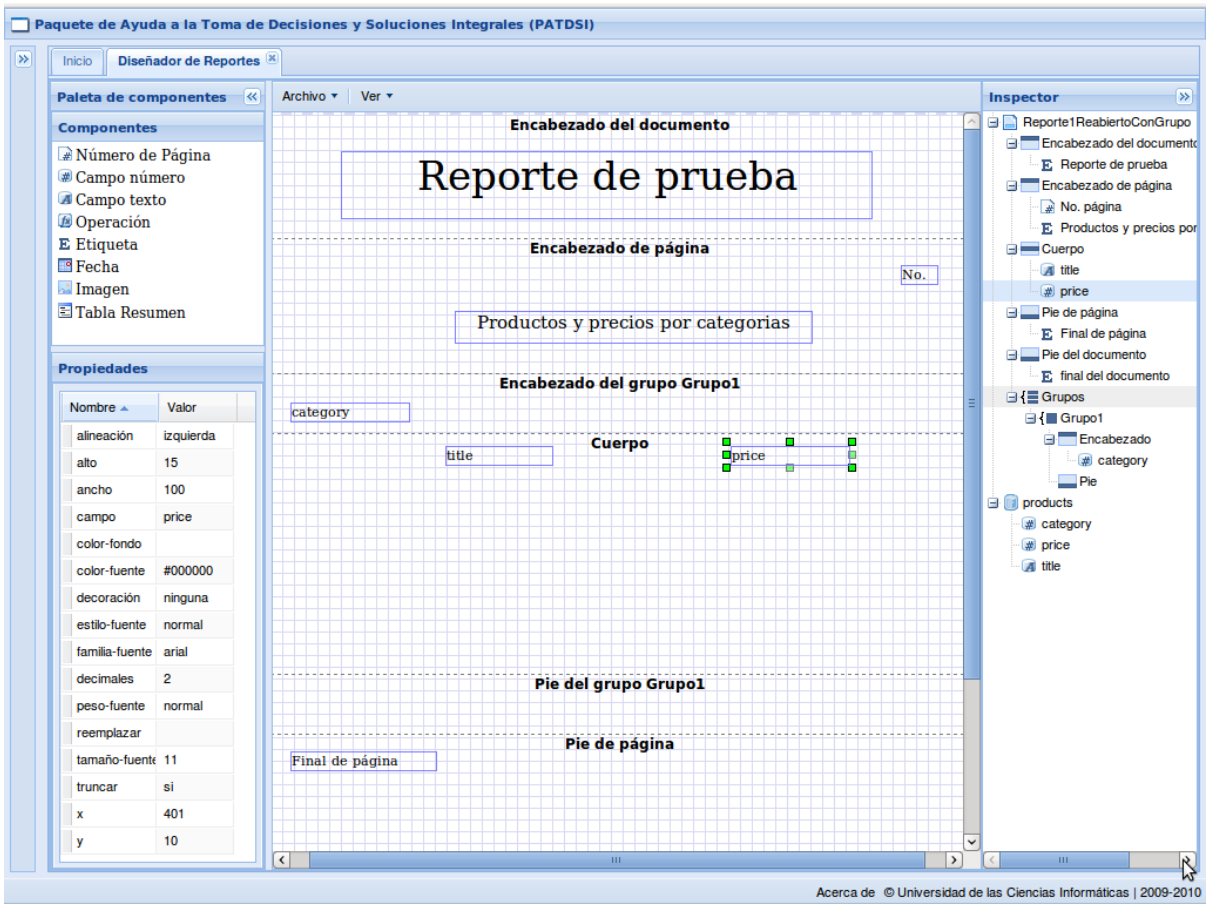


Figura 6. Uso de componentes de EXT JS y Draw2D

2.4.1 Componentes de EXT JS

El framework EXT JS permitió la reutilización de componentes que son usados comúnmente en el desarrollo de interfaces gráficas de usuario. En particular la solución propuesta aprovecha entre otros los siguientes componentes de interfaz personalizables:

- **TreePanel**: Útil para facilitar la navegación sobre los elementos que conforman el reporte, ya que son representados en forma de jerarquía. Se utilizó en el Inspector permitiendo comprender y explorar la estructura del reporte.
- **Menu**: Se emplea para brindar las opciones más importantes del módulo Diseñador de Reportes (Salvar, Nuevo, Abrir y Vista previa).
- **Panel**: Se considera el contenedor universal para organizar y agrupar componentes. Un ejemplo concreto de ello es la Paleta de Componentes que contiene los elementos de reporte disponibles.
- **PropertyGrid**: Facilita la reconfiguración de los componentes del reporte, ya

que representa de una forma sencilla las relaciones propiedad-valor correspondientes al elemento de reporte seleccionado. Se empleó en la sección de Propiedades al estilo de los entornos de desarrollo más conocidos.

2.4.2 Componentes de Draw2D

La librería Draw2D permitió alcanzar opciones de edición avanzadas que son frecuentemente encontradas en herramientas de diseño o modelado. El Diseñador de Reportes se beneficia fundamentalmente de los siguientes componentes:

- **Workflow**: Se trata de un área de dibujo altamente interactiva al estilo de Visio que facilita la edición de modelos bien conformados, puesto que garantiza una libre colocación de los objetos y la alineación entre ellos. Su utilización quedó reflejada en el área central del Diseñador de Reportes, aumentando notablemente la interacción con los componentes de reporte.
- **Node**: Representa características básicas de los objetos de dibujo tales como las dimensiones y la posición. Permite realizar el redimensionamiento y facilita el dibujo de los componentes en el Workflow. Fue extendido por cada uno de los componentes de reporte, siendo posible dibujarlos en el área antes indicada.

2.5 Descripción de las principales clases

El paradigma orientado a objetos enfatiza la creación de clases que encapsulan tanto los datos como los algoritmos que se utilizan para manejar los datos. Si se diseñan y se implementan adecuadamente, las clases orientadas a objetos son reutilizables por las diferentes aplicaciones y arquitecturas de sistemas basados en computadora. (S. Pressman, 2005)

2.5.1 Acciones

La capa controlador, contiene el código que asocia la lógica de negocio con la presentación. El sistema Generador de Reportes se beneficia de un controlador frontal que constituye el único punto de entrada a la aplicación y determina la acción a ejecutarse ante cada solicitud. Las acciones definidas contienen la lógica de la aplicación Diseñador de Reportes. De manera general verifican la integridad de las peticiones y preparan los datos necesitados por la capa de presentación.

Tabla 1 Descripción de la clase: createReportFromDesignAction

Nombre: createReportFromDesignAction	
Tipo de clase: controladora	
Atributo	**Tipo**
-	**-**
Responsabilidad:	
Nombre:	execute
Descripción:	Función que construye la definición del reporte en formato XML, a partir de su diseño y la envía a la acción "addReportAction".

Tabla 2 Descripción de la clase: addReportAction

Nombre: addReportAction	
Tipo de clase: controladora	
Atributo	**Tipo**
-	**-**
Responsabilidad:	
Nombre:	execute
Descripción:	Función que guarda la definición de un reporte diseñado.

Tabla 3 Descripción de la clase: getXmlReportByIdAction

Nombre: getXmlReportByIdAction	
Tipo de clase: controladora	
Atributo	**Tipo**

-	-
Responsabilidad:	
Nombre:	execute
Descripción:	Función que devuelve la definición de un reporte en formato XML dado su id.

Tabla 4 Descripción de la clase: getTableFieldsAction

Nombre: getTableFieldsAction	
Tipo de clase: controladora	
Atributo	**Tipo**
-	-
Responsabilidad:	
Nombre:	execute
Descripción	Función que devuelve los campos de una tabla en formato JSON, dado su nombre y el id del modelo de datos en el que se encuentra referenciada.

Tabla 5 Descripción de la clase: showPreviewFromDesignAction

Nombre: showPreviewFromDesignAction	
Tipo de clase: controladora	
Atributo	**Tipo**
-	-
Responsabilidad:	
Nombre:	execute
Descripción	Función que utiliza la librería phpreports para generar el reporte con los datos en formato XML y lo envía a la acción “getReportPagesAction”.

Tabla 6 Descripción de la clase: getReportPagesAction

Nombre: getReportPagesAction
Tipo de clase: controladora

Atributo	**Tipo**
-	-
Responsabilidad:	
Nombre:	execute
Descripción	Función que devuelve el reporte listo para ser visualizado en el navegador. Utiliza la librería phpreports para obtener el reporte con los datos en formato HTML y con una cantidad de páginas determinada.

2.5.2 Clases del Modelo

A partir de que PHP 5 y Symfony están orientados a objetos, se vuelve necesario acceder de forma efectiva a la base de datos desde un contexto orientado a objetos. Dicho requisito es facilitado por el ORM Propel, el cual crea el modelo de objetos de datos que utiliza Symfony.

Las clases del modelo fueron generadas de una forma sencilla a partir de un esquema relacional de la base de datos en formato YAML. Entre las clases correspondientes a la capa modelo del sistema Generador de Reportes, son imprescindibles para el módulo Diseñador de Reportes las siguientes: Report, Template y Model.

Tabla 7 Descripción de la clase: Report

Nombre: Report	
Tipo de clase: modelo	
Atributo	**Tipo**
-	-
Responsabilidad:	
Nombre:	getById
Descripción:	Función que devuelve un reporte dado su id.
Nombre:	getDataSource
Descripción:	Función que devuelve el origen de datos asociado al reporte.
Nombre:	salvar
Descripción:	Función que guarda el reporte.
Nombre:	getFields
Descripción:	Función que devuelve los campos de la consulta asociada al reporte.

Nombre:	getAllFields
Descripción:	Función que devuelve todos los campos asociados al reporte, incluyendo los que no pertenecen a la consulta asociada al mismo.
Nombre:	getReportByName
Descripción:	Función que devuelve un reporte dado su nombre.
Nombre:	getAll
Descripción:	Función que devuelve todos los reportes pertenecientes a una determinada categoría dado el id de la misma.
Nombre:	CanBeDeleted
Descripción:	Función que verifica si el reporte se puede eliminar.
Nombre:	setParameters
Descripción:	Función que modifica la consulta asociada al reporte a partir de nuevos parámetros.
Nombre:	getParameters
Descripción:	Función que devuelve los parámetros de la consulta asociada al reporte.
Nombre:	setParametersValues
Descripción:	Función que modifica la consulta asociada al reporte a partir de nuevos valores para sus parámetros.
Nombre:	getByIdCatAndName
Descripción:	Función que devuelve un reporte dados su nombre y el id de la categoría a la que está asociado.
Nombre:	ExistName
Descripción:	Función que verifica si ya un reporte existe dado su nombre.

2.5.3 Clases de Interfaz

Aunque el lenguaje JavaScript no se considera propiamente orientado a objetos, sí permite desarrollar algunos mecanismos de orientación a objetos. De forma general es posible encapsular comportamientos y reutilizar los objetos que representan a las clases.

Tabla 8 Descripción de la clase: TLabel

Nombre: TLabel	
Tipo de clase: interfaz	
Atributo	**Tipo**
Label	string
fontFamily	string
fontStyle	string
fontWeight	string
Decoration	string
Overflow	string
fontSize	int
Align	string
Color	Color
Responsabilidad:	
Nombre:	TLabel
Descripción	Constructor del objeto TLabel que extiende al objeto Node de Open-jACOB Draw2D.
Nombre:	createHTMLElement
Descripción	Función que construye el elemento HTML correspondiente a la Etiqueta con valores predeterminados para cada uno de los atributos.
Nombre:	isResizeable
Descripción	Función que indica si la Etiqueta se puede redimensionar.
Nombre:	getProperty
Descripción	Función que devuelve el valor dado el nombre de la propiedad.
Nombre:	setValueByProperty
Descripción	Función que cambia el valor de una propiedad dado su nombre y el nuevo valor.
Nombre:	getProperties
Descripción	Función que devuelve la configuración de la Etiqueta, es decir sus relaciones propiedad-valor.
Nombre:	getXML

Descripción	Función que devuelve la Etiqueta en formato XML.
Nombre:	clone
Descripción	Función que devuelve una copia de la Etiqueta.

Tabla 9 Descripción de la clase: Text

Nombre: Text	
Tipo de clase: interfaz	
Atributo	**Tipo**
label	string
campo	string
field	FieldTD
Responsabilidad:	
Nombre:	Text
Descripción	Constructor del objeto Text que extiende a TLabel.
Nombre:	createHTMLElement
Descripción	Función que construye el elemento HTML correspondiente al campo Texto con valores predeterminados para cada uno de los atributos.
Nombre:	isResizeable
Descripción	Función que indica si el campo Texto se puede redimensionar.
Nombre:	getProperty
Descripción	Función que devuelve el valor dado el nombre de la propiedad.
Nombre:	setValueByProperty
Descripción	Función que cambia el valor de una propiedad dado su nombre y el nuevo valor.
Nombre:	getProperties
Descripción	Función que devuelve la configuración del campo Texto, es decir sus relaciones propiedad-valor.
Nombre:	getXML
Descripción	Función que devuelve el campo Texto en formato XML.

Nombre:	clone
Descripción	Función que devuelve una copia del campo Texto.

Tabla 10 Descripción de la clase: FieldTD

Nombre: FieldTD	
Tipo de clase: interfaz	
Atributo	**Tipo**
field	string
type	string
alias	string
idModel	int
entityType	string
entityName	string
schema	string
Responsabilidad:	
Nombre:	FieldTD
Descripción	Constructor del objeto FieldTD.
Nombre:	setValueByProperty
Descripción	Función que cambia el valor de una propiedad dado su nombre y el nuevo valor.
Nombre:	getProperty
Descripción	Función que devuelve el valor dado el nombre de la propiedad.

Tabla 11 Descripción de la clase: WorkflowTD

Nombre: WorkflowTD	
Tipo de clase: interfaz	
Atributo	**Tipo**
areas	ArrayList
height	int
width	int
align	string
title	string
typePage	string
flip	string

Responsabilidad:	
Nombre:	WorkflowTD
Descripción	Constructor del objeto WorkflowTD que extiende al objeto Workflow de Open-jACOB Draw2D. Este objeto representa al reporte en tiempo de diseño.
Nombre:	setValueByProperty
Descripción	Función que cambia el valor de una propiedad dado su nombre y el nuevo valor.
Nombre:	getProperty
Descripción	Función que devuelve el valor dado el nombre de la propiedad.
Nombre:	addArea
Descripción	Función que adiciona una nueva área al reporte.
Nombre:	getArea
Descripción	Función que devuelve el área a la que pertenecen coordenadas dadas dentro del espacio de diseño del reporte.
Nombre:	addFigure
Descripción	Función que utilizando la función getArea permite adicionar una figura al reporte dados la figura y sus coordenadas en el espacio de diseño.
Nombre:	removeFigure
Descripción	Función que elimina una figura del reporte.
Nombre:	removeArea
Descripción	Función que elimina un área del reporte.
Nombre:	getProperties
Descripción	Función que devuelve la configuración del reporte, es decir sus relaciones propiedad-valor.
Nombre:	clearAll
Descripción	Función que elimina las áreas y figuras adicionadas al reporte.
Nombre:	getXMLBegin
Descripción	Función que devuelve un XML que contiene la configuración del reporte.

Tabla 12 Descripción de la clase: Factory

Nombre: Factory	
Tipo de clase: interfaz	
Atributo	**Tipo**
workflow	WorkflowTD
groups	ArrayList
alias	string
idModel	int
entityType	string
entityName	string
schema	string
Responsabilidad:	
Nombre:	Factory
Descripción	Constructor del objeto Factory.
Nombre:	configurePage
Descripción	Función que cambia la configuración de página del reporte dados el tipo de página y la orientación.
Nombre:	newDefaultTemplate
Descripción	Función que define una plantilla de reporte básica con áreas convencionales y una configuración predeterminada.
Nombre:	addComponent
Descripción	Función que adiciona un componente al reporte.
Nombre:	addGroupArea
Descripción	Función que adiciona al reporte las áreas correspondientes a una agrupación.
Nombre:	clear
Descripción	Función que elimina todos los componentes y agrupaciones adicionadas al reporte.
Nombre:	getXML
Descripción	Función que devuelve un XML con la configuración del diseño realizado, incluye componentes, áreas, agrupaciones, configuración de página, la fuente de datos y las relaciones entre ellos.

Nombre:	addGroup
Descripción	Función que adiciona una agrupación al reporte.
Nombre:	removeGroup
Descripción	Función que elimina una agrupación del reporte.
Nombre:	getGroup
Descripción	Función que devuelve una agrupación dado su nombre.
Nombre:	setSource
Descripción	Función que cambia la fuente de datos del reporte.
Nombre:	getFieldsTD
Descripción	Función que devuelve los campos de datos asociados a los componentes presentes en el diseño del reporte.
Nombre:	Load
Descripción	Función que carga el diseño de un reporte existente dado el XML que lo define.
Nombre:	getParameters
Descripción	Función que devuelve los parámetros del reporte en caso de que existan y la fuente de datos sea una función.
Nombre:	setParameters
Descripción	Función que cambia los parámetros del reporte en caso de que la fuente de datos sea una función dados los nuevos parámetros.

Tabla 13 Descripción de la clase: Paleta

Nombre: Paleta	
Tipo de clase: interfaz	
Atributo	**Tipo**
html	string
width	string
split	bool
region	string
collapsible	bool
title	string

height	int
autoScroll	bool
Responsabilidad:	
Nombre:	constructor
Descripción	Constructor del objeto Paleta que extiende a Ext.Panel. Se definen los elementos HTML correspondientes a los componentes de reporte, con la funcionalidad de arrastrar y colocar hacia el espacio de diseño.

Tabla 14 Descripción de la clase: gridProperties

Nombre: gridProperties	
Tipo de clase: interfaz	
Atributo	**Tipo**
frame	bool
width	string
clicksToEdit	int
stripeRows	bool
collapsible	bool
height	int
autoScroll	bool
Responsabilidad:	
Nombre:	constructor
Descripción	Constructor del objeto gridProperties que extiende a Ext.grid.PropertyGrid.
Nombre:	gridPropAfterEdit
Descripción	Función que actualiza el valor de una propiedad después de ser editada. Efectúa la modificación correspondiente en el diseño.

2.6 Salida principal del Diseñador de Reportes

La salida principal del Diseñador de Reportes es el XML que contiene las especificaciones de diseño del reporte. Dicho documento organiza la estructura del reporte de una forma comprensible que facilita su procesamiento. De manera general el reporte se divide en bandas que contienen los componentes del reporte. Estas se

corresponden con el encabezado y pie de cada uno de los siguientes términos: documento, página y agrupación.

La siguiente figura muestra la estructura de la definición de reporte generada a partir del diseño de un reporte con la aplicación implementada.

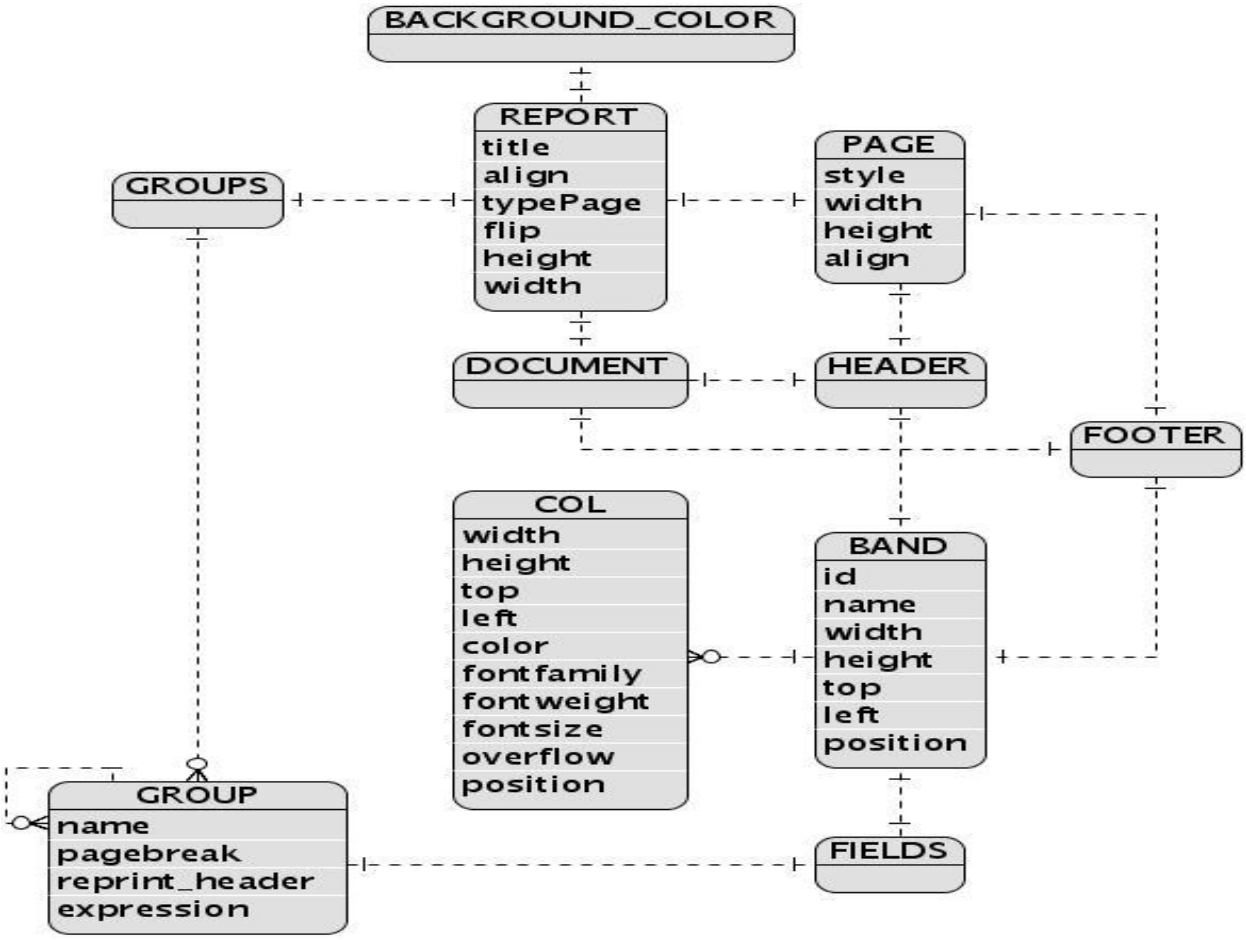


Figura 7. Documento XML con definición del reporte

Como se puede observar cada unidad de diseño está acompañada de propiedades relacionadas al posicionamiento y dimensión del objeto, permitiendo así una mayor correspondencia entre el diseño y la visualización final del reporte.

La mayoría de las propiedades manejadas se acercan bastante a las encontradas en el lenguaje de etiquetas HTML, lo cual confiere una mayor calidad de presentación para el reporte.

Se considera que la flexibilidad lograda con la conformación de este documento, potencia una mayor personalización por los usuarios y una mejor calidad de presentación para el reporte.

2.7 Conclusiones del capítulo

En este capítulo se realizó una descripción de la implementación del módulo Diseñador de Reportes utilizando los artefactos: modelo de despliegue y vista de implementación. Se explicó el modo en que se integra con otros componentes facilitando el funcionamiento del sistema Generador Dinámico de Reportes. Se detallaron las principales clases presentes en la solución así como los componentes más importantes de EXT JS y Draw2D involucrados en la implementación. Para un mayor entendimiento del diseño del reporte se explicó la estructura del documento XML correspondiente, el cual se considera la salida principal del módulo implementado.

CAPÍTULO 3. VALIDACIÓN DE LA SOLUCIÓN PROPUESTA

3.1 Introducción

En este capítulo se introduce el “Modelo en V” como estrategia de prueba que guía el desarrollo de las pruebas del Diseñador de Reportes. Se reflejan las pruebas de sistema, siendo especificado el diseño de algunos casos de prueba ejecutados. Se muestra la realización de pruebas unitarias aplicando el método de pruebas de caja blanca y la herramienta Lime para la automatización de las mismas. Se evalúan las pruebas, siendo posible descubrir los defectos del software y atenderlos de la forma adecuada. Se valoran los resultados obtenidos, permitiendo distinguir el camino evolutivo del módulo Diseñador de Reportes. Además mediante la aplicación de métricas se cuantifica la complejidad y facilidad de mantenimiento de dicho módulo.

3.2 El Modelo V

Las pruebas forman parte del ciclo de vida del software y se deben realizar a lo largo de su desarrollo, siendo de gran significación disponer de una estructura organizativa que establezca las actividades a ejecutar y guíe la realización de este proceso, garantizando que el software sea finalmente aceptado.

Las pruebas del proyecto se realizan en función del Modelo en V. El mismo establece como principio que los procedimientos utilizados para probar si la aplicación cumple las especificaciones ya deben haberse creado en la fase de diseño. Además demuestra las relaciones entre cada fase del ciclo vital del desarrollo y su fase asociada de prueba. Los niveles de prueba abordados por este modelo son los siguientes:

- Prueba de Unidad.
- Prueba de Integración.
- Prueba de Sistema.
- Prueba de Aceptación.

Con el modelo en V, el diseño de las pruebas se realiza en el principio del proyecto antes de la codificación y por ende se ahorra una cantidad notable de tiempo del proyecto.

Figura 8. Modelo V

La figura anterior muestra una estructura que cubre todas las fases en las cuales está dividido el proceso de desarrollo logrando de esta manera una revisión estricta de cada paso que se realice.

3.3 Pruebas de sistema

En particular la prueba del sistema compara las especificaciones de sistema contra el sistema real. El diseño de la misma se deriva del diseño externo de la solución que es resultado del análisis. Una vez que se integren todos los módulos, varios errores pueden presentarse, por tanto resulta sumamente importante la realización correcta de este tipo de prueba.

3.3.1 Pruebas de Caja Negra

Como técnica de prueba se utilizan las pruebas de caja negra que se llevan a cabo sobre la interfaz del software. Su objetivo es demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y se produce un resultado correcto, y que la integridad de la información externa se mantiene. Son diseñadas para validar los requisitos funcionales sin fijarse en el funcionamiento interno del programa. Las mismas se centran en el ámbito de información del programa, de forma que se proporcione una cobertura completa de prueba.

3.3.2 Diseño de casos de prueba de sistema

El objetivo fundamental del diseño de casos de prueba es conseguir un conjunto de pruebas que tengan la mayor probabilidad de encontrar los defectos del software.

Un caso de prueba es un conjunto de entradas de pruebas, condiciones de ejecución y resultados esperados desarrollados para verificar una función esperada. La prueba debe comprobar que el software satisface los requerimientos del usuario y que se comporta de la forma deseada.

En particular, el diseño de casos de prueba del sistema para el módulo Diseñador de Reportes incluye los siguientes escenarios:

1. **Seleccionar campos de una tabla:** Permite seleccionar campos correspondientes a una tabla de base de datos en uno de los modelos existentes.
2. **Seleccionar campos de una función:** Permite seleccionar campos correspondientes a una función de base de datos en uno de los modelos existentes.
3. **Seleccionar campos de una vista:** Permite seleccionar campos correspondientes a una vista de base de datos en uno de los modelos existentes.
4. **Seleccionar campos de una consulta:** Permite seleccionar campos correspondientes a una consulta de base de datos en uno de los modelos existentes.
5. **Adicionar componente Número de página**: Permite adicionar un componente de tipo Número de página al reporte en la posición deseada.
6. **Adicionar componente Campo número**: Permite adicionar un componente de tipo Campo número al reporte en la posición deseada.
7. **Adicionar componente Campo texto**: Permite adicionar un componente de tipo Campo texto al reporte en la posición deseada.
8. **Adicionar componente Operación en área permitida**: Permite adicionar un componente de tipo Operación al reporte en la posición deseada.
9. **Adicionar componente Etiqueta**: Permite adicionar un componente de tipo Etiqueta al reporte en la posición deseada.
10. **Adicionar componente Fecha**: Permite adicionar un componente de tipo Fecha al reporte en la posición deseada.
11. **Mostrar vista previa**: Permite visualizar una vista del reporte diseñado.
12. **Nuevo reporte**: Permite empezar un nuevo diseño de reporte.
13. **Abrir reporte**: Permite cargar el diseño de un reporte existente.

14. **Abrir plantilla**: Permite cargar el diseño de una plantilla de reporte existente.
15. **Salvar el diseño como plantilla**: Permite salvar la plantilla del reporte diseñado.
16. **Salvar el diseño como reporte**: Permite salvar el reporte diseñado.

Los principales casos de uso del sistema para el Diseñador de Reportes son Diseñar reporte y Diseñar plantilla. Estos a su vez referencian varios casos de uso, de los cuales a continuación se presentan algunos casos de prueba.

Diseño del caso de prueba correspondiente al Caso de Uso "Adicionar Componente con datos".

Descripción General:

El caso de uso se encarga de adicionar al área de diseño un componente con datos asociados que provengan de la fuente de datos.

Condiciones de Ejecución:

- El usuario debe estar autenticado.
- Debe seleccionar el vínculo Diseñador de Reportes.

Secciones a probar en el Caso de Uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Adicionar componente con datos.	EC 1.1: Adicionar componente Campo texto.	El sistema permitirá agregar al reporte un componente de tipo Campo texto con datos asociados en la posición que se indique.	1. El usuario arrastra hasta el área de diseño, un nodo de la fuente de datos que represente un campo texto. 2. El usuario mueve el componente adicionado a la posición que desee. 3. El usuario alinea el componente adicionado con otros componentes.
	EC 1.2: Adicionar componente Campo número.	El sistema permitirá agregar al reporte un componente de tipo Campo número con datos asociados en la posición que se indique.	1. El usuario arrastra hasta el área de diseño, un nodo de la fuente de datos que represente un campo número. 2. El usuario mueve el componente adicionado

			a la posición que desee. 3. El usuario alinea el componente adicionado con otros componentes.

SC 1: Adicionar componente con datos

Id del escenario	Escenario	Área contenedora	Respuesta del Sistema	Resultado de la Prueba
EC 1	Adicionar componente Campo texto.	V	Se dibuja un componente de tipo Campo texto. Este se podrá mover a la posición deseada, así como alinearlo. La propiedad campo del componente toma como valor el nombre del dato asociado.	Satisfactorio
EC 2	EC 1.2: Adicionar componente Campo número.	V	Se dibuja un componente de tipo Campo número. Este se podrá mover a la posición deseada, así como alinearlo. La propiedad campo del componente toma como valor el nombre del dato asociado.	Satisfactorio

Diseño del caso de prueba correspondiente al Caso de Uso "Mostrar vista previa".

Descripción General:

El caso de uso se encarga de mostrar una vista del reporte en tiempo de diseño.

Condiciones de Ejecución:

- El usuario debe estar autenticado.
- Debe seleccionar el vínculo Diseñador de Reportes.

Secciones a probar en el Caso de Uso:

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo Central
SC 1: Mostrar vista	EC 1.1: Mostrar vista previa.	El sistema permitirá visualizar una vista del reporte y navegar por las páginas del mismo.	1. El usuario hace clic en el

previa.			menú **Ver**. 2. El usuario selecciona la opción **Vista previa**.

SC 1: Mostrar vista previa

Id del escenario	Escenario	Respuesta del Sistema	Resultado de la Prueba
EC 1	Mostrar vista previa.	El sistema muestra en una ventana independiente una vista del reporte en formato HTML y brinda un mecanismo para navegar sobre las páginas del reporte.	Satisfactorio

3.4 Pruebas de Unidad

El objetivo de las pruebas unitarias es aislar cada parte del programa y mostrar que las partes individuales son correctas. Las pruebas unitarias facilitan que el programador cambie el código para mejorar su estructura, puesto que permiten realizar pruebas sobre los cambios y así asegurarse de que los nuevos cambios no han introducido errores.

3.4.1 Pruebas de Caja Blanca

La prueba de unidad es un tipo de prueba que utiliza el método de Caja Blanca. La prueba de Caja Blanca es una técnica de prueba que garantiza que se ejerciten todos los caminos independientes del módulo, se ejerciten todas las decisiones lógicas, se ejecuten todos los bucles y se ejecuten las estructuras de datos internas. Este método se centra en cómo diseñar los casos de prueba atendiendo al comportamiento interno y la estructura del programa.

En realidad puede ser impracticable realizar una prueba exhaustiva de todos los caminos de un programa. Por ello se han definido distintos criterios de cobertura lógica, que permiten decidir qué sentencias o caminos se deben examinar con los casos de prueba.

Estos criterios son:

- Cobertura de Sentencias: Se escriben casos de prueba suficientes para que cada sentencia en el programa se ejecute al menos una vez.

- Cobertura de Decisión: Se escriben casos de prueba suficientes para que cada decisión en el programa se ejecute una vez con resultado verdadero y otra con el falso.
- Cobertura de Condiciones: Se escriben casos de prueba suficientes para que cada condición en una decisión tenga una vez resultado verdadero y otra falso.
- Cobertura Decisión/Condición: Se escriben casos de prueba suficientes para que cada condición en una decisión tome todas las posibles salidas, al menos una vez, y cada decisión tome todas las posibles salidas, al menos una vez.
- Cobertura de Condición Múltiple: Se escriben casos de prueba suficientes para que todas las combinaciones posibles de resultados de cada condición se invoquen al menos una vez.
- Cobertura de Caminos: Se escriben casos de prueba suficientes para que se ejecuten todos los caminos de un programa. Entendiendo camino como una secuencia de sentencias encadenadas desde la entrada del programa hasta su salida.

Para el desarrollo de las pruebas de unidad se decide emplear como criterio la cobertura de caminos, asegurando que los casos de prueba diseñados permiten que todas las sentencias del programa sean ejecutadas al menos una vez y que las condiciones sean probadas tanto para su valor verdadero como falso.

Como técnica para aplicar dicho criterio de cobertura se tiene a la Prueba del Camino Básico. Esta técnica se basa en obtener una medida de la complejidad del diseño procedimental de un programa. Dicha medida es la complejidad ciclomática, y representa un límite superior para el número de casos de prueba que se deben realizar para asegurar que se ejecuta cada camino del programa.

Los pasos a realizar para aplicar esta técnica son:

- Representar el programa en un grafo de flujo.
- Calcular la complejidad ciclomática.
- Determinar el conjunto básico de caminos independientes.
- Derivar los casos de prueba que fuerzan la ejecución de cada camino.

3.4.2 Casos de prueba de Caja Blanca

En lo adelante se refleja una pequeña porción del diseño de casos de prueba de caja blanca, tomando en cuenta al método getTabularReport. Primeramente se determinó el

grafo de flujos y la complejidad ciclomática, obteniéndose la cantidad de caminos independientes necesarios para desarrollar los casos de prueba correspondientes. Luego de la elaboración del Grafo de Flujos y los caminos a recorrer, se preparan los casos de prueba que forzarán la ejecución de cada uno de esos caminos.

```
public static function
getTabularReport($host,$port,$db,$user,$pass,$sInterface,$sql,$xmlpath,$format='default')
  {
     $oRpt=new PHPReportMaker(); 1
     $oRpt->setConnection(trim($host)); 1
     $oRpt->setPort($port); 1
     $oRpt->setDatabase(trim($db)); 1
     $oRpt->setPassword(trim($pass)); 1
     $oRpt->setUser(trim($user)); 1
     $oRpt->setDatabaseInterface($sInterface); 1
     $oRpt->setSQL(addslashes((htmlspecialchars_decode($sql)))); 1
     $oRpt->setXML($xmlpath); 1
     if ($format == 'default') 2
     {
        $filepath=$oRpt->run(true); 3
     }
        else
     {
       $oOut = $oRpt->createOutputPlugin($format); 4
       $oRpt->setOutputPlugin($oOut); 4
        $path=tempnam(sfConfig::get("sf_root_dir").DIRECTORY_SEPARATOR.'reportsXM
L',$format); 4
        $oRpt->setOutput($path); 4
        $xmlOutput=tempnam(sfConfig::get("sf_root_dir").DIRECTORY_SEPARATOR.'repor
        tsXML','xmlout'); 4
        $oRpt->setXMLOutputFile($xmlOutput); 4
        $filepath = $oRpt->run(); 4
        unlink($xmlOutput); 4
     }
     unlink($xmlpath); 5
```

```
    return $filepath; 5
} 6
```

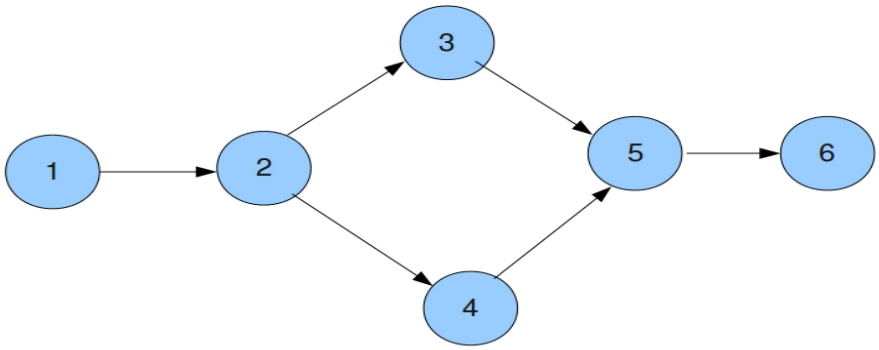


Figura 9. Grafo de Flujo para el método getTabularReport

En este caso se determinó que:

- El grafo de flujo tiene dos regiones.
- V (G) = 5 aristas - 5 nodos + 2 = 2.
- V (G) = 1 nodo predicado + 1 = 2.

Por tanto la complejidad ciclomática del grafo de flujo ilustrado es igual a 2 y se tienen como caminos independientes los siguientes:

- Camino1:1-2-3-5-6
- Camino2:1-2-4-5-6

Para estos caminos independientes se conformaron los siguientes casos de prueba:

Caso de prueba para el Camino1.

Descripción y asignación:
Se quiere determinar si dados los siguientes elementos: información para establecer la conexión a la fuente de datos, consulta del reporte y definición del reporte, se puede generar un reporte con datos en formato XML, para esto es asignado a la variable $format el valor 'default', siguiendo estas especificaciones se garantiza que se ejecute dicho camino independiente en su totalidad.

Resultado esperado:
Al entrar los parámetros y verificar que la variable $format posee el valor 'default' se requiere la generación del reporte en formato XML, el sistema debe generar un reporte con

datos en formato XML satisfactoriamente y debe quedar eliminado cualquier archivo temporal construido durante dicha funcionalidad.
Caso de prueba para el Camino2.

Descripción y asignación:
Se quiere determinar si dados los siguientes elementos: información para establecer la conexión a la fuente de datos, consulta del reporte y definición del reporte, se puede generar un reporte con datos en formato PDF, para esto es asignado a la variable $format el valor 'pdf', siguiendo estas especificaciones se garantiza que se ejecute dicho camino independiente en su totalidad.

Resultado esperado:
Al entrar los parámetros y verificar que la variable $format posee un valor diferente de 'default' se requiere la generación del reporte en el formato indicado, el sistema debe generar un reporte con datos en formato PDF satisfactoriamente y debe quedar eliminado cualquier archivo temporal construido durante dicha funcionalidad.

3.4.3 Automatización de pruebas unitarias. Framework Lime.

Symfony incluye el framework LIME para automatizar las pruebas; las pruebas automatizadas constituyen uno de los mayores avances en la programación desde la orientación a objetos. Lime proporciona el soporte para las pruebas unitarias. Ejecuta los archivos de prueba en un entorno independiente para evitar conflictos entre las diferentes pruebas. Cada prueba unitaria consiste en una llamada a un método de la instancia de lime_test.

Las pruebas unitarias validan la forma en la que los métodos trabajan en cada caso particular. Son archivos PHP cuyo nombre termina en test.php y se encuentran en el directorio test/unit de la aplicación. Para ejecutar el conjunto de pruebas, se utiliza la tarea test-unit desde la línea de comandos. El resultado de esta tarea en la línea de comandos es muy explícito, lo que permite localizar fácilmente las pruebas que han fallado y las que se han ejecutado correctamente. Se crea un nuevo objeto lime_test y el número de pruebas a lanzar se pasa como argumento.

Todas las pruebas unitarias escritas con lime comienzan con el código:

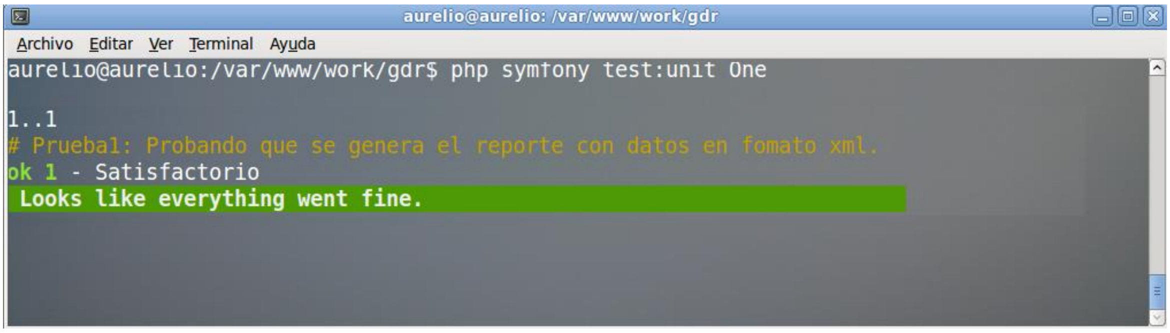
require_once dirname(__FILE__).'/../bootstrap/unit.php';

Para comprobar el funcionamiento interno del método getTabularReport se le realizan pruebas unitarias basadas en los casos de prueba obtenidos mediante el método de Caja Blanca.

```php
<?php
include(dirname(__FILE__).'/../bootstrap/unit.php');
new sfDatabaseManager(ProjectConfiguration::getApplicationConfiguration('report_generator',
        'test', true));
$host = '10.36.20.113';$port = '5432';$db = 'redmine';$user = 'reportes';$pass = 'reportes';
$sInterface = 'postgresql';$format = 'default';$sql = 'SELECT ("id") AS "id" ,("name") AS "name" FROM
                "public"."roles"';
$transformedTemplate = '<?xml version="1.0"?>
<REPORT TITLE="Reporte" ALIGN="none" TYPEPAGE="carta" FLIP="vertical" TYPE="tabular" HEIGHT="797px" WIDTH="612px"
';
$output_expected = '<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
<RP TITLE="untitled" BGCOLOR="#ABC7EC" FLIP="vertical" TYPEPAGE="carta"><PG SZ="50" AL="none" PN="1" WI="612" HE='
<R HEIGHT="27px" WIDTH="612px" POSITION="RELATIVE" TOP="0px" ><C WIDTH="100px" HEIGHT="15px" TOP="12px" LEFT="6px'
<R HEIGHT="27px" WIDTH="612px" POSITION="RELATIVE" TOP="0px" ><C WIDTH="100px" HEIGHT="15px" TOP="12px" LEFT="6px'
<R HEIGHT="27px" WIDTH="612px" POSITION="RELATIVE" TOP="0px" ><C WIDTH="100px" HEIGHT="15px" TOP="12px" LEFT="6px'
<R HEIGHT="27px" WIDTH="612px" POSITION="RELATIVE" TOP="0px" ><C WIDTH="100px" HEIGHT="15px" TOP="12px" LEFT="6px'
</RP>
';
$xmlpath=tempnam(sfConfig::get('sf_cache_dir'), 'rdl');
$handle = fopen ( $xmlpath, 'w' );
fwrite ( $handle,$transformedTemplate );
fclose ( $handle );
$path = Report::getTabularReport($host, $port, $db, $user, $pass, $sInterface, $sql, $xmlpath,$format);
$xml_with_data = file_get_contents($path);
$t = new lime_test(1, new lime_output_color());
$t->diag('Prueba1: Probando que se genera el reporte con datos en fomato xml.');
$t->is($xml_with_data, $output_expected,'Satisfactorio');
?>
```

Figura 10. Prueba unitaria sobre el método getTabularReport

El método de prueba utilizado correspondiente a la clase lime_test fue "is" que compara 2 valores y la prueba pasa si ambos son iguales.



```
aurelio@aurelio: /var/www/work/gdr
Archivo Editar Ver Terminal Ayuda
aurelio@aurelio:/var/www/work/gdr$ php symfony test:unit One

1..1
# Prueba1: Probando que se genera el reporte con datos en fomato xml.
ok 1 - Satisfactorio
 Looks like everything went fine.
```

Figura 11. Ejecución de la prueba unitaria

3.5 Evaluación de las pruebas

Los elementos que se garantizan en la realización de las pruebas son la usabilidad y la funcionalidad del módulo Diseñador de Reportes. Para ello se intentan encontrar errores en las siguientes categorías:

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores de estructuras de datos.

- De acceso a bases de datos.
- Errores de rendimiento.
- De inicialización y terminación.

3.5.1 Gestión de incidencias con RedMine

De forma general los procesos de gestión se formalizan por medio de herramientas que facilitan y mejoran sus actividades. Los proyectos de DATEC en particular, son gestionados a través de la herramienta RedMine. Dicha herramienta facilita una serie de capacidades necesarias para un correcto seguimiento y administración de los elementos de gestión correspondientes a los proyectos informáticos y sus respectivos productos.

Para el módulo Diseñador de Reportes correspondiente al Generador Dinámico de Reportes, con el RedMine se garantiza una correcta documentación y gestión de las No Conformidades. Los clientes o revisores del módulo reflejan las No conformidades detectadas, reduciendo al máximo el tiempo entre la detección y el registro de las incidencias. Por otra parte son analizadas rápidamente por los miembros del equipo de desarrollo y la dirección del proyecto para su corrección en un tiempo estimado según la prioridad asignada a las mismas. Si se realiza algún pedido de cambio por los clientes se reúne el comité de cambio para deliberar si procede o no.

3.5.2 Resultados obtenidos

Como se muestra en la siguiente gráfica, la realización de pruebas unitarias utilizando el framework Lime abarcó más del 80% del código correspondiente al módulo Diseñador de Reportes.

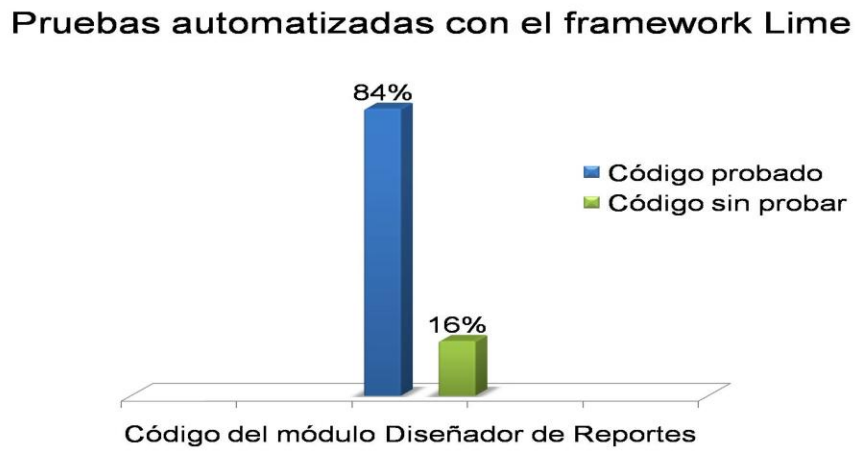


Figura 13. Alcance de las pruebas unitarias

Con la realización de pruebas de sistema se comprobaron todas las funcionalidades del módulo Diseñador de Reportes, puesto que se diseñaron y ejecutaron casos de prueba para todos los casos de uso implicados en dicho módulo.

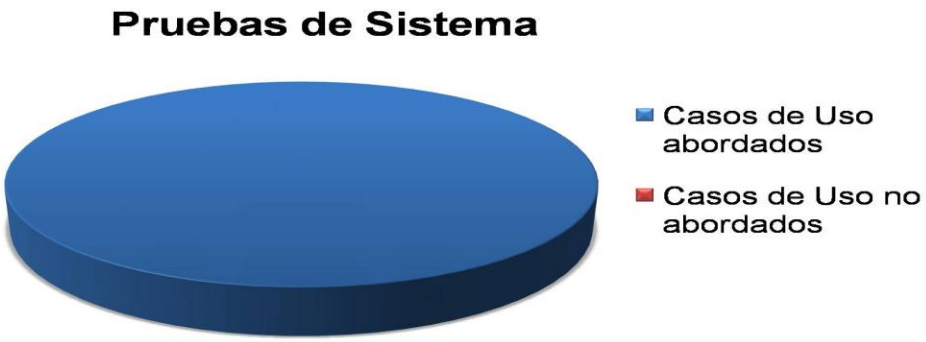


Figura 12. Casos de Uso implicados en las pruebas de sistema

Para un mejor entendimiento de los resultados arrojados por las pruebas de sistema, se dividieron en dos etapas. La primera etapa comprende los resultados de las pruebas realizadas en el proyecto por el grupo de calidad y algunos de los desarrolladores. Se realizó la revisión del proyecto en tres iteraciones de las pruebas internas donde se hicieron pruebas de sistema. La segunda etapa está comprendida por los resultados obtenidos de la liberación del proyecto por parte de ERP.

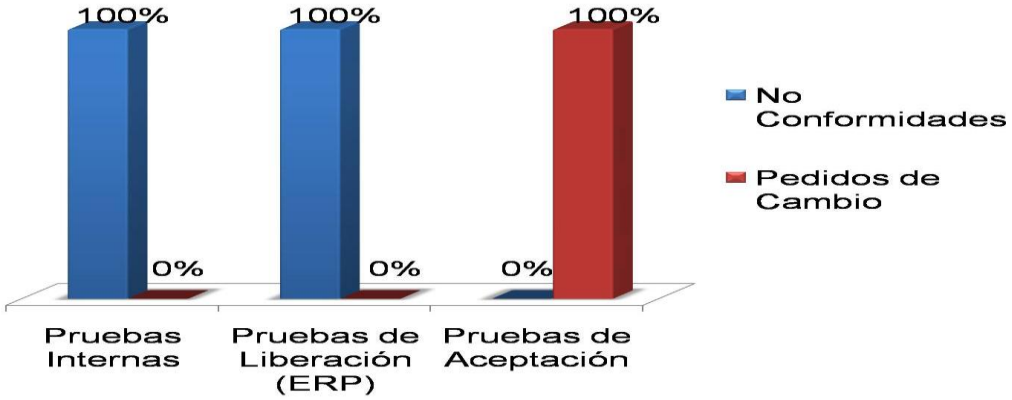


Figura 14. Tendencia de las incidencias detectadas

La gráfica anterior muestra que se limaron todas las asperezas detectadas, antes de poner la aplicación en manos del cliente y en un entorno real. El número de No Conformidades posterior al despliegue se redujo a 0, demostrando un elevado grado de satisfacción por parte de los clientes. También se puede apreciar un incremento de las solicitudes de cambio durante dicha etapa, debido a la colaboración directa de los clientes quienes tienen un dominio real del negocio.

El módulo Diseñador de Reportes incluido en el Generador Dinámico de Reportes, se encuentra en una versión prototipo que está siendo explotada por varios clientes, desde mediados del 2009 y por tanto surgen solicitudes de cambio que se concretan generalmente en la adición de nuevas funcionalidades o usos. Entre los principales clientes que colaboran en su evolución y mantenimiento se destacan: ERP, ONE, ADUANA, Alfa Omega, Dirección Técnica y el propio DATEC.

3.6 Valoración de los resultados

La realización de pruebas unitarias garantizó el correcto funcionamiento de los métodos probados en el módulo Diseñador de Reportes. Como dichas pruebas abarcaron la mayor parte del código correspondiente al módulo en cuestión, se considera mucho más seguro refactorizar su código y añadirle nuevas características.

Figura 15. Ventajas de probar la mayor parte del código

Los resultados obtenidos de las pruebas de sistema realizadas al Diseñador de Reportes fueron satisfactorios puesto que se obtuvo una solución con un elevado grado de usabilidad y operatividad. En consecuencia fue demostrada la satisfacción de los clientes así como la calidad de la aplicación.

Figura 16. Ventajas de la realización de pruebas de sistema

Cuando un software es útil, los interesados lo prueban hasta llevarlo al límite o más allá de su dominio original. La presión para aumentar las funciones viene principalmente de usuarios a los que les gustan las funciones básicas y les descubren nuevos usos. De

manera general el conjunto de Solicitudes de Cambio identificadas, se tradujo en nuevas funcionalidades para el módulo Diseñador de Reportes. Lógicamente, la adición de nuevos usos a este módulo le impone una mayor completitud y le provee mejores oportunidades para concretar nuevos clientes.

Figura 17. Ciclo de los pedidos de cambio

Los siguientes factores influyen en la evolución del Diseñador de Reportes y además tributan a la calidad del mismo.

- El módulo Diseñador de Reportes incluido en el Generador Dinámico de Reportes, se encuentra en explotación por numerosos clientes considerados muy importantes tanto para la Universidad de las Ciencias Informáticas como para el país.
- La gestión de No Conformidades aplicando la herramienta RedMine que permite formalizar dicha actividad priorizando la colaboración de los clientes.
- La puesta en marcha de forma organizada y rápida de un proceso de soporte al Generador Dinámico de Reportes con el módulo Diseñador de Reportes incluido, paralela a la construcción de otras versiones del mismo. (Ver Anexo 4)
- El tiempo en explotación del módulo Diseñador de Reportes por los clientes, confirmando a la práctica como criterio de la verdad, que en este caso envuelve las especificaciones del programa.

El camino evolutivo del Diseñador de Reportes se infiere a partir de la presión de cambio por parte de sus principales clientes y de la necesidad de alcanzar la mayor estabilidad posible de sus funciones básicas.

3.7 Métricas del software aplicadas

Aunque las métricas técnicas para el software de computadora no son absolutas, nos proporcionan una manera sistemática de valorar la calidad basándose en un conjunto de reglas claramente definidas. También le proporcionan al ingeniero del software una visión interna en el acto, en vez de a posteriori. Esto permite al ingeniero descubrir y corregir problemas potenciales antes de que se conviertan en defectos catastróficos. (Pressman, 2005)

3.7.1 Métricas de diseño arquitectónico

Toman en cuenta los rasgos asociados a la estructura y eficiencia de los componentes que forman la arquitectura de un sistema.

Card y Glass definen tres medidas de la complejidad del diseño del software: complejidad estructural, complejidad de datos y complejidad del sistema. (Pressman, 2005)

La complejidad estructural $S(i)$ de un módulo i se determina usando la ecuación: $S(i) = f_{out}^2(i)$ donde $f_{out}(i)$ es la expansión del módulo i, que representa el número de módulos inmediatamente subordinados.

La complejidad de datos $D(i)$ de un módulo i se define como la proporción entre el número de variables de entrada y salida del módulo $V(i)$ y su expansión $f_{out}(i)$.

$$D(i) = \frac{V(i)}{f_{out}(i) + 1}$$

La complejidad del sistema $C(i)$ se calcula sumando las complejidades estructural y de datos.

$$C(i) = S(i) + D(i)$$

Tabla 15 Umbrales de Complejidad

Complejidad del Sistema			
Complejidad	**S(i)**	**D(i)**	**C(i)**
No complejo	1, 4. 9, 16 ,25	<=7	<=32
Complejo	36, 49, 64	>7 y <=12	>32 y <=50
Muy complejo	81…	>12	>50

La aplicación de dichas métricas sobre el módulo Diseñador de Reportes arrojó los siguientes resultados:

$$S(i) = 2^2 = 4$$

$$D(i) = \frac{10}{2+1} = 3.33$$

$$C(i) = 4 + 3.33 = 7.33$$

Según las clasificaciones mostradas en la tabla anterior se puede llegar a la conclusión de que el sistema no es muy complejo.

3.7.2 Métrica Árbol de Profundidad de Herencia

En su totalidad los autores destacan la necesidad de medir las estructuras hereditarias en términos de profundidad o de densidad de nodos. Dichas jerarquías pueden medirse como la profundidad de cada clase dentro de su jerarquía, es decir, la longitud máxima desde el nodo que representa la clase hasta la raíz del árbol. A medida que crece el valor del Árbol de Profundidad de Herencia (APH) la complejidad del módulo aumenta haciéndose más difícil predecir el comportamiento de las clases así como mantenerlas.

Lorenz y Kidd sugieren un umbral de 6 niveles como indicador de un abuso en la herencia en distintos lenguajes de programación. Durante la aplicación de la métrica APH al módulo Diseñador de Reportes se obtuvo el valor 4 como el nivel más alto de herencia, lo cual se encuentra por debajo del umbral definido. En consecuencia se determina que se hace un uso correcto de la herencia proporcionada por la orientación a objetos y que los objetos existentes no son: complejos, difíciles de testear y reusar.

3.7.3 Métrica Tamaño de Clase

El tamaño de una clase i está basado en el número de sus operaciones y atributos. Valores elevados de Tamaño de Clase (TC), indican que la clase posee demasiada responsabilidad, por lo que disminuye la reusabilidad de la misma y se dificulta su mantenimiento. (Pressman, 2005)

Para medir el tamaño de clase se tienen en cuenta los siguientes aspectos:

- Total de operaciones (las propias y las heredadas de las clases padres).
- Cantidad de atributos (los propios y los heredados de las clases padres).
- Promedio general de los dos anteriores para el módulo completo.

Las clases se clasifican en tres grupos según su tamaño, los cuales se representan en la siguiente tabla junto con los umbrales seleccionados para su clasificación.

Tabla 16 Umbrales para el TC

Clasificación	Valores de los umbrales
Pequeña	<=20
Media	>20 y <=30
Grande	>30

Se aplicó dicha métrica para las principales clases de las tres capas definidas en el módulo.

Tabla 17 Tamaño de Clase

Clases	N°. de atributos	N°. de operaciones
1. createReportFromDesignAction	1	12
2. addReportAction	1	12
3. getXmlReportByIdAction	1	12
4. getTableFieldsAction	1	12
5. showPreviewFromDesignAction	1	12
6. getReportPagesAction	1	12
7. Report	20	33
8. Template	16	20
9. Model	9	21
10. TLabel	9	8
11. Text	11	9
12. FieldTD	7	3
13. WorkflowTD	7	11
14. Factory	7	15
Promedio	**6.57**	**13.71**

Se presentó un total de 14 clases para un promedio de atributos de 6.57 y un promedio de operaciones de 13.71. Concluyéndose que tanto el promedio de atributos como de operaciones se encuentran en el umbral de clases medianas.

3.8 Conclusiones del capítulo

Durante este capítulo se abordaron las pruebas unitarias y las pruebas de sistema que se le realizaron al software. Para lograr aumentar la detección de No Conformidades se conformaron los casos de prueba a ejecutar. Se mostraron los resultados obtenidos durante la realización de las pruebas siendo posible determinar que el software posee una calidad aceptable. Se identificó que la evolución del módulo está guiada por la fuerte comunicación establecida con los clientes. La aplicación de métricas permitió determinar que dicho módulo es poco complejo, presenta de manera general clases de complejidad media y no es difícil de mantener.

CONCLUSIONES

- El estudio del estado del arte a los diseñadores de reportes: Crystal Report Designer, ActiveReport Designer, Pentaho Report Designer y Report Manager Designer, mostró que no satisfacen las necesidades de integración y evolución requeridas por los clientes.
- A partir del análisis de tecnologías actuales para el desarrollo de software y bajo el principio de emplear tecnologías libres y de fácil integración, se seleccionaron lenguajes, frameworks, entorno de desarrollo y metodología a utilizar para la implementación del módulo Diseñador de Reportes.
- La aplicación del estándar de codificación definido por DATEC potenció la legibilidad y el mantenimiento del código.
- Los patrones de diseño aplicados lograron una lógica organizacional que promovió la alta cohesión y el bajo acoplamiento entre las clases.
- Con la implementación del módulo Diseñador de Reportes, se obtuvo una aplicación web basada en tecnologías libres, que permite al usuario final diseñar reportes personalizados.
- La interfaz gráfica implementada se distingue por ser orientada al usuario final y por tener un alto grado de usabilidad y operatividad.
- La realización de pruebas de Caja Blanca y de Caja Negra, demostraron que el módulo Diseñador de Reportes permite diseñar y generar reportes satisfactoriamente.
- La evolución del módulo Diseñador de Reportes está guiada por la fuerte comunicación entre el equipo de desarrollo y los clientes.
- La aplicación de métricas permitió determinar que dicho módulo es poco complejo, presenta de manera general clases de complejidad media y no es difícil de mantener.

RECOMENDACIONES

Luego de la presentación del estudio realizado que culmina con la implementación del módulo Diseñador de Reportes, perteneciente al Generador Dinámico de Reportes incluido en PATDSI, se listan a continuación una serie de recomendaciones para la construcción de nuevas versiones de dicho módulo:

- Agregar un componente para gráfico que se integre con el servidor de gráficos (ChartServer) de DATEC, el cual ya se encuentra desarrollado.
- Incorporar en la herramienta el diseño de reportes con tablas cruzadas y tablas pivote, para aumentar la capacidad de análisis a los usuarios.
- Impartir cursos de capacitación a las personas que trabajarán con la aplicación.
- Mantener sobre el sistema un estricto cumplimiento del proceso de soporte y la más estrecha colaboración con los clientes, logrando así que se eleve la fiabilidad y funcionamiento óptimo del sistema.
- Preservar el principio de desarrollar con tecnologías novedosas, libres y de fácil integración.

BIBLIOGRAFÍA REFERENCIADA

Arias, Emilio. 2006. *Un año de Business Intelligence.* 2006. pág. 11.

ComponentSource. 1996-2010. [En línea] 1996-2010. [Citado el: 14 de Diciembre de 2009.] http://www.componentsource.com/products/activereports-6/summary-es.html.

Eclipse. 2008. Introduction to OpenUP. [En línea] 2008. [Citado el: 22 de Enero de 2010.] http://epf.eclipse.org/wikis/openup/.

Eguíluz Pérez, Javier. 2009. *Introducción a JavaScript.* 2009.

Ext. 2006-2010. Ext JS. [En línea] 2006-2010. [Citado el: 28 de Enero de 2010.] http://www.extjs.com/ .

González, Fernando, Cid, Moisés y Cuesta, Pedro. 2000. Desarrollo de aplicaciones web utilizando ASP (Active Server Pages). [En línea] Agosto de 2000. [Citado el: 1 de Febrero de 2010.] http://trevinca.ei.uvigo.es/~pcuesta/publicaciones/asp.pdf.

Herz, Andreas. 2010. Open-jACOB Draw2D . [En línea] 2010. [Citado el: 1 de Febrero de 2010.] http://www.draw2d.org/draw2d/.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Education.S.A, 2000. 84-7829-036-2.

Microsoft Corporation. 2009. [En línea] 2009. [Citado el: 14 de Diciembre de 2009.] http://msdn.microsoft.com/es-es/library/aa287920%28VS.71%29.aspx.

—. 2001. Ayuda de javascript y Vbscript para Astalaweb. [En línea] 2001. [Citado el: 1 de Febrero de 2010.] http://javascripts.astalaweb.net/Ayuda/index.asp.

Pentaho Corporation. 2007. Manual Generador de Reportes de Pentaho. [En línea] 2007. [Citado el: 15 de Diciembre de 2009.] http://www.google.com.cu/url?sa=t&source=web&ct=res&cd=1&ved=0CAcQFjAA&url=http%3A%2F%2Fwww.onuva.com%2Ffiles%2Fpentaho%2FManual_PRD.pdf&rct=j&q=dise%C3%B1ar+reportes+con+Pentaho&ei=3qInS-XZBcaolAe8gs2kDQ&usg=AFQjCNF6d8_Og0ZcZkbLrv7KZLowr8pOPQ.

Potencier, Fabien y Zaninotto, Francois. 2008. *Symfony la guía definitiva.* 2008.

Propel project. 2004. Propel - Guía de usuario. [En línea] 2004. [Citado el: 1 de Febrero de 2010.] http://propel.phpdb.org/docs/es/user_guide/.

S. Pressman, Roger. 2005. *Ingeniería de software. Un enfoque práctico.* 2005.

Sun Microsystems, Inc. 1994-2010. [En línea] 1994-2010. [Citado el: 2 de Febrrero de 2010.] http://es.sun.com/sunnews/press/2009/20091214.jsp.

Tavárez, David. 2009. Maestros del web. *Comparación de Frameworks en Javascript.* [En línea] 10 de Noviembre de 2009. [Citado el: 28 de Enero de 2010.] http://www.maestrosdelweb.com/editorial/comparacion-frameworks-javascript/.

the PHP Documentation Group. 1997 - 2008. Manual de PHP. [En línea] 1997 - 2008. [Citado el: 25 de Enero de 2010.] http://www.php.net/docs.php.

W3C. 2010. XML ESSENTIALS. [En línea] 2010. [Citado el: 7 de Febrero de 2010.] http://www.w3.org/standards/xml/core.

Zend Technologies Ltd. 2009. Zend Studio. [En línea] 2009. [Citado el: 6 de Febrero de 2010.] http://www.zend.com/en/products/studio/.

BIBLIOGRAFÍA CONSULTADA

Arias, Emilio. 2006. *Un año de Business Intelligence.* 2006. pág. 11.

Brooks, Frederick P. 1987. *No Silver Bullet: Essence and Accidents of Software Engineering.* 1987. Vol. 20.

Caballero, Leonardo y Lara, Jesús. 2007. [En línea] 2007. http://www.slideshare.net/lcaballero/patrones-de-diseo-y-orienacin-a-objetos-en-php5.

ComponentSource. 1996-2010. [En línea] 1996-2010. [Citado el: 14 de Diciembre de 2009.] http://www.componentsource.com/products/activereports-6/summary-es.html.

Eclipse. 2008. Introduction to OpenUP. [En línea] 2008. [Citado el: 22 de Enero de 2010.] http://epf.eclipse.org/wikis/openup/.

Eguíluz Pérez, Javier. 2009. *Introducción a JavaScript.* 2009.

Ext. 2006-2010. Ext JS. [En línea] 2006-2010. [Citado el: 28 de Enero de 2010.] http://www.extjs.com/ .

González, Fernando, Cid, Moisés y Cuesta, Pedro. 2000. Desarrollo de aplicaciones web utilizando ASP (Active Server Pages). [En línea] Agosto de 2000. [Citado el: 1 de Febrero de 2010.] http://trevinca.ei.uvigo.es/~pcuesta/publicaciones/asp.pdf.

GrapeCity, inc. 1997-2009. [En línea] 1997-2009. http://www.datadynamics.com/Products/ProductOverview.aspx?Product=ARNET3.

Gravitar. 2009. [En línea] 2009. http://www.gravitar.biz/index.php/herramientas-bi/pentaho/caracteristicas-pentaho/.

Hernández Meléndrez, Edelsys. 2006. *Cómo escribir una tesis.* 2006.

Hernández Sampieri, Roberto, Fernández Collado, Carlos y Baptista Lucio, Pilar. 2006. *Metodologia de la investigación.Cuarta edición.* 2006. ISBN 970-10-5753-8.

Herz, Andreas. 2010. Open-jACOB Draw2D . [En línea] 2010. [Citado el: 1 de Febrero de 2010.] http://www.draw2d.org/draw2d/.

Jacobson, Ivar, Booch, Grady y Rumbaugh, James. 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid : Pearson Education.S.A, 2000. 84-7829-036-2.

Lorenz, M. y Kidd, J. 1994. *Object Oriented Metrics.* 1994.

Microsoft Corporation. 2009. [En línea] 2009. [Citado el: 14 de Diciembre de 2009.] http://msdn.microsoft.com/es-es/library/aa287920%28VS.71%29.aspx.

—. 2001. Ayuda de javascript y Vbscript para Astalaweb. [En línea] 2001. [Citado el: 1 de Febrero de 2010.] http://javascripts.astalaweb.net/Ayuda/index.asp.

—. 2010. MSDN. *Microsoft Developer Network.* [En línea] 2010. [Citado el: 3 de Febrero de 2010.] http://msdn.microsoft.com/es-es/library/bb972242.aspx.

Muñoz Organero, Mario. 2008. Capa de presentación: Java Server Pages (JSP). [En línea] 2008. [Citado el: 1 de Febrero de 2010.] http://www.it.uc3m.es/mario/si/07-08/Tema7.pdf.

Pentaho Corporation. 2007. Manual Generador de Reportes de Pentaho. [En línea] 2007. [Citado el: 15 de Diciembre de 2009.] http://www.google.com.cu/url?sa=t&source=web&ct=res&cd=1&ved=0CAcQFjAA&url=http%3A%2F%2Fwww.onuva.com%2Ffiles%2Fpentaho%2FManual_PRD.pdf&rct=j&q=dise%C3%B1ar+reportes+con+Pentaho&ei=3qInS-XZBcaolAe8gs2kDQ&usg=AFQjCNF6d8_Og0ZcZkbLrv7KZLowr8pOPQ.

Potencier, Fabien y Zaninotto, Francois. 2008. *Symfony la guía definitiva.* 2008.
Propel project. 2004. Propel - Guía de usuario. [En línea] 2004. [Citado el: 1 de Febrero de 2010.] http://propel.phpdb.org/docs/es/user_guide/.
Rangel, Eustaquio. 2006. *PHPReports Manual.* Brasil : s.n., 2006.
Rodríguez Baena, Luis. 2003. Programación en Java. [En línea] 2003. http://www.colimbo.net/documentos/documentacion/Java/JavaTema07.pdf.
S. Pressman, Roger. 2005. *Ingeniería de software. Un enfoque práctico.* 2005.
sourceforge.net. [En línea] http://reportman.sourceforge.net/indexes.html.
Sun Microsystems, Inc. 1994-2010. [En línea] 1994-2010. [Citado el: 2 de Febrrero de 2010.] http://es.sun.com/sunnews/press/2009/20091214.jsp.
Tavárez, David. 2009. Maestros del web. *Comparación de Frameworks en Javascript.* [En línea] 10 de Noviembre de 2009. [Citado el: 28 de Enero de 2010.] http://www.maestrosdelweb.com/editorial/comparacion-frameworks-javascript/.
the PHP Documentation Group. 1997 - 2008. Manual de PHP. [En línea] 1997 - 2008. [Citado el: 25 de Enero de 2010.] http://www.php.net/docs.php.
Tobarra Narro, Manuel. 2003. *CMM y RUP: Una perspectiva común.* 2003.
W3C. 2010. XML ESSENTIALS. [En línea] 2010. [Citado el: 7 de Febrero de 2010.] http://www.w3.org/standards/xml/core.
Zend Technologies Ltd. 2009. Zend Studio. [En línea] 2009. [Citado el: 6 de Febrero de 2010.] http://www.zend.com/en/products/studio/.

ANEXOS

Anexo 1: Representación del flujo de información durante la vida del reporte.

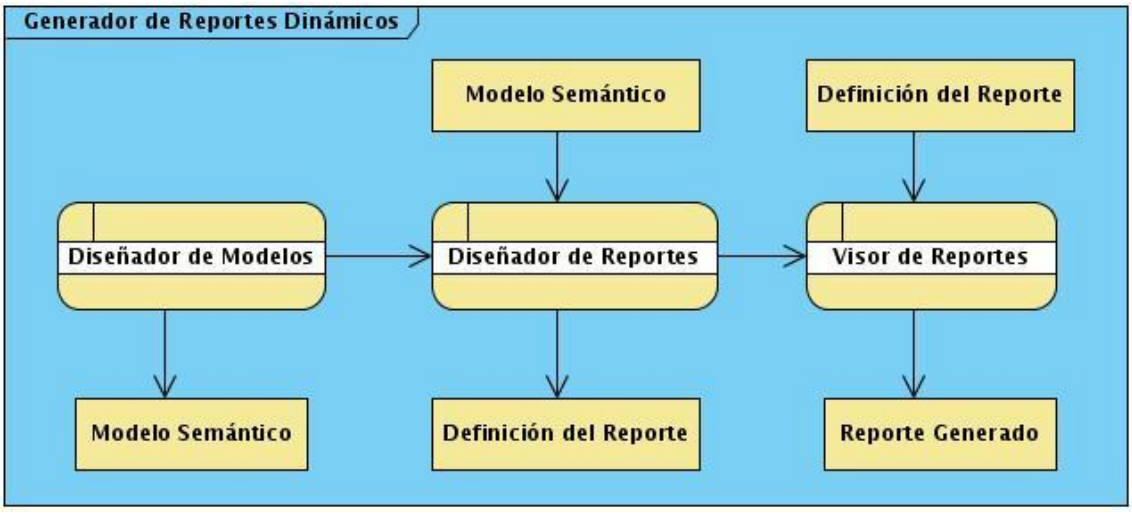


Figura 8. Ciclo de vida del reporte

Anexo 2: Diagramas de componentes correspondientes al subsistema Vista.

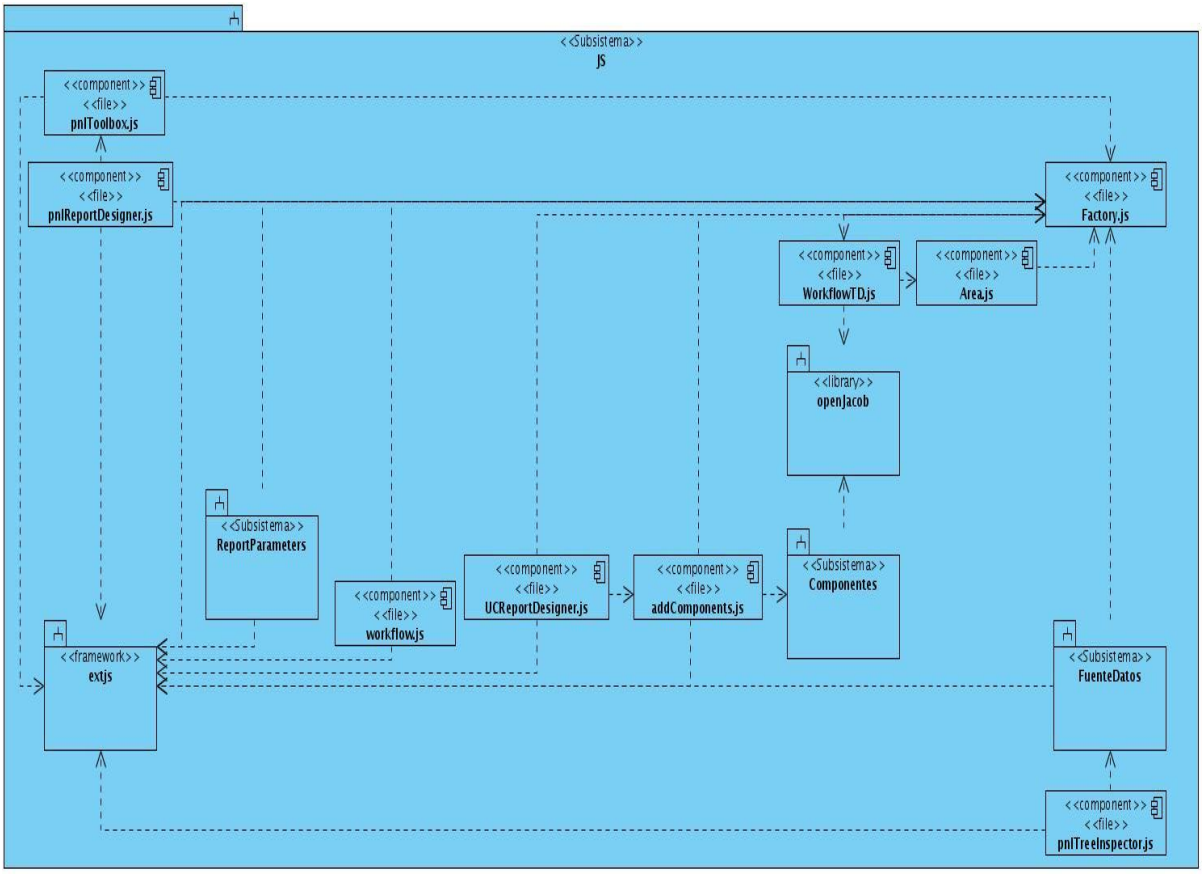


Figura 19. Diagrama de componentes del subsistema JS

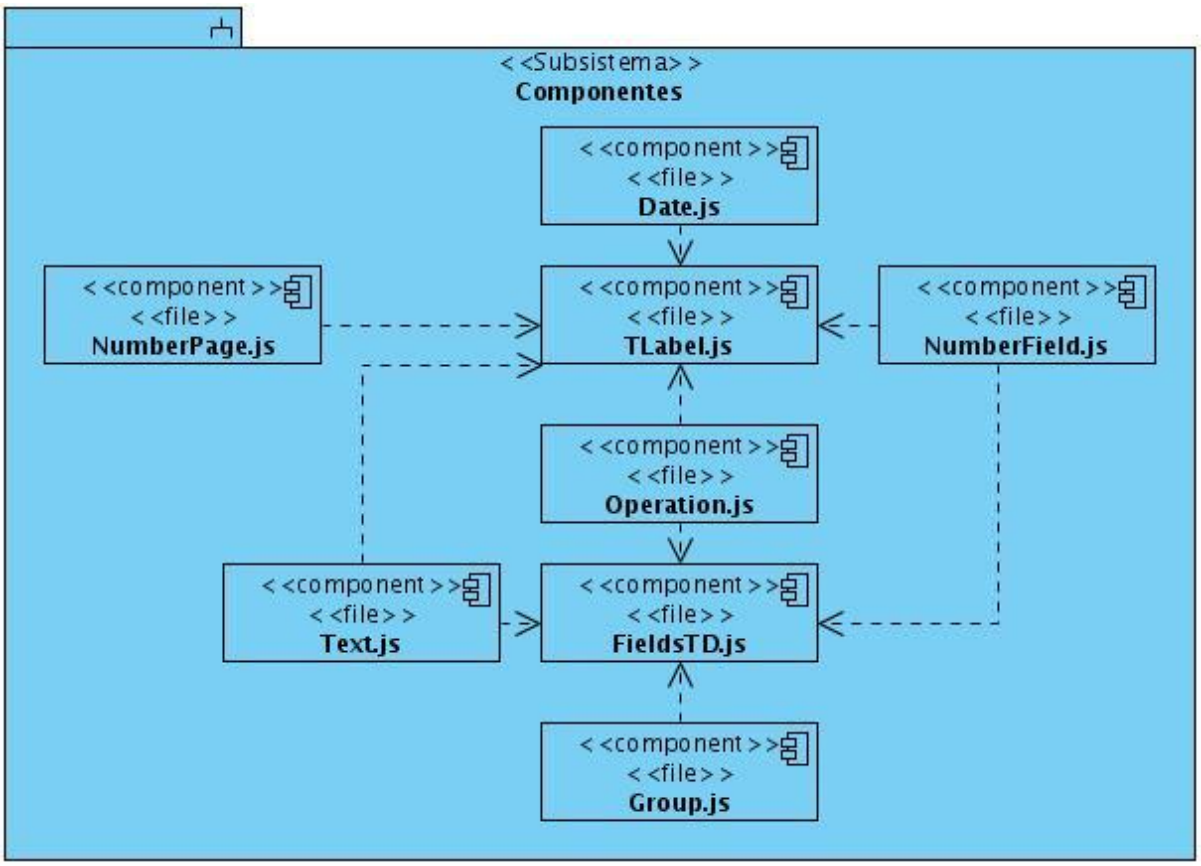


Figura 20. Diagrama de componentes del subsistema Componentes

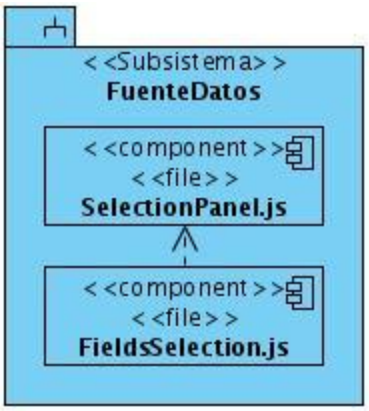


Figura 21. Diagrama de componentes del subsistema FuenteDatos

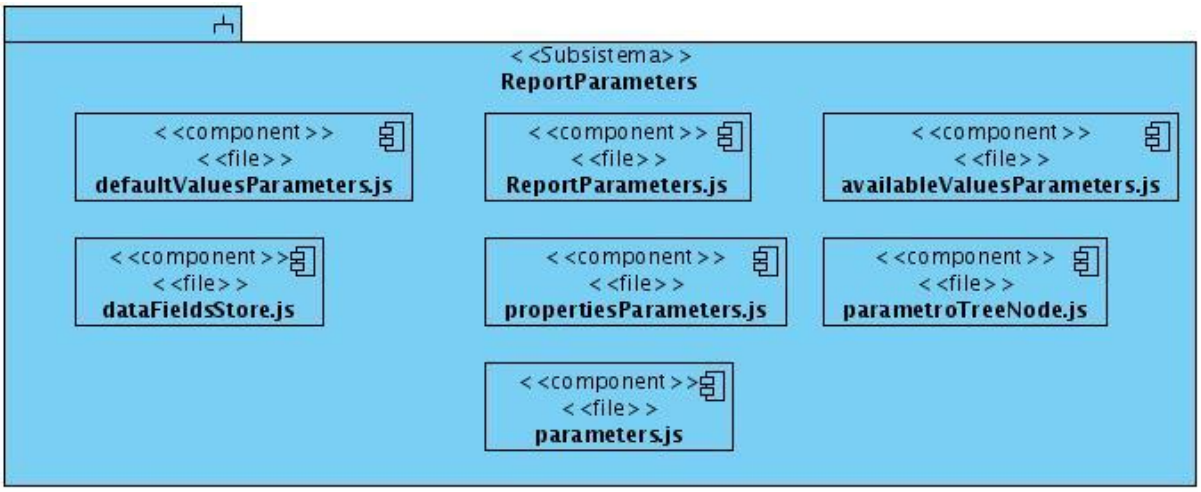


Figura 22. Diagrama de componentes del subsistema ReportParameters

Anexo 3: Evidencia de la No Conformidad "Error en la Vista Previa"

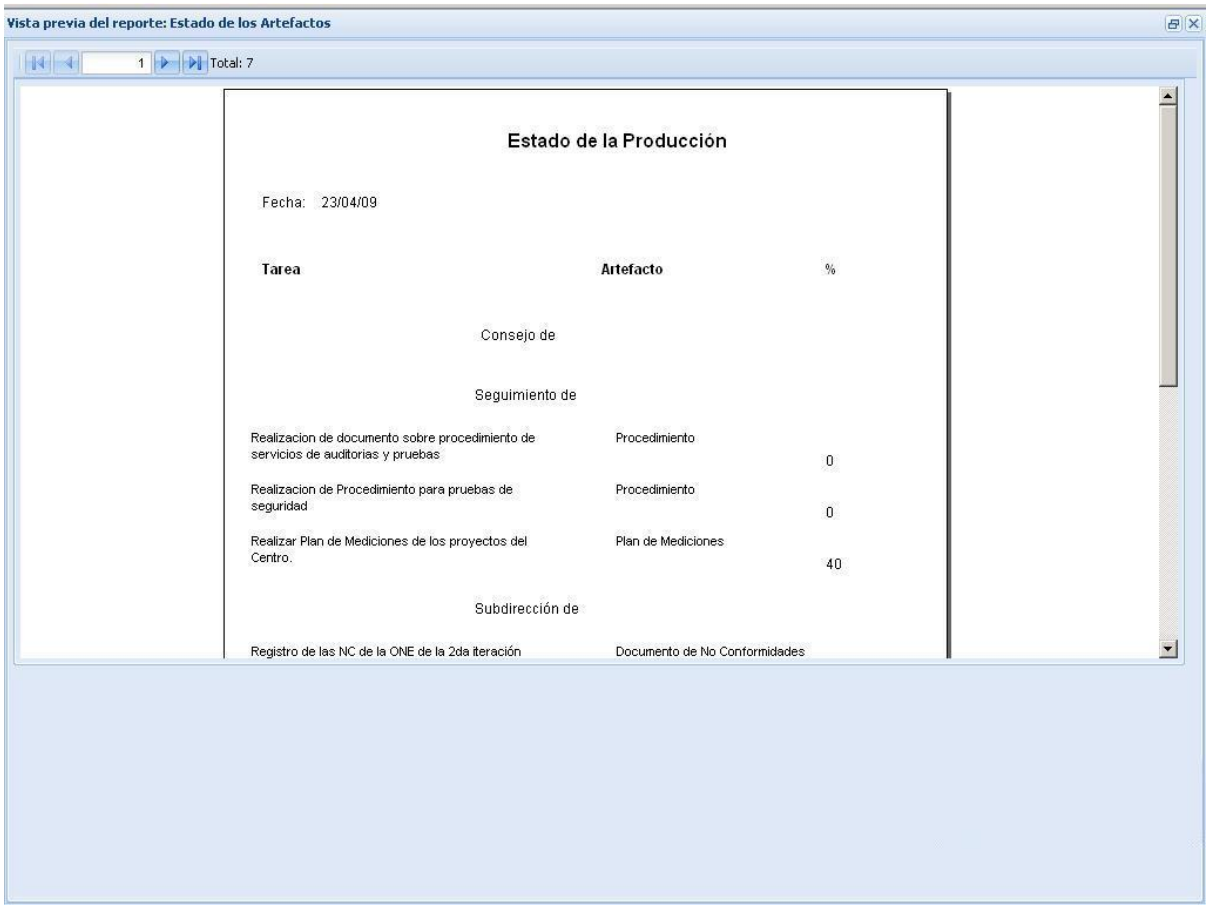


Figura 23. La Vista Previa no se ajusta al tamaño de la pantalla

Anexo 4: Procedimiento de soporte llevado a cabo por el DATEC para sus productos.

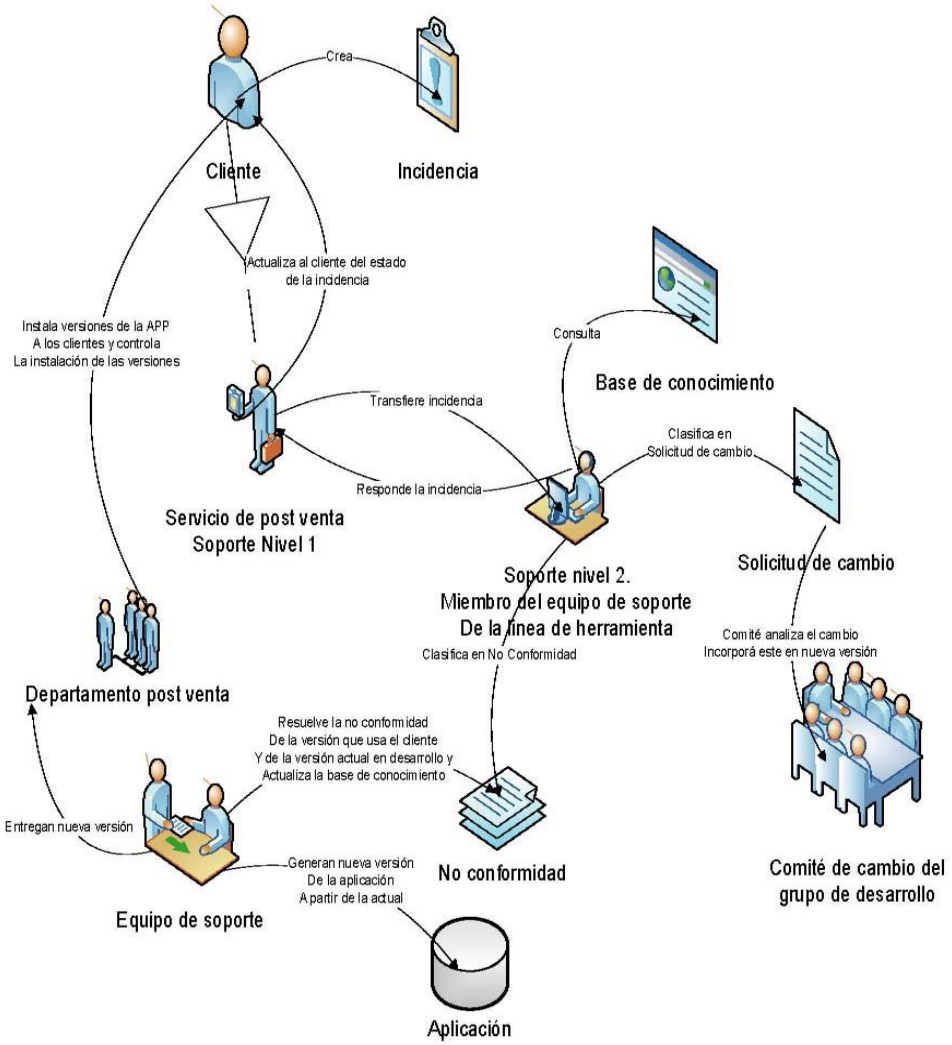


Figura 24. Ciclo de vida de las incidencias