Universidad de las Ciencias Informáticas

Facultad 6

Título: “Componente para la representación gráfica de datos en los reportes diseñados con el Generador Dinámico de Reportes v2.0”

Trabajo de diploma para optar por el título de ingeniero en ciencias informáticas.

Autores:

Tahimi Núñez Rondón

Laura Mercedes Camacho González

Tutores:

MSc. Yunet González Mulet

Ing. Yoander Iñiguez Bermúdez

24 de junio de 2015

“Año 57 del Triunfo de la Revolución”


"Seamos realistas y hagamos
lo imposible."
Che

Declaración de Auditoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Tahimi Núñez Rondón

_____

Firma del Autor

Laura Mercedes Camacho González

_____

Firma del Autor

MSc. Yunet González Mulet.

_____

Firma del Tutor

Ing. Yoander Iñiguez Bermúdez

_____

Firma del Tutor

Datos de Contactos

Tutor: MSc. Yunet González Mulet.

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

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

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

Correo Electrónico: ygonzalezmu@uci.cu

Tutor: Ing. Yoander Iñiguez Bermúdez.

Especialista del centro DATEC con 4 años de experiencia y un alto dominio en el desarrollo de aplicaciones web.

Formación Académica: Ingeniero en Ciencias Informáticas (2010)

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

Correo Electrónico: yiniguez@uci.cu

Agradecimientos

A todas las personas que de una forma u otra hicieron que mi estancia en la UCI fuera más agradable.

A los tutores Yunet y Yoander por su paciencia y oportunas indicaciones.

A los amigos que empezaron y no pudieron terminar y a los que están aquí todavía.

A mis amigos Raciel, Emilio, Luis, Omar, Esley por brindarme su amistad incondicional.

A mis amigas Celia, Tahimi y Aylin, que han hecho de estos cinco años de carrera algo memorable y a Jisel, Lisandra y Karolay por darme la oportunidad de conocerlas.

A mí novio Eliodanis, por haber estado a mi lado y haberme brindado su cariño y apoyo durante estos últimos tres años y por compartir conmigo todo lo bueno y lo malo.

A mi familia, por nunca decir que no y darme ánimos para continuar.

Agradecer a mis abuelas, por compartir sus experiencias y demostrarme que el que lucha alcanza, en especial a mi abuelita Emma que siempre ha estado a mi lado.

A mis hermanos, por darme a las sobrinas más lindas, por quererme, respetarme y esperar siempre lo mejor de mí.

A mis padres, por ser los mejores padres del mundo y conocerme mejor que nadie, por ser el bastón en el que siempre he encontrado sostén, por guiarme en la vida por el mejor camino posible y por hacer posible este sueño. Gracias por ser mis padres.

Laura M. Camacho González

Lo más valioso en la vida no es lo que tenemos, sino a quienes tenemos. Por eso quisiera agradecer a las personas más valiosas de mi vida, mi familia en especial a mi mama, mi papa y mi hermana.

A mis amigos que son la familia que se escoge y que pienso que yo hice de las mejores elecciones Ana Ailin, Ailin, Lisandra, Jisel, Karo, Jose, Emilio, Raciel , Luis Enrique, Eric y en especial a mi compañera de tesis Laura.

A mis compañeros de aula, de edificio y del barrio y a Mis mamas Migdalia e Irasema.

A mis tutores Yoander, Asnay, Yunet, Esley y a mis profes que me han preparado a lo largo de la carrera.

A Deysi, Polo, Yandy y en especial a mi novio Yoandi por hacerme ver cuando todo estaba difícil que si se podía.

Gracias a todos por compartir sueños como este, y ayudarme a convertirme en una persona más preparada y de convicciones firmes en la vida.

Tahimi Núñez Rondón

Dedicatoria

En especial a mis padres por estar presentes en todos los momentos de mi carrera y a mi familia y amigos.

Laura M. Camacho González

Dedico los resultados de mi trabajo a el esfuerzo intenso de cada de día de mi mama Mariluz, mi hermana Yordania y en especial a mi papá Luis Enrique, soy todo lo que soy por ustedes y no creo q me alcanzaría una vida para agradecerles. Los quiero mucho.

Tahimí Núñez Rondón.

Resumen

En el Centro de Tecnologías de Gestión de Datos, de la Universidad de las Ciencias Informáticas, se desarrolla la versión 2.0 del Generador Dinámico de Reportes (GDR v2.0). Uno de sus módulos es el Diseñador de Reportes, incluye un grupo de elementos como campos de texto, líneas, imágenes, entre otros, que se emplean en el diseño de reportes, pero no incluye componente alguno para la representación gráfica de los datos que estos contienen. El presente trabajo se desarrolló con el objetivo de incorporar el componente de graficado en dicho módulo y aumentar así el enriquecimiento visual e interpretación de datos de los reportes. Para ello se realiza un análisis de la propuesta de solución y de su desarrollo empleando varias herramientas y tecnologías como LyCAN v1.0, ExtJS v3.4 y Symfony v2. El desarrollo se basó en Open UP como metodología guía de todo el proceso y una vez obtenido el resultado, se validó el mismo a través de pruebas que permitieron comprobar que cada funcionalidad se ejecuta correctamente. Se realizaron, además, pruebas de integración que posibilitaron confirmar la correcta unificación del componente de graficado con el resto del sistema GDR v2.0, quedando así listo para su empleo en el módulo Diseñador de Reportes. También, en compañía de los interesados, se hizo la prueba de aceptación quedando la Carta de Aceptación como muestra de su conformidad con el componente de graficado.

Palabras clave: reportes, componente, gráficas.

Abstract

In the Data Technologies Center, of the University of Informatics Sciences, it is being developed the version 2.0 of the Dynamic Reports Generator (GDR v2.0). One of its modules is the Reports Designer, which includes elements like text fields, lines, pictures, amongst others, that are used in the reports design, but it doesn't include any component for the graphical representation of the data that these ones contain. Thus the need to implement a component for transmitting an overall idea about the main aspects of the data through graphics, as to increase the visual enrichment and to facilitate the interpretation of the reports arises. The present job answers to this need through the analysis of the proposed solution and its implementation by utilizing several tools and technologies like LyCAN v1.0, ExtJS v3.4 and Symfony v2. The development was based on Open UP as methodology to guide the process and once the result was obtained, it was validated through functional tests that allowed checking that each functionality performs correctly. Also, integration tests were conducted to verify the correct unification of the graphing component with the rest of the GDR v2.0, resulting thus ready for its use in the Reports Designer module. In addition, the acceptance test was performed in the presence of the interested people, laying the Acceptance Letter as evidence of their conformity with the graphing component.

Keywords: reporting, component, graphics.

ÍNDICE DE CONTENIDO

Índice de Figuras

Índice de Tablas

Introducción

En la actualidad las organizaciones empresariales definen sus objetivos y se planifican para lograr sus metas, proceso en el cual la toma de decisiones realizada a partir de la información que se utilice es fundamental. Precisamente un recurso que favorece el apoyo a la toma de decisiones lo constituyen los reportes. Los mismos son documentos o informes que tienen como propósito facilitar el análisis de la información a diferentes niveles de profundidad o detalle en dependencia del diseño de estos, además, permiten llevar a cabo la recopilación, organización y control de los datos manejados por las entidades. Debido al creciente uso de los reportes se han creado sistemas informáticos que permiten el diseño y generación de los mismos, facilitando así la obtención de los resultados en un menor margen de tiempo (Sánchez, y otros, 2013)

Los sistemas informáticos dedicados a la generación de reportes son los que permiten mostrar los datos que están contenidos en una Base de Datos mediante una estructura organizada y sugerente. En adición a esto, constituyen una poderosa herramienta diseñada para cumplir con las cambiantes necesidades de información que demandan las empresas. La generación de reportes está centrada especialmente en los usuarios finales, es decir, en las personas que realizan el análisis de la información a través de los mismos. Esto les permite diseñar sus reportes en poco tiempo obteniendo una mejor visualización de la información gracias a los elementos (textos, imágenes, tablas, entre otros) que se les pueden incluir a los mismos (Hernández, 2003).

La generación de reportes es un campo que han desarrollado las empresas a nivel mundial. Cuba se ha hecho eco sobre este tema y justamente el empleo de reportes en apoyo a la toma de decisiones se ha incrementado con el transcurso del tiempo, llegando a la necesidad de contar con herramientas capaces de automatizar la generación de los mismos. Instituciones como la Universidad de Cienfuegos y los Joven Club de Computación han incursionado en la automatización de la generación de reportes, con el Sistema de Gestión de Reportes Dinámicos (GeReport) y el módulo Gestión de Reportes del Sistema de Reservación de Tiempo de Máquina respectivamente. También lo ha hecho la Universidad de las Ciencias Informáticas (UCI), donde se implementa, en el departamento Desarrollo de Componentes del Centro de Tecnologías de Gestión de Datos (DATEC), la versión 2.0 del Generador Dinámico de Reportes (GDR v2.0).

La versión estable y en uso del GDR, la 1.8, utiliza PHPReports como motor generador de reportes, el cual actualmente se ha tornado obsoleto por el hecho de que no recibe soporte para continuar con su funcionamiento, razón por la que se decide desarrollar una nueva versión, la 2.0. Esta versión cuenta con una serie de mejoras como la inclusión de nuevos elementos (elipses, líneas, rectángulos, tablas, saltos de línea, entre otros) dentro de los reportes. Además, el empleo de JasperReport como motor de reportes y la exportación de los reportes en los formatos PDF, XML, HTML, CSV, XLS, RTF, TXT.

Hasta el momento la versión 2.0 del GDR no cuenta con ningún elemento para la representación gráfica de los datos en los reportes, limitando así la posibilidad de que los usuarios tengan un apoyo visual que les permita de manera sencilla y sugerente interpretar la información y generalizar los datos para llegar a conclusiones sobre los comportamientos y tendencias de los mismos de manera más rápida. Por otra parte el graficado posibilitaría realizar comparaciones de indicadores y proporciones apoyando así la toma de decisiones dentro de las empresas.

Teniendo en cuenta la situación problémica referida anteriormente, se define como **problema a resolver** de la investigación **¿Cómo elaborar reportes que incluyan gráficos con el Generador Dinámico de Reportes v2.0?**

A partir del problema se define como **objeto de estudio**: La representación gráfica de datos, enmarcando la investigación en el **campo de acción**: Visualización gráfica de los datos en los reportes creados con el Generador Dinámico de Reportes v2.0.

Como **objetivo general** se plantea: Desarrollar un componente que permita la representación gráfica de los datos en los reportes generados con GDR 2.0.

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

- ¿Cuáles son los fundamentos teóricos que sustentan al proceso de desarrollo del componente de graficado en el Sistema Generador Dinámico de Reportes en su versión 2.0 para lograr que en los reportes que se generan en dicho sistema se visualicen los datos mediante gráficas?
- ¿Cuáles son las características y capacidades que debe tener el componente de graficado para que los reportes que se crean en el Generador Dinámico de Reportes v2.0 muestren la información de manera resumida, clara, precisa y agreguen valor analítico y estadístico a los reportes?

- ¿Cómo estructurar el proceso de desarrollo del componente de graficado para lograr un resultado en tiempo y con la calidad requerida?

- ¿Cómo evaluar el correcto funcionamiento del componente de graficado en el módulo Diseñador de Reportes?

Para dar cumplimiento al objetivo general se realizaron las siguientes **tareas de la investigación**:

- Análisis de los conceptos fundamentales referentes a los sistemas generadores dinámicos de reportes, haciendo énfasis en el componente de graficado de cada uno, para la elaboración del marco teórico que sostendrá los elementos necesarios que guiarán la investigación.

- Análisis de sistemas generadores de reportes ya existentes que contengan el componente de graficado para tener una visión de la solución al problema planteado.

- Determinación de la metodología, herramientas y tecnologías necesarias para el desarrollo del componente de graficado del Generador Dinámico de Reportes v2.0.

- Captura de los requisitos funcionales y no funcionales que debe cumplir el componente de graficado, para definir las características y funcionalidades que debe tener el mismo.

- Análisis y diseño de la propuesta de solución que defina la implementación del componente de graficado para el Generador Dinámico de Reportes v2.0.

- Implementación de la solución propuesta para obtener el componente de graficado.

- Comprobación del componente implementado para verificar su correcto funcionamiento.

Con el fin de apoyar la presente investigación se utilizaron varios métodos de investigación científica, siendo estos:

Métodos Teóricos:

Analítico-Sintético: método que se empleó para realizar el análisis de las bibliografías seleccionadas para desarrollar el fundamento teórico de la investigación. El mismo está relacionado con la implementación de un componente de graficado para el sistema Generador Dinámico de Reportes versión 2.0 que permita mostrar los datos mediante gráficas en los reportes que genera dicho sistema.

Modelación: método que se aplicó para la modelación de los distintos procesos, artefactos y eventos que se realizan en la construcción del componente de graficado para el GDR v 2.0.

Método Empíricos:

Entrevista: se realizaron entrevistas a los desarrolladores del sistema GDR v2.0 para capturar las principales deficiencias que dan origen a la situación problemática existente, definiendo los requisitos funcionales y no funcionales que debe tener el componente de graficado que se va a desarrollar para dicho sistema (ver Anexo 1).

Para obtener conocimientos acerca de las particularidades que tendrá el componente de graficado se aplicará la técnica de recopilación de información siguiente:

Tormenta de ideas: Se realizaron encuentros con los tutores de la investigación, el jefe a cargo del proyecto y los desarrolladores del GDR v2.0. Con estas reuniones se definirán las características y funcionalidades que debe cumplir el componente de graficado para ser integrado al módulo “Diseñador de Reportes” de la versión 2.0 del Generador Dinámico de Reportes.

El presente trabajo de diploma se encuentra estructurado en tres capítulos que están guiados por los siguientes temas a tratar:

Capítulo I. Fundamentación teórica

En este capítulo se plasman los principales conceptos, sobre la generación dinámica de reportes, que guiaron la investigación. Se exponen algunas formas empleadas para la representación de datos por medio de gráficos. Además se analizan algunas de las soluciones existentes lo cual permitió decidir si contribuyen o no a la solución propuesta. Finalmente se describen la metodología, las herramientas y las tecnologías utilizadas para el desarrollo del componente de graficado siguiendo la línea base del proyecto en desarrollo GDR v2.0.

Capítulo II: Presentación y características de la solución propuesta

En este capítulo se muestran las características generales de la solución propuesta, donde se modela el dominio del problema y se realiza una descripción general de la solución. También se describen los requisitos funcionales y no funcionales detectados durante la captura de requisitos y el modelo de Casos de Usos del Sistema con su respectiva descripción por cada caso uso. Se define la arquitectura que se emplea para el desarrollo del componente, derivándose los patrones de diseño que se utilizan en la implementación. Con todo esto se estructura el modelo de clases del diseño, donde se muestran los objetos y clases que contiene el componente.

Capítulo III. Implementación y prueba

En este capítulo se especifican los estándares de codificación que se utilizaron en la implementación del componente de graficado. Se describe el plan de estrategia de pruebas (pruebas funcionales, pruebas de integración y pruebas de aceptación) para verificar el funcionamiento y la calidad del mismo.

Capítulo I: Fundamentación Teórica

En este capítulo se enuncian los principales aspectos teóricos sobre la generación dinámica de reportes, donde se incluyen los conceptos de reporte, componente, gráfico, generadores de reportes. Además se analizan algunas soluciones existentes en el ámbito internacional y nacional para decidir si contribuyen de apoyo a la solución propuesta. Finalmente se describen la metodología, las herramientas y las tecnologías que se utilizaron para el desarrollo del componente siguiendo la línea base del proyecto en desarrollo GDR V2.0.

1.1 Fundamentaciones relacionadas a la solución del problema

Los primeros pasos de la investigación están encaminados a analizar y comprender los principales conceptos entorno al problema de la investigación, para a continuación lograr llegar a un resultado práctico a partir del conocimiento adquirido. Es por ello que a continuación se tratan los siguientes conceptos.

1.1.1 Gráfico

Es un tipo de representación de datos, generalmente numéricos, mediante recursos gráficos (líneas, vectores, superficies o símbolos), que manifiesta visualmente la relación matemática o correlación estadística que guardan entre sí. Estadísticamente los gráficos son aquellas imágenes que, combinando la utilización de sombreado, colores, puntos, líneas, símbolos, números, texto y un sistema de referencia (coordenadas), permiten presentar la información de manera cuantitativa. La utilidad de los gráficos es doble, ya que pueden servir no sólo como sustituto a las tablas sino que también constituyen por sí mismos una poderosa herramienta para el análisis de los datos, siendo en ocasiones el medio más efectivo no sólo para describir y resumir la información, sino también para analizarla (Suárez, 2011).

1.1.2 Elementos de un gráfico

Todo gráfico debe tener una serie de parámetros que faciliten la interpretación de la información. A continuación se muestran las propiedades que describen a un gráfico:

- **Área del gráfico:** Este es el sitio que se encuentra definido por el marco del gráfico y que incluye todas sus partes.
- **Título del gráfico:** Texto descriptivo del gráfico que se coloca en la parte superior.

- **Puntos de datos:** Es un símbolo dentro del gráfico (barra, área, punto, línea) que representa un solo valor.
- **Series de datos:** Son los puntos de datos relacionados entre sí trazados en un gráfico. Cada serie de datos tiene un color exclusivo. Un gráfico puede tener una o más series de datos a excepción de los gráficos circulares que solamente pueden tener una serie de datos.
- **Ejes:** Un eje es la línea que sirve como referencia de medida. El eje Y es conocido como el eje vertical y generalmente contiene datos. El eje X es conocido también como el eje horizontal y suele contener las categorías del gráfico.
- **Área de trazado:** Es el área delimitada por los ejes e incluye todas las series de datos.
- **Líneas de división:** Son líneas opcionales que extienden los valores de los ejes de manera que faciliten su lectura e interpretación.
- **Título de eje:** Texto descriptivo que se alinea automáticamente al eje correspondiente.
- **Leyenda:** Un cuadro que ayuda a identificar los colores asignados a las series de datos (Moisés, 2011).

1.1.3 Tipos de representaciones gráficas de datos

Cuando se muestran los datos estadísticos a través de representaciones gráficas, se ha de adaptar el contenido a la información visual que se pretende transmitir. A continuación se muestran varios tipos de gráficos:

1.1.3.1 Gráfico de barras

Un gráfico de barras, también conocido como gráfico de columnas, es una forma de representar gráficamente un conjunto de datos o valores y está conformado por barras rectangulares de longitudes proporcionales a los valores representados. Los gráficos de barras son usados para comparar dos o más valores. Las barras pueden orientarse verticalmente u horizontalmente (Suárez, 2011). A continuación en la Figura 1 se muestra un ejemplo de gráfico de barras.

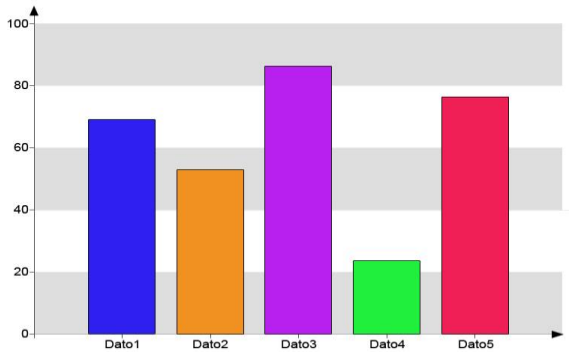

Figura 1: Gráfico de barras.

1.1.3.2 Gráfico de líneas

Los gráficos de líneas muestran una serie como un conjunto de puntos conectados mediante una sola línea. Los gráficos de líneas se usan para representar grandes cantidades de datos que tienen lugar durante un período continuo de tiempo. En este tipo de gráfico se representan los valores de los datos en dos ejes (Suárez, 2011). A continuación en la Figura 2 se muestra un ejemplo de gráfico de líneas.

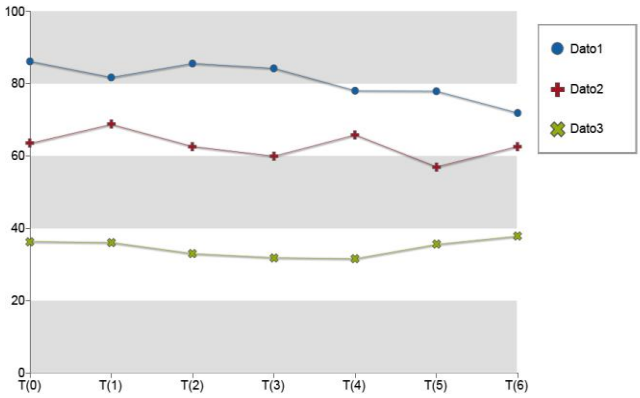

Figura 2: Gráfico de líneas.

1.1.3.3 Gráfico de pastel

Los gráficos de pastel o gráficos circulares, son recursos estadísticos que se utilizan para representar porcentajes y proporciones. El número de elementos comparados dentro de un gráfico de pastel puede ser de más de cuatro. Al igual que en el gráfico de barras, el empleo de tonalidades o colores facilita la diferenciación de los porcentajes o proporciones. A diferencia de otros tipos de gráficos, el circular no tiene ejes x o y. Se utilizan en aquellos casos donde interesa no sólo mostrar el número de veces que se da una

característica o atributo de manera tabular sino más bien de manera gráfica, de tal manera que se pueda visualizar mejor la proporción en que aparece esa característica respecto al total (Suárez, 2011). A continuación en la Figura 3 se muestra un ejemplo de gráfico de pastel.

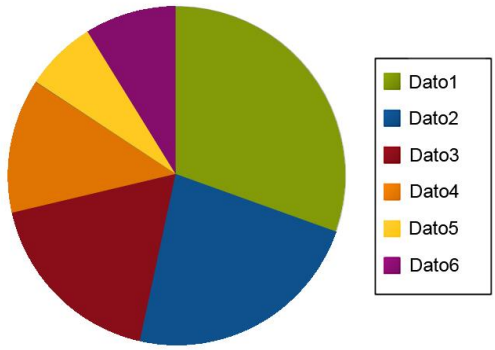

Figura 3: Gráfico de pastel.

1.1.3.4 Gráfico de áreas

Un gráfico de áreas muestra una serie como un conjunto de puntos conectados por una línea y con toda el área rellenada por debajo de la línea. En este tipo de gráfico se busca mostrar la tendencia de la información generalmente en un período de tiempo (Suárez, 2011). A continuación en la Figura 4 se muestra un ejemplo de gráfico de áreas.

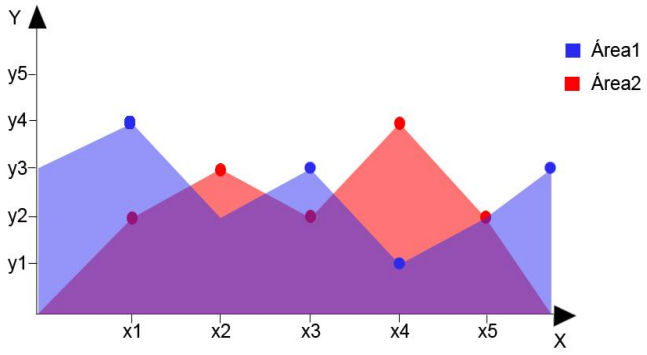

Figura 4: Gráfico de áreas.

Después de analizar las características de estos tipos de gráficos, se concluye que además de ser sencillos, sugerentes y fáciles de comprender, permiten a usuarios especializados o no, interpretar mejor la información. Estos permiten presentar de forma generalizada los números y proporciones obtenidas como resultado de un estudio, dándole mayor utilidad y mejorando la presentación de los datos de los reportes obtenidos en el GDR v2.0.

1.1.4 Componente

Dentro de los conceptos referidos a lo que es un componente se encuentra el escrito por Pressman donde específica que *"Un componente es un bloque de construcción modular para el software de ordenador"* (Pressman, 2010).

Los componentes pueblan la arquitectura del software y, por lo tanto, ayudan a cumplir con los objetivos y requisitos de sistema en construcción. Debido a que los componentes residen en el interior de la arquitectura del software, deben comunicarse y colaborar con otros componentes y con entidades (como otros sistemas, dispositivos, personas) que existen fuera de los límites del software. (Pressman, 2010).

En el contexto de la ingeniería de software orientada a objetos, un componente contiene un conjunto de clases que colaboran entre sí. Cada clase de un componente se ha elaborado completamente para incluir todos los atributos y las operaciones relevantes para su implementación. Como parte de la elaboración del diseño, también deben definirse todas las interfaces que permiten que las clases se comuniquen y colaboren con otras clases del diseño. (Pressman, 2010).

Luego de analizar las anteriores definiciones de componente y teniendo en cuenta la definición de gráfico se concluye que el graficado tiene como principal función la de presentar la información de manera cuantitativa mediante recursos gráficos, dándole un valor agregado a los reportes obtenidos en el GDR v2.0.

1.1.5 Reporte

Un reporte es un informe o una noticia que puede ser impresa, digital o audiovisual, pretende transmitir una información, aunque puede tener diversos objetivos. Existen reportes divulgativos que son aquellos cuyo objetivo es que su contenido llegue a conocimiento de varias personas. También existen reportes persuasivos que son los que pretenden que con argumentos o elementos suficientes una o varias personas actúen o piensen de un modo determinado. Específicamente en el ámbito de la Informática los reportes son informes que organizan y exhiben la información contenida en una base de datos. Su función

es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y fácil de interpretar por los usuarios. Los reportes tienen diversos niveles de complejidad, desde una lista o enumeración hasta gráficos mucho más desarrollados (Sánchez, y otros, 2013).

Teniendo en cuenta que el concepto anterior aborda de una manera muy amplia las características fundamentales de un reporte, se decide adoptar el mismo durante la investigación. Tratándose del GDR v2.0, en dependencia del entorno donde se emplee, puede ser utilizado para generar reportes estadísticos, analíticos, técnicos e incluso policiales.

1.1.6 Sistemas generadores de reportes

Los generadores de reportes son herramientas complementarias de los sistemas de información, que utilizan una especie de lenguaje transparente para el usuario, por medio del cual se realizan consultas a la base de datos, obteniendo la información a mostrar en los reportes. Además, tienen la capacidad de interactuar con los resultados obtenidos basándose en los datos para tomar sus propias decisiones (Silva Hernández, 2003).

Los generadores de reportes están compuestos principalmente por dos elementos básicos, un diseñador o editor de informes y un motor de generación (Marrero, 2009).

Luego de analizar esta definición se llega a la conclusión de que los generadores de reportes permiten mostrar mediante documentos bien estructurados la información de cualquier entidad que esté contenida en una Base de Datos.

1.1.7 Generador Dinámico de Reportes (GDR)

El GDR es una aplicación web que tiene como objetivo generar reportes de forma rápida, interactiva y con una amplia gama de alternativas para los usuarios. Permite a sus clientes consultar las bases de datos de sus organizaciones y poder generar reportes con la información que estos manejan, independientemente del SGBD que utilicen ya sea MySQL, SQLite o PostgreSQL. El desarrollo de dicho sistema está basado en tecnología web, lo cual hace del sistema una gran ventaja para las organizaciones puesto que lo mantiene disponible desde cualquier estación de trabajo, las 24 horas del día. Tiene una interfaz gráfica amigable y está orientado al usuario final. Se encuentra en constante perfeccionamiento con el objetivo de que cada día, logre satisfacer las expectativas del mercado o del cliente. Durante su implementación se emplearon tecnologías novedosas en el desarrollo web actual como son: PHP, JavaScript, Symfony, Ext-

JS, XML, PostgreSQL. Es una aplicación multiplataforma y soporta imágenes, gráficas y varios orígenes de datos (Rodríguez, 2011).

1.2 Análisis de soluciones existentes de Generadores de Reportes.

En la actualidad existe en el mundo una gran cantidad de sistemas informáticos empleados en la generación de reportes. Estas son herramientas complementarias de los sistemas de información que utilizan una especie de lenguaje transparente para el usuario por medio del cual este realiza consultas a la base de datos y obtiene información de ella en forma de reporte.

A continuación se detallan algunas características de las herramientas para la generación dinámica de reportes Crystal Reports, Generador Dinámico de Reportes 1.8 (GDR v1.8), iReport y Eclipse BIRT.

Crystal Reports

Crystal Reports es un sistema comprobado por muchos expertos en el diseño de reportes que cumplan los desafíos que en el día a día enfrentan los analistas de negocios y los desarrolladores por su alta tecnología de elaboración de informes. Es una aplicación de inteligencia empresarial utilizada para diseñar y generar informes desde una amplia gama de fuentes de datos. Además cuenta con varias funcionalidades, entre ellas las opciones de formato y componentes, donde estos últimos se catalogan en mapas, tablas cruzadas, imágenes, gráficos, diagramas, entre otros. Varias aplicaciones, como Microsoft Visual Studio, incluyen una versión de Crystal Reports como una herramienta de propósito general para reportes con datos. El componente gráfico de Crystal Reports tiene entre sus tipos principales de gráficas a la de: barra, línea, área, gráfico circular, gráfico de radar, Gantt, gráfico de burbujas, gráfico de anillos, entre otras (CrystalReports, 2003).

IReport

La herramienta iReport es un constructor y diseñador de informes visual, potente, intuitivo y muy fácil de usar. Permite que los usuarios corrijan visualmente informes complejos con cartas, imágenes, subinformes, gráficas, tablas, entre otros. El iReport está integrado con JFreeChart, una de las bibliotecas gráficas OpenSource más extendida para Java. Además utiliza la biblioteca Jasper Reports inherentemente para crear informes de código abierto más populares para la tecnología Java. También es un diseñador visual para el diseño de informes. Nació como una herramienta de desarrollo pero puede utilizarse como una herramienta de oficina para adquirir datos almacenados en una base de datos, sin pasar a través de

alguna otra aplicación. Dentro de las principales gráficas del IReport están: de barras, de líneas, de pastel, de áreas (Laterza, 2011).

Eclipse BIRT

Eclipse BIRT, herramienta para el reporte y la inteligencia de negocio es un proyecto de software de código abierto que proporciona capacidades de creación de informes para aplicaciones web, especialmente aquellas basadas en Java y Java EE. Sus diseños de informes se hacen en XML. Está conformado por dos componentes principales: un diseñador de informes visuales dentro de Eclipse IDE para crear informes BIRT, y un componente de rutina para generar informes que pueden ser puestos en uso en cualquier entorno Java. El proyecto BIRT también incluye un motor de gráficos que está integrado en el diseñador de informes y además puede ser usado por separado para incluir gráficas en una aplicación. Los tipos de gráficos a insertar pueden ser: de barra, de línea, de área, de pastel, de metro, de dispersión, de tubo, de burbujas, de cono, Gantt, entre otras (Sánchez, y otros, 2013).

Generador Dinámico de Reportes 1.8 (GDR v1.8)

El GDR 1.8 es una herramienta que brinda la posibilidad de controlar el funcionamiento periódico de una o varias entidades, mediante el diseño de cualquier tipo de reportes, incluyendo los tabulares (con gráficos incluidos), tabla pivote y cruzada, desde los gestores de bases de datos: SQL Server, SQLite, Oracle 10g, MySQL y PostgreSQL. Proporciona a los usuarios la ventaja de agilizar el proceso de la toma de decisiones.

Características:

- Aplicación desarrollada sobre el marco de trabajo Symfony 1.1.7.
- Escrita en el lenguaje de desarrollo PHP.
- Multiplataforma.
- Soporta imágenes y gráficas.
- Soporta varios orígenes de datos.

La versión 1.8 del GDR cubre el ciclo básico de la generación de reportes y soluciona el problema de obtener los diferentes informes en los sistemas de gestión de la información que se desarrollan en cualquier entorno empresarial, incluyendo la UCI. Dentro de los tipos de gráficas que maneja se encuentran: de barra, de pastel, de línea y de curva (Bouly, 2013).

A continuación se muestra en la Tabla 1 **¡Error! No se encuentra el origen de la referencia.** Una comparación de las herramientas anteriormente analizadas para mostrar la presencia del componente de graficado en las mismas.

Tabla 1: Comparación de Generadores de Reportes en cuanto a elementos gráficos.

Sistemas generadores de reportes	Gráficos	Tipos de gráficos
Crystal Reports	Si	Barra, línea, área, circular, radar, Gantt, burbujas, anillos, corona circular, proyección 3D, superficie 3D, dispersión XY, bursátil, eje numérico, esfera, embudo, histograma.
iReport	Si	Barras, líneas, pastel, áreas, circular, radar, Gantt, burbujas, anillos, corona circular, proyección 3D, superficie 3D, dispersión XY, bursátil, eje numérico, esfera, embudo, histograma.
Generador Dinámico de Reportes 1.8	Si	Barra, pastel, línea, curva.
Eclipse BIRT	Si	Barra, línea, área, pastel, metro, dispersión, tubo, burbujas, cono, Gantt, proyección 3D, dispersión XY.

El análisis realizado a los diferentes generadores de reportes en cuanto a la incorporación de gráficos en los informes demostró que el componente de graficado es de gran utilidad para este tipo de sistemas ya que brindan la posibilidad de realizar mejores análisis, más profundos y certeros. Además el análisis realizado a los tipos de gráficos que poseen indicó que los más comunes y más utilizados son las gráficas de barra, pastel, área y línea. Por tanto, en el presente caso, se decide incorporar estos elementos de graficado a los reportes generados por el GDRv2.0, facilitando así una mejor interpretación de la información.

1.3 Metodología, herramientas y tecnologías

Por cuestiones de continuar con la línea de trabajo que se determinó para el desarrollo de la versión 2.0 del GDR y de acuerdo a la decisión tomada por el equipo de desarrollo del proyecto en cuanto al empleo de la metodología, las herramientas y las tecnologías, se decide alinearse a las mismas para el desarrollo del componente en cuestión. A continuación se describen cada una de ellas destacando las principales características y ventajas que brindan para el desarrollo del componente.

1.3.1 Metodología Open Up

Open UP[1] es un proceso unificado ágil y liviano, que aplica un enfoque iterativo e incremental dentro de un ciclo de vida estructurado y contiene un conjunto mínimo de prácticas que ayuda al equipo a ser más efectivo desarrollando software. Open UP abarca una filosofía pragmática y ágil de desarrollo, que se enfoca en la naturaleza colaborativa del desarrollo de software. Se trata de un proceso

Mantiene las características esenciales de RUP, en el cual se incluyen las siguientes características:

- Desarrollo incremental.
- Uso de casos de uso y escenarios.
- Manejo de riesgos.
- Diseño basado en la arquitectura.

Open UP es para equipos pequeños que trabajan juntos en la misma ubicación. Los equipos necesitan involucrarse en una interacción plena cara a cara diariamente. Los miembros del equipo incluyen a los stakeholders (interesados), desarrolladores, arquitectos y administradores de proyecto. Ellos toman sus propias decisiones sobre en qué necesitan trabajar, cuáles son las prioridades y cómo alcanzar las necesidades del stakeholder de la mejor manera.

Los miembros del equipo trabajan colaborativamente. La presencia de los stakeholders como miembros del equipo es crítico para la implementación exitosa de Open UP. Además, los miembros del equipo participan en reuniones para comunicar los estados y situaciones. Los problemas se resuelven fuera de estas reuniones, similar a Scrum. Open UP se enfoca en reducir el riesgo de manera temprana en el ciclo de desarrollo. Esto requiere que las reuniones sean regulares para revisar los riesgos y una implementación rigurosa de estrategias de mitigación (Gimson, 2012).

Open UP está gobernada por cuatro principios fundamentales. A continuación se muestran en la

Tabla 2 :

1 Open Unified Process o Proceso Unificado Abierto.

Tabla 2: Descripción de los principios de la metodología Open UP.

Principios de Open UP	**Descripción**
Colaborar para alinear los intereses y compartir entendimiento.	Al individuo y a las interacciones del equipo de desarrollo por encima del proceso y las herramientas
Balancear las prioridades involucradas para maximizar el valor para el stakeholder.	Colaboración con el cliente por sobre la negociación contractual
Enfocarse en la arquitectura tempranamente para minimizar riesgos y organizar el desarrollo.	Software que funcione por sobre una documentación exhaustiva
Evolucionar para obtener constantemente retroalimentación y mejorar.	Respuesta al cambio por sobre el seguimiento de un plan.

La metodología Open Up guía el proceso de desarrollo de software mediante las tareas que se muestran a continuación en la Tabla 3:

Tabla 3: Tareas de Open Up.

Disciplinas	**Tareas**
Arquitectura	•Refinar la arquitectura •Visualizar (Prever) la arquitectura
Desarrollo	•Implementar las pruebas de desarrollo •Implementar la solución •Correr las pruebas de desarrollo •Integrar y crear el desarrollo •Diseñar la solución
Administración de Proyecto	•Evaluar resultados •Administrar La iteración •Planear la iteración •Planear el proyecto •Solicitar cambios

Requerimientos	•Identificar y resaltar requerimientos •Detallar Escenarios de Caso de Uso •Detallar requerimientos generales del sistema •Desarrollar una visión técnica
Prueba	•Crear Casos de Prueba •Implementar pruebas •Correr pruebas

En el proceso de desarrollo del componente de graficado se decide continuar con el empleo de la metodología Open Up para seguir la línea base del proyecto GDR v2.0 y no crear conflictos con los artefactos generados.

1.3.2 Herramientas y tecnologías

Una herramienta de desarrollo de software es un programa informático que usa un programador para crear, depurar, gestionar o mantener un programa. Por otra parte las tecnologías estándares de la web pueden ser vistas como una plataforma de aplicaciones que utilizan el motor del navegador para mostrar interfaces de usuario e interpretar código y usar protocolos web para comunicarse con un servidor. Para el caso del desarrollo del componente gráfico se emplearán las siguientes herramientas y tecnologías:

1.3.3 Lenguajes

Un lenguaje de programación es un conjunto de caracteres, reglas para combinarlos y especificar sus efectos al ejecutarlas en una computadora, según el autor (Ralston, y otros, 2000) tiene las siguientes características:

1. No se requiere de conocimientos acerca del lenguaje de máquina por parte del usuario.
2. Es independiente de la máquina.
3. Es traducido al lenguaje de máquina.
4. Emplea una notación que se acerca más a la del problema en resolución que el lenguaje de máquina.

1.3.3.1 Lenguajes de Marcado

XML

El Lenguaje de marcas extensible (XML) es usado para estructurar información en un documento o en general en cualquier fichero que contenga texto, como por ejemplo ficheros de configuración de un programa o una tabla de datos.

Comparado con otros sistemas usados para crear documentos, el XML tiene la ventaja de poder ser más exigente en cuanto a la organización del documento, lo cual resulta en documentos mejor estructurados (Montalvo, 2010).

XML se emplea para el desarrollo del componente de graficado. Teniendo en cuenta que a diferencia de otros lenguajes, XML da soporte a bases de datos y permite almacenar datos en forma legible, este permitirá almacenar las configuraciones de las propiedades de los gráficos.

1.3.3.2 Lenguajes de Programación

PHP5.3

PHP es un lenguaje de script del lado del servidor. Los scripts PHP están incrustados en los documentos HTML y el servidor los interpreta y ejecuta antes de servir las páginas al cliente. El cliente no ve el código PHP sino los resultados que produce.

Según el autor (php, 2015) tiene las siguientes características:

- Rápido.
- Lenguaje fácil de aprender y potente.
- Integración perfecta con diversos servidores HTTP.
- Acceso a diversos tipos de Bases de Datos.
- Diseño modular de fácil ampliación.
- Licencia abierta.

Dentro de las nuevas funcionalidades que maneja el PHP 5.3 se encuentran:

- Soporte para espacios de nombres.
- Soporte para Enlaces Estáticos en Tiempo de ejecución.
- Soporte para etiquetas de saltos.

Se utiliza para el desarrollo del componente porque es un lenguaje muy sencillo de aprender. Además de que es totalmente libre y abierto. También los entornos de desarrollo que utiliza son de rápida y fácil configuración y facilita además el acceso a bases de datos.

Java Script

Java Script es un lenguaje de programación que se utiliza principalmente del lado del cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas. Técnicamente Java Script es un lenguaje de programación interpretado por lo que no es necesario compilar el código para ejecutarlos, esto permite que los programas se puedan probar directamente en cualquier navegador sin necesidad de procesos intermedios.

Para el desarrollo del componente de graficado se utiliza Java Script porque es un lenguaje muy simple que se puede aprender de manera rápida. Además es un lenguaje orientado a objetos, o sea que soporta las características principales del paradigma de la programación orientado a objetos (encapsulación, herencia y polimorfismo). También porque se define como un lenguaje seguro puesto que se implementaron barreras de seguridad en este y en el sistema de ejecución en tiempo real. Por otra parte está diseñado para soportar aplicaciones que serán ejecutadas en los más variados entornos de red, estaciones de trabajo y sobre arquitecturas distintas o con sistemas operativos diversos, al igual que PHP5.3 es un lenguaje multiplataforma que funciona en cualquier sistema operativo que tenga instalada la máquina virtual de Java (Javier Eguiluz, 2015).

Por todas estas razones el sistema Generador Dinámico de Reportes v2.0, emplea en su desarrollo el lenguaje java Script, y se seguirá utilizando para la implementación del componente de graficado.

1.3.4 Marcos de trabajo

Un framework o marco de trabajo es una aplicación reutilizable, semi-completa que puede ser especializada para producir aplicaciones concretas y específicas. El framework describe los objetos que componen el sistema, cómo éstos interactúan, y cuáles son sus responsabilidades (Martínez, y otros, 2013).

Es necesario utilizar marcos de trabajo para el desarrollo de la solución porque permiten simplificar la implementación del mismo mediante la reutilización de algunos de los componentes que están ya predefinidos. Además, el framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, facilita la implementación del componente, ya que encapsula operaciones complejas en instrucciones sencillas. A continuación se describen los marcos de trabajo que se emplearán en la implementación del componente de graficado.

ExtJS v3.4

ExtJs es un conjunto de librerías JavaScript que permite el desarrollo de aplicaciones RIA (Aplicaciones enriquecidas en Internet) basadas en un navegador, sin embargo se integra mediante plug-ins[2] con Adobe AIR para generar aplicaciones RIA basadas en escritorio.

Al desarrollador ExtJs ofrece un gran conjunto de widgets (grids[3], ventanas de diálogo, gráficas, entre otros) plenamente integrados y un API (Interfaz de programación de aplicaciones) para obtener interfaces web más dinámicas e interactivas con el usuario (Lozano, 2010).

El marco de trabajo ExtJS se utilizará como herramienta para el desarrollo del componente de gráfica, ya que después de analizada la definición anterior se concluye en que este nos permite crear aplicaciones complejas utilizando componentes que están predefinidos, y precisamente uno de ellos es el componente gráfico.

Symfony v2.0.18

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Contiene muchas particularidades que lo hacen propicio para el desarrollo del componente de graficado como: separa la lógica de negocio, la lógica del servidor y la presentación de la aplicación web. Además proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Es capaz de automatizar las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. Agregar además que Symfony está desarrollado completamente con PHP 5 y es compatible con la mayoría de gestores de bases de datos. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, entre otros) como en plataformas Windows (Eguiluz, 2015).

La utilización de esta tecnología propició que la aplicación se favoreciera de sus principales ventajas:

- Versátil: ya que se ajusta al desarrollo de grandes, medianas y pequeñas aplicaciones. Su estructura de componentes y el uso de algunas librerías externas como doctrine se unen para formar los independientes y reutilizables bundles.
- Útil: desde la perspectiva del trabajo con elementos importantes en el trabajo de aplicaciones web

[2] Un **plug-in** es un módulo de hardware o software que añade una característica o un servicio específico a un sistema más grande.

3 Función que divide un área de trabajo en cuadrículas. Suele estar disponible en las aplicaciones de diseño.

tales como la persistencia de datos, validación, internacionalización y la importante seguridad de las mismas.

- Buenas prácticas: basadas en la mayoría de excelentes marco de trabajo como Spring, Django, y Rails de los cuales incluye elementos fundamentales como la configuración de la seguridad de acceso a usuarios y el uso de plantillas para la presentación del contenido.
- Flexible: para los diferentes tipos de desarrolladores con específicos niveles de conocimientos en el lenguaje.
- Rendimiento: con excelentes resultados ya que por medio del mecanismo de cache convierte todos los ficheros de configuración en código ejecutable PHP y lo almacena en cache para mejorar el rendimiento, acelerando los tiempos de respuesta de la aplicación.
- Documentación: aunque todavía pequeña cuenta con buenos libros donde de forma explicativa se aborda las principales elementos de trabajo de este framework y además todo ello de forma gratuita. (Rodríguez, 2011).

LyCAN v1.0

Lacan constituye un incipiente desarrollo tecnológico perteneciente al departamento de DATEC. Está basado en la librería de JavaScript ExtJS en su versión 3.4.0. En Lacan se encuentran agrupados una serie de mecanismos y comportamientos encapsulados que indudablemente fueron previamente analizados por los arquitectos de software para decidir y apostar por esta herramienta. El uso de este marco de trabajo, además de aprovechar ventajas propias de ExtJS tales como la validación de sus formularios, el uso de interfaces atractivas y con ricos componentes para la web, incorpora nuevas características como:

- **Área de trabajo:** se hace una óptima organización de los componentes para aprovechar al máximo el área de trabajo. Esto permite que se definen los diferentes módulos como elementos que se registran en forma de pestañas en la aplicación. Además se le define una pequeña y minimizarle barra de herramientas en forma de listón y una extensa área de trabajo para facilitar la gestión al usuario.
- **Estructura de CRUD:** propone un conjunto de interfaces concebidas para las entidades del negocio donde se realizan las operaciones elementales de un CRUD, entiéndase por crear, listar, modificar, eliminar y buscar.

- **Uso de extensiones para CRUD:** define además las interfaces convencionales para gestionar esta operación. Otras opciones como menuces contextuales a través de la opción con el clic secundario y en la barra inferior del módulo, que realizan de igual manera las funcionalidades del CRUD.
- **Validación de formularios:** Contiene personalizaciones importantes donde se validan los campos de forma tal que resulta imposible terminar una operación hasta cuanto no se haya llenado correctamente la información que requiere la interfaz.
- **Eventos y comportamientos:** Estos acompañan a cada una de las interfaces y de acorde a los principales escenarios donde se involucran. Es importante destacar que la comunicación entre la vista y los controladores de Symfony se realiza a través del mecanismo de Llamada a Procedimientos Remotos, por sus siglas en inglés RPC. Esto permite ejecutar funcionalidades en otra máquina de forma remota sin preocuparse por la comunicación entre ambos (Guerra, y otros, 2014).

1.3.5 Entorno de desarrollo

Un Entorno de Desarrollo Integrado (IDE), es una aplicación de software, que proporciona servicios integrales para facilitarle al programador de computadora el desarrollo de software.

A continuación se describe el entorno de desarrollo que se utilizara para la implementación del componente de graficado:

NetBIOS v7.3

El IDE NetBIOS es de código abierto y escrito en el lenguaje de programación Java. Provee servicios comunes para la creación de aplicaciones de escritorio tales como gestión de ventanas y menús, almacenamiento de configuraciones. También es el primer IDE en soportar completamente las características de JDK 5. La plataforma NetBIOS y el IDE son libres para uso comercial y no comercial (Oracle, 2015).

1.3.6 Motor de reportes

JasperReport

JasperReport es una herramienta gratuita y de código abierto que se compone de un conjunto de bibliotecas java para facilitar la generación de reportes en aplicaciones web y de escritorio. Los reportes se

definen en un fichero JRXML el cuál será compilado por las bibliotecas JasperReport y generarán un fichero Jasper que se utiliza para conformar el reporte final. Actualmente estos reportes se generan en varios formatos: PDF, CSV, XML, TXT, HTML, XLS, RTF y Jasper Viese. La definición de los reportes se puede realizar directamente en los XML o con la utilización de los restantes módulos del GDR para el diseño gráfico de los mismos.

La utilización de esta biblioteca permite exportar en varios formatos los reportes diseñados, realizar gráficas desarrolladas e integradas y brinda la posibilidad de crear sub-reportes. El manejo de la información dentro de la aplicación pasa por varios procesos. En un inicio se utilizan varios de los módulos implementados para la obtención de datos y para el diseño de los reportes que posteriormente se generan utilizando el motor de reportes para GDR v2.0 que es una aplicación java que utiliza las bibliotecas de JasperReport con el objetivo de generar reportes.

Para lograr el éxito del motor de reportes del GDR v2.0 se pensó en una aplicación ligera, portable, rápida y de alto rendimiento. Esta aplicación java utiliza para la comunicación con el resto del sistema el protocolo HTTPS[4] para la seguridad de los datos y envía la información mediante el método POST (Rodríguez, y otros, 2010).

1.3.7 Herramienta de modelado

Visual Paradigma v8.0

Es una herramienta de ingeniería de software multiplataforma, que propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, generación del código fuente de los programas y la documentación de los mismos. Su propósito general es soportar el ciclo de vida del proceso de desarrollo del software a través de la representación de todo tipo de diagramas.

Esta herramienta genera diseños centrado en casos de uso y enfocados al negocio contribuyendo así a un software de mayor calidad. Usa un lenguaje estándar y común a todo el equipo de desarrollo que facilita la comunicación. Mantiene capacidades de ingeniería directa e inversa y posibilita que el modelo y el código permanezcan sincronizados en todo el ciclo de desarrollo (Paradigma, 2014).

[4] HTTPS: protocolo de transferencia de hipertexto.

Esta herramienta va a permitir diseñar distintos tipos de diagramas que van a modelar toda la lógica del negocio. Por tanto se emplea esta herramienta para el desarrollo del componente de graficado.

Conclusiones del capítulo

Partiendo del estudio teórico que involucró conceptos como sistemas generadores de reportes, reporte, gráfico, tipos de representación gráfica de datos y componente, para entender posteriormente la solución propuesta.

Por otra parte, el estudio de algunas soluciones de software empleadas para la generación de reportes arrojó que los gráficos son de gran utilidad e importancia dentro de los sistemas generadores de reportes, ya que permiten incorporar gráficas a los reportes y con ellas la posibilidad de una mejor interpretación de la información que se encuentra reflejada de forma gráfica.

Por último, aunque por cuestiones de continuidad de la línea de trabajo establecida para el desarrollo de la versión 2.0 del GDR se decide reutilizar la metodología, las herramientas y las tecnologías para el desarrollo del componente en cuestión, el estudio realizado acerca de las mismas permitió constatar que son adecuadas y suficientes para obtener el resultado práctico de la investigación, es decir, un componente de graficado capaz de incluir gráficas en los reportes generados con el GDR. Además, cumplen con las tendencias actuales de la programación web y están enfocadas al código abierto, lo cual posibilita la obtención del componente de forma económica, aspecto esencial para el desarrollo de software en Cuba.

Capítulo II: Presentación y características de la solución propuesta.

En este capítulo se muestra una descripción de los flujos de trabajo propuestos por la metodología Open UP, el modelado de dominio de la solución propuesta, así como la captura de requisitos funcionales y no funcionales que debe cumplir el componente a desarrollar. A partir de estos se modela el componente quedando representado en un diagrama de casos de uso y sus descripciones textuales. Además se modelan algunos de los artefactos definidos por la metodología Open UP como clases del diseño y modelo de datos.

2.1 Modelo del Dominio

Un modelo de dominio presenta uno o más diagramas de clases que no contienen conceptos propios de un sistema de software sino de la propia realidad física. El modelo de dominio captura los tipos más importantes de objetos que existen, o los eventos que suceden en el entorno donde estará el sistema, se identifican conceptos, se definen estos conceptos y se unen o relacionan en un diagrama de clases UML (Grand, 2010).

El modelo de dominio que se presenta a continuación permite entender de mejor manera el área del negocio en el cual está enmarcado el componente a desarrollar. En este se identifican las relaciones que existen entre las entidades comprendidas en el ámbito del problema y los atributos contenidos por cada una de ellas. Las entidades y atributos son representados mediante un diagrama de clases y se define un vocabulario donde se describe cada uno de los conceptos asociados al problema el cual es utilizado de ayuda como herramienta de comunicación. A continuación en la Figura 5 se muestra el diagrama del modelo de dominio.

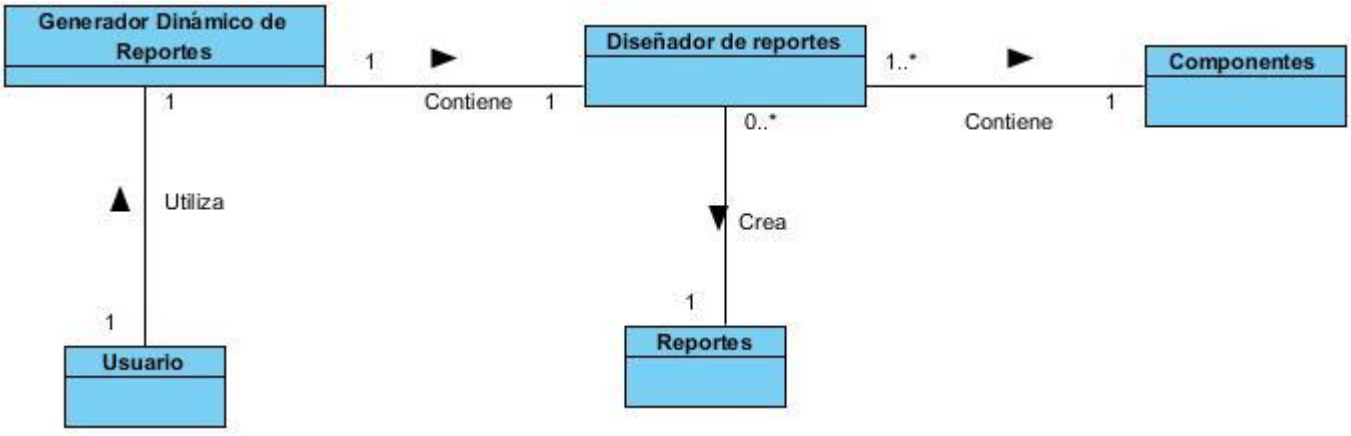

Figura 5: Diagrama del Modelo del Dominio.

2.1.1 Descripción de las clases del dominio

- **Generador Dinámico de Reportes:** Sistema informático que permite generar reportes para el análisis de la información.
- **Diseñador de Reportes:** Módulo del GDR v2.0 que permite diseñar los reportes que luego son generados.
- **Reporte:** Informe o documento que pretende transmitir una información mediante elementos como: textos, tablas, diagramas, imágenes, figuras y gráficas.
- **Componentes:** Es una parte modular o elemento del sistema GDR v2.0 que ofrece un conjunto de servicios y funcionalidades a través de interfaces definidas. Donde se encuentran los componentes que se le incluyen a los reportes como: texto, tablas, línea, elipse, salto de línea y rectángulo.

2.1.2 Especificación de los requisitos del sistema

Los requisitos funcionales de un sistema describen lo que el sistema debe hacer. Estos requerimientos dependen del tipo de software que se desarrolle, de los posibles usuarios del software y del enfoque general tomado por la organización al redactar dichos requerimientos (Somerville, 2005).

A continuación se muestran los requisitos funcionales y no funcionales que describen al componente de graficado, así como sus propiedades y las restricciones asociadas a su funcionamiento.

2.1.3 Requisitos funcionales

- **RF1 Añadir nueva gráfica de barras en el reporte.**

Descripción: Esta funcionalidad permite al usuario insertar una gráfica de barras en un reporte.

Entrada: Componente de gráfico.

Salida: Componente gráfico de barras pre-configurado por la ventana de configuración de propiedades.

- **RF2 Añadir nueva gráfica de líneas en el reporte.**

Descripción: Esta funcionalidad permite al usuario insertar una gráfica de líneas en un reporte.

Entrada: Componente gráfico.

Salida: Componente gráfico de líneas pre-configurado por la ventana de configuración de propiedades.

- **RF3 Añadir nueva gráfica de pastel en el reporte.**

Descripción: Esta funcionalidad permite al usuario insertar una gráfica de pastel en un reporte.

Entrada: Componente de gráfico.

Salida: Componente gráfico de pastel pre-configurado por la ventana de configuración de propiedades.

- **RF4 Añadir nueva gráfica de áreas en el reporte.**

Descripción: Esta funcionalidad permite al usuario insertar una gráfica de área en un reporte.

Entrada: Componente de gráfico.

Salida: Componente gráfico de área pre-configurado por la ventana de configuración de propiedades.

- **RF5 Eliminar gráfica de barras en el reporte.**

Descripción: Esta funcionalidad permite al usuario eliminar una gráfica de barras de un reporte.

Entrada: Arreglo de componentes seleccionados.

- **RF6 Eliminar gráfica de líneas en el reporte.**

Descripción: Esta funcionalidad permite al usuario eliminar una gráfica de líneas de un reporte.

Entrada: Arreglo de componentes seleccionados.

Salida: Ventana de mensaje.

- **RF7 Eliminar gráfica de pastel en el reporte.**

Descripción: Esta funcionalidad permite al usuario eliminar una gráfica de pastel de un reporte.

Entrada: Arreglo de componentes seleccionados.

- **RF8 Eliminar gráfica de áreas en el reporte.**

Descripción: Esta funcionalidad permite al usuario eliminar una gráfica de áreas de un reporte.

Entrada: Arreglo de componentes seleccionados.

- **RF9 Modificar gráfica de barras en el reporte.**

Descripción: Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de barras en cuanto a: color, tamaño, valores de las coordenadas(x, y).

Entrada: Componente gráfico.

Salida: Componente gráfico con las propiedades de configuración editadas.

- **RF10 Modificar gráfica de líneas en el reporte.**

Descripción: Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de líneas en cuanto a: color, tamaño, valores de las coordenadas(x, y).

Entrada: Componente gráfico.

Salida: Componente gráfico con las propiedades de configuración editadas.

- **RF11 Modificar gráfica de pastel en el reporte.**

Descripción: Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico circular en cuanto a: color, tamaño, valores.

Entrada: Componente gráfico.

Salida: Componente gráfico con las propiedades de configuración editadas.

- **RF12 Modificar gráfica de áreas en el reporte.**

Descripción: Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de áreas en cuanto a: color, tamaño, valores de las coordenadas(x, y).

Entrada: Componente gráfico.

Salida: Componente gráfico con las propiedades de configuración editadas.

2.1.4 Requisitos no funcionales

Los requisitos no funcionales son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de esta como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento (Somerville, 2005)

A continuación se muestran los requisitos no funcionales que debe cumplir el componente de graficado:

RNF1 Software requerido para desplegar y utilizar la aplicación

- **Servidor para instalar la aplicación:** El servidor donde se instalará la aplicación debe cumplir con los siguientes requisitos de software:
    - SO: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debían 4 GNU/Linux o superior.
    - Usuario con privilegios de administración del SO.

RNF 2 Hardware requerido para desplegar y utilizar el componente

PC Cliente:

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

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

PC Servidor:

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

RNF 3 Requisitos de interfaz

El usuario deberá acceder a la aplicación a través del protocolo HTTP usando el navegador Firefox 2 o una versión superior.

- **Interfaces de usuario**

Las interfaces de usuario serán diseñadas a modo de aplicaciones RIA (Reich Internet Aplicación) lo que permite a los usuarios contar con aplicaciones web con una experiencia de usuario similar a la de las aplicaciones de escritorio. Para lograr este fin se usará la librería JavaScript Exijas, la cual conjuga una serie de componentes visuales que proveen funcionalidades, que ayudan al diseño de este tipo de aplicaciones WEB con apariencia de escritorio.

- **Interfaces Hardware**

  Por su naturaleza, el sistema no interactúa con ningún tipo de interfaz de Hardware.

- **Interfaces Software**

  El sistema interactuará con un servidor de gráficos (Chart Server) quien se encargará de mostrar todos los gráficos que se le soliciten cuando se quiera diseñar un reporte gráfico y un servidor de análisis estadísticos (R-Server) que provee análisis estadísticos variados. Además debe contar con un API como mecanismo de integración con otros sistemas, este mecanismo debe ser independiente del lenguaje de programación en el cual se encuentre la aplicación que lo utilizará.

- **Interfaces de Comunicación**

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

RNF 4 Requisitos de usabilidad

Esto permite que exista mayor facilidad de uso del componente, al evitar dar más de cuatro clic para ejecutar cualquier funcionalidad del mismo.

2.2 Modelo de Casos de Uso del Sistema

Los casos de uso son una técnica que se basa en escenarios para la obtención de requerimientos. Un caso de uso identifica el tipo de interacción y los actores involucrados (Somerville, 2005).

En la definición del modelo de casos de uso del sistema se describen los pasos o actividades que deberán realizarse para llevar a cabo un proceso de interacción entre un usuario que en este caso es el diseñador de reportes y el componente de graficado que permite incluir y configurar diferentes tipos de gráficas en los reportes. El modelo de casos de uso contiene una secuencia de interacciones que se desarrollarán entre el actor y componente de graficado en respuesta a un evento que se inicia sobre el propio componente.

2.2.1 Definición de los actores del sistema

Los actores representan un tipo de usuario del sistema, es cualquier elemento externo que interactúa con el sistema. No necesariamente tiene que ser humano, puede ser otro sistema informático. Los casos de uso a los que un actor tiene acceso definen su rol global en el sistema y el alcance de la acción. Son los responsables de realizar las actividades que serán automatizadas en el sistema. (Orozco, y otros, 2012). A continuación en la Tabla 4 se describe como interactúa el actor con el componente.

Tabla 4: Actores del sistema.

Actores	Descripción
Diseñador de reportes.	Interactúa con el componente gráfico durante la ejecución de sus funcionalidades. Visualiza las representaciones de los datos mediante gráficas que se muestran en los reportes.

2.2.2 Identificación de los Casos de Uso

Los casos de uso pueden tener relaciones con otros casos de uso. Los tres tipos de relaciones más comunes entre casos de uso son:

- **Inclusión**: Especifica una situación en la que un caso de uso tiene lugar dentro de otro caso de uso. Lo cual quiere decir que si un caso de uso (1) extiende de un caso de uso (2), el segundo es parte esencial del primero por lo que la ejecución del primer caso de uso depende de la ejecución del segundo (Ortiz, 2012).

- **Extensión:** Especifica que en ciertas situaciones, o en algún punto (llamado punto de extensión) uncu será extendido por otro. Lo que quiere decir que el caso de uso que extiende de otro caso de uso base no es obligatorio que ocurra, pero de hacerlo ofrece un valor extra para la ejecución del caso de uso base (Ortiz, 2012).
- **Generalización:** Especifica que un su hereda las características del caso de uso base y puede volver a especificar algunas o todas ellas de una forma muy similar a las herencias entre clases.

A continuación los casos de uso identificados, así como los requisitos funcionales que cada uno de estos agrupan. A continuación en la Tabla 5 se muestran los requisitos identificados, agrupados por casos de uso.

Tabla 5: Trazabilidad.

Caso de Uso	Referencia a Requisitos
Administrar gráfica de barras	RF1, RF5, RF9
Administrar gráfica de líneas	RF2, RF6, RF10
Administrar gráfica circular	RF3, RF7, RF11
Administrar gráfica de áreas	RF4, RF8, RF12

2.2.3 Diagrama de casos de uso del sistema

A continuación se describen gráficamente los casos de uso del componente gráfico mediante un Diagrama de Casos de Uso el cual se encarga de describir las relaciones y dependencias que existen entre los actores y los casos de uso del sistema. A continuación en la Figura 6 se muestra el diagrama de Casos de Uso del componente de graficado.

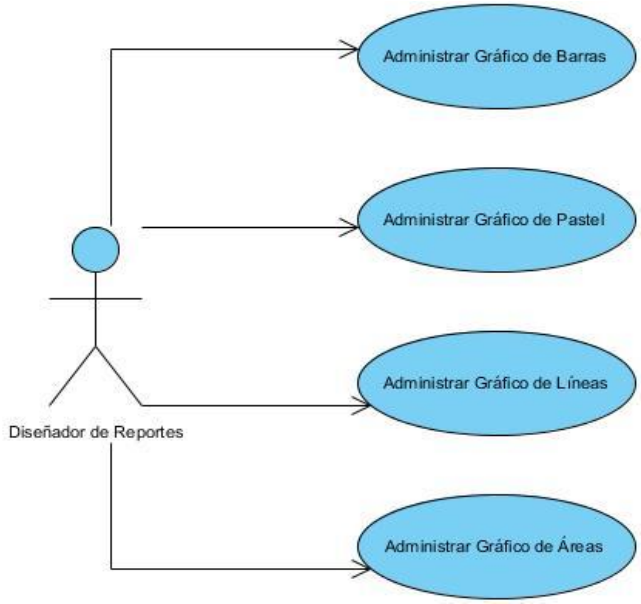

Figura 6: Diagrama de Casos de Uso del componente de graficado.

2.2.4 Descripción textual de casos de uso del sistema

Mediante la descripción textual de los casos de uso se exponen los procesos o eventos que tienen lugar en cada uno de ellos permitiendo un mejor entendimiento de cada una de las funcionalidades de cada caso de uso. A continuación la descripción textual de las Tabla 6 corresponde al caso de uso Administrar gráfico de barras.

Tabla 6: Descripción textual del caso de uso Administrar gráfico de barras.

Nombre	**Administrar gráfico de barras.**
Objetivo	Este caso de uso tiene como objetivo insertar, eliminar y modificar una gráfica de barras en un reporte.
Actores	Diseñador de Reportes.
Resumen	El caso uso se inicia cuando el actor selecciona el componente gráfico de barras y concluye cuando se representa el gráfico de barras en un reporte.
Complejidad	Alta

Precondiciones	Existe un reporte creado en el sistema GDR v2.0.	
Flujo básico Administrar gráfico de barras		
	Actor	**Sistema**
1	El usuario selecciona la opción: **Gráfico de barras** de la paleta de Gráficas y lo arrastra hacia el área de diseño del reporte donde se quiere insertar la gráfica.	El sistema adiciona el gráfico de barras en el área de diseño y muestra la respectiva ventana de configuración del gráfico.
2	El usuario configura las propiedades y opciones necesarias para que el componente funcione correctamente luego las guarda. En caso de que el usuario decida cancelar se cierra la ventana de configuración y se desaparece el componente.	Almacena el componente con las configuraciones previamente realizadas por el usuario.
3	El usuario después de haber insertado una gráfica tiene la posibilidad de modificar o eliminar la misma. 1. Ver Sección 1: “Modificar gráfico de barras”. Luego las guarda. 2. Ver Sección 2: “Eliminar gráfico de barras”. Luego las guarda.	Termina el caso de uso.
Sección 1: “Modificar gráfico de barras”		
Flujo básico Modificar gráfico de barras		
	Actor	**Sistema**

1	El usuario puede modificar las configuraciones previas que se han guardado de la gráfica dependiendo de las propiedades que desee cambiar: 1. Ver Sección 1.1: “Alinear gráfico de barras”. Luego guarda los cambios. 2. Ver Sección 1.2: “Modificar propiedades en el Editor del gráfico de barras”. Luego guarda los cambios. 3. Ver Sección 1.3: “Renombrar gráfico de barras”. Luego guarda los cambios. 4. Ver Sección 1.4: “Modificar propiedades del gráfico de barras”. Luego guarda los cambios.	El sistema almacena las nuevas configuraciones y se muestran en el gráfico de barras.
Sección 1.1: “Alinear gráfico de barras”		
Flujo básico Alinear gráfico de barras		
	Actor	**Sistema**
1	El usuario puede alinear la gráfica de barra de dos formas diferentes: 1. Haciendo un clic izquierdo sobre la gráfica para seleccionarla y luego clic derecho y aparece una lista con las propiedades que desea modificar. Selecciona la propiedad “Alineación”. Esta propiedad despliega las siguientes propiedades: a. Reiniciar posición: Al dar clic en esta opción reinicia a la gráfica a la posición inicial que tenía. b. Alinear: Al dar clic en esta opción ordena la gráfica en diferentes direcciones: izquierda, derecha, superior, inferior. c. Centrado: Al dar clic en esta opción Ajusta la gráfica vertical u horizontalmente.	El sistema almacena las nuevas configuraciones y se muestran en el gráfico de barras.

	d. Espaciado: Al dar clic en esta opción elimina los espacios de forma vertical u horizontalmente, además proporciona espaciado vertical promedio y espaciado horizontal promedio. e. Redimensionar: Al dar clic en esta opción transforma la dimensión a la gráfica al igualar la altura o igualar el ancho o ambas cosas. 2. Haciendo un clic izquierdo sobre la gráfica para seleccionarla y luego clic izquierdo sobre una de las propiedades que se encuentran en la paleta "Orientación" que se ubica en la parte superior del sistema, en el segundo menú del módulo "Diseñador de Reportes". En dicha paleta se encuentran las siguientes propiedades: a. Reiniciar posición: Al dar clic en esta opción reinicia a la gráfica a la posición inicial que tenía. b. Alinear: Al dar clic en esta opción ordena la gráfica en diferentes direcciones: izquierda, derecha, superior, inferior. c. Centrado: Al dar clic en esta opción Ajusta la gráfica vertical u horizontalmente. d. Espaciado: Al dar clic en esta opción elimina los espacios de forma vertical u horizontalmente, además proporciona espaciado vertical promedio y espaciado horizontal promedio. e. Redimensionar: Al dar clic en esta opción transforma la dimensión a la gráfica al igualar la altura o igualar el ancho o ambas cosas.	
Sección 1.2: "Modificar propiedades en el Editor del gráfico de barras"		
Flujo básico "Modificar propiedades en el Editor del gráfico de barras"		
	Actor	**Sistema**

1	El usuario puede modificar las propiedades del editor del gráfico de barras al insertar la gráfica o haciendo un clic izquierdo sobre la propiedad “categoryDataset” que se encuentra en la paleta de “Propiedades” en la parte derecha inferior del sistema en el Módulo ¨Diseñador de Reportes¨.	El sistema muestra una ventana con el título: “Editor del gráfico de barras”
2	El usuario configura las propiedades del editor del gráfico de barras: • Expresión de serie: El número de la gráfica. • Expresión de categoría: Los datos del eje X, o sea las expresiones. • Expresión de valor: Los valores de las Y, o sea los valores numéricos. • Expresión de etiqueta: El número de sesiones de la gráfica.	El sistema almacena las nuevas configuraciones y se muestran en el gráfico de barras.
Sección 1.3: “Renombrar gráfico de barras”		
Flujo básico “Renombrar gráfico de barras”		
	Actor	**Sistema**
1	El usuario puede renombrar la gráfica haciendo clic izquierdo para seleccionar la gráfica y luego clic izquierdo en la propiedad “Renombrar” apareciendo un cuadro de diálogo con un campo de texto para el nuevo nombre, un botón Aceptar y uno Cancelar.	El sistema almacena las nuevas configuraciones y se muestran en el gráfico de barras.
Sección 1.4: “Modificar propiedades del gráfico de barras”		
Flujo básico “Modificar propiedades del gráfico de barras”		
	Actor	**Sistema**

1	El usuario puede modificar las propiedades de la gráfica que se encuentran en la paleta “Propiedades” en la parte derecha inferior del sistema en el Módulo ¨Diseñador de Reportes¨. Las propiedades que se pueden configurar son las siguientes: • Backcolor: Cambia el color de fondo de la gráfica. • categoryDataset: Al dar clic en esta propiedad aparece una ventana para añadir más secciones a la gráfica. • Forecolor: Para darle color a cada sección. • Heigth: Para darle valor a la altura de la gráfica. • Width: Para darle valor al ancho de la gráfica. • X: Eje de coordenada horizontal. • Y: Eje de coordenada vertical. • Name: nombre de la gráfica.	El sistema almacena las nuevas configuraciones y se muestran en el gráfico de barras.

Sección 2: “Eliminar gráfico de barras”		
Flujo básico Eliminar gráfico de barras		
	Actor	**Sistema**
1	El usuario tiene dos opciones para eliminar: 1. Selecciona la gráfica con clic izquierdo y luego Clic derecho y selecciona la opción “Eliminar”. 2. Selecciona la gráfica con clic izquierdo y presionar la tecla “Delete”. 3. Seleccionar “Eliminar” en la paleta “Archivo”, que se encuentra en la parte superior del sistema en el segundo menú del módulo “Diseñador de Reportes”.	El sistema muestra una ventana de confirmación de eliminación de la gráfica. Si el usuario seleccionó “Si” entonces se elimina el gráfico de barras del reporte. Si el usuario seleccionó “No” entonces el reporte no sufre ningún cambio, o sea que no se elimina el gráfico de barras del reporte.

Relaciones	***Caso de Uso Incluidos***	Ninguno.
	Caso de Uso Extendidos	Ninguno.

2.3 Descripción de la solución propuesta

Después de conformar el modelo del dominio, especificar los requisitos funcionales y no funcionales, además de elaborar los casos de usos que describen las relaciones y dependencias que existen entre los actores y los casos de uso del componente, se propone como solución a la problemática planteada desarrollar un componente que permita la representación de los datos en los reportes en forma de: gráficas de barras, gráficas de líneas, gráficas de pastel y/o gráficas de áreas. El componente de graficado se ubica en la parte derecha-superior dentro del menú Paleta de componentes. Cuando el usuario selecciona el componente de graficado se despliegan los tipos de gráficas: gráficas de barras, gráficas de líneas, gráficas de pastel y gráficas de áreas. Al seleccionar y arrastrar hacia el reporte un tipo de gráfica, en el panel de propiedades se despliegan las propiedades y configuraciones correspondientes al elemento creado. Una vez configuradas las opciones necesarias para que el componente grafique los datos deseados, se guardan los cambios realizados y se puede visualizar la gráfica en el reporte haciendo uso de la vista previa del reporte para corroborar que el mismo esta como el usuario lo desea.

2.4 Arquitectura

El diseño arquitectónico se ha descrito como un proceso de varios pasos en el cual las representaciones de la estructura de los datos y el programa, las características de la información y el detalle procedimental se sintetizan a partir de los requisitos (Pressman, 2005).

Symfony, como framework de desarrollo que se emplea en el desarrollo del componente, contiene una arquitectura modular basada en el Modelo Vista Controlador (MVC) como patrón de diseño web. Para el diseño de la solución propuesta se decide utilizar como patrón el MVC para continuar con la arquitectura que se utiliza en el sistema GDR v2.0.

A continuación se muestra una breve descripción de este patrón:

El MVC es un patrón de diseño de software que separa la lógica de funcionamiento de una aplicación en tres componentes distintos (el modelo, la vista y el controlador). El patrón MVC se ve frecuentemente en aplicaciones Web, donde la vista es la página HTML, el controlador es la página PHP y el modelo es la fuente de datos.

- **Modelo:** Es responsable del procesamiento, validación, asociación, entre otros, así como de la recuperación de datos desde la fuente de datos, convirtiéndolos luego en conceptos significativos para el usuario. Es la representación específica de la información con la cual opera el reporte previamente generado.
- **Vista:** Utilizada por los usuarios para interactuar con el componente de graficado. La vista hace una presentación de los datos del modelo y es responsable del uso de la información de la cual dispone para producir la interfaz de presentación según la petición que haya realizado el usuario. En este caso, según el tipo de gráfica que sea solicitada.
- **Controlador:** Recibe las entradas, traducidas a solicitudes de servicio para el modelo. Es un bloque de código que realiza llamadas al modelo para obtener los datos y se los pasa a la vista para que los muestre al usuario. Este responde a eventos, usualmente acciones del usuario, e invoca cambios tanto en el modelo y como en la vista.

El patrón MVC se adapta perfectamente a la programación web sobre la que está basada la solución propuesta. Esto permitió separar al componente de graficado en tres capas (modelo, vista y controlador). Todas las peticiones realizadas por el usuario son a través de la vista, la cual tiene embebido al componente de graficado, esta a su vez se las envía a la clase controladora. La clase controladora traduce las entradas y se las envía al modelo que es quien contiene la fuente de datos, en este caso es el Reporte. El modelo envía los datos a la controladora y este a la vista mostrándole así una respuesta al usuario. A continuación en la Figura 7 se muestra el diseño de la arquitectura MVC para el componente de graficado:

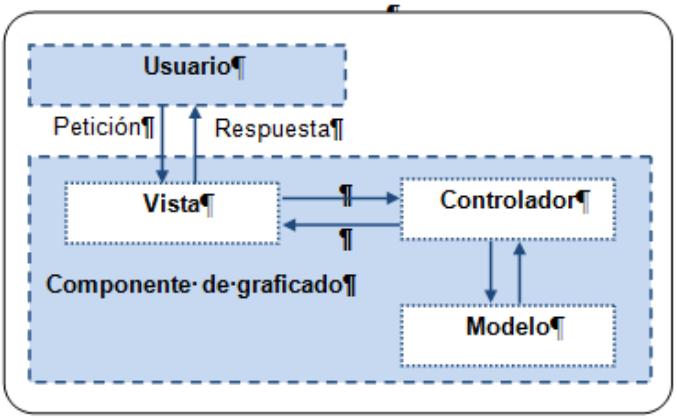

Figura 7: Patrón de diseño Modelo Vista Controlador (MVC) para el componente de graficado.

2.4.1 Patrones de diseño

Para la realización de la implementación del componente se tuvieron en cuenta una serie de patrones de diseño, con el objetivo de hacer el componente de graficado reusable y entendible para otros diseñadores formalizando un vocabulario común entre diseñadores y estandarizando el modo en que se realiza el diseño. Para este fin se utilizaron varios patrones entre los que se encuentran:

2.4.1.1 Patrones Grasp

- **Experto**: el patrón experto es el principio básico de asignación de responsabilidades. Dado que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. Lo cual nos da una serie de beneficios como que se mantiene el encapsulamiento, los objetos utilizan su propia información para llevar a cabo sus tareas. Se distribuye el comportamiento entre las clases que contienen la información requerida. Son más fáciles de entender y mantener. Este patrón se evidencia en la relación que existe entre la BarChart.js y la Chart.js donde la primera especializa el comportamiento de la segunda.
- **Creador:** el patrón creador es el que ayuda a identificar quién debe ser el responsable de la creación de nuevos objetos o clases. La cual solo puede ser creada por la clase que tiene la información necesaria para realizar la creación del objeto lo cual supone facilidad de mantenimiento y reutilización. Este patrón se evidencia en la clase Designer.js donde se maneja la creación de cada uno de los componentes que componen el diseño de los reportes.

- **Controlador:** un Controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar en evento del sistema. Define además el método de su operación. A este patrón se le asigna la responsabilidad de las operaciones del sistema. Un ejemplo de la utilización de este patrón es la clase ReportController.php, la cual es la encargada de manejar las peticiones realizadas por los usuarios sobre los reportes.

La alta cohesión y el bajo acoplamiento están relacionados directamente con el patrón experto y el creador. Aunque los conceptos de Alta Cohesión y bajo acoplamiento no están relacionados entre sí, dado que son dos patrones distintos sin embargo se recomienda tener un mayor grado de cohesión con un menor grado de acoplamiento y se logra cuando existe la utilización del patrón experto y creador.

2.4.1.2 Patrones Gof

Patrones GOF (Gang of Four o “Banda de los Cuatro”). Los patrones de diseño GOF dan una descripción de clases y objetos que se comunican entre sí, adaptada para resolver un problema general de diseño en un contexto particular. Se clasifican en:

- **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.

Para el desarrollo del componente de graficado se utilizaron los siguientes patrones GOF:

- **Factory Method:** define una interfaz para crear un objeto, pero deja que sean las subclases quienes decidan qué clase instanciar. Permite que una clase delegue en sus subclases la creación de objetos. Este patrón se evidencia en la clase Chart.js la cual delega, a las clases que se especializan en la construcción del componente grafico en cuestión, la creación de los mismos.
- **Visitor:** El propósito principal de es abstraer la funcionalidad que puede ser aplicada a una jerarquía de objetos (elementos). En la implementación del componente se hace evidente en las clases, donde visita cada una de las etiquetas y atributos de la primera convirtiéndolos a objetos json y de objetos json sus respectivas etiquetas y atributos de la segunda obteniendo el documento XML.

2.5 Modelo de Diseño

Los modelos de diseño muestran los objetos o clases en un sistema, y donde sea apropiado los diferentes tipos de relaciones entre estas entidades .Son el puente entre los requerimientos y la implementación del sistema (Somerville, 2005).

2.5.1 Diagrama de clases del Diseño

El diagrama de clases del diseño es la representación de una estructura que muestra gráficamente las características de un sistema, mediante clases, atributos y las relaciones entre ellos. A continuación en la Figura 8 se describe el diagrama de clases del diseño de la estructura del componente gráfico para el Generador Dinámico de Reportes v2.0:

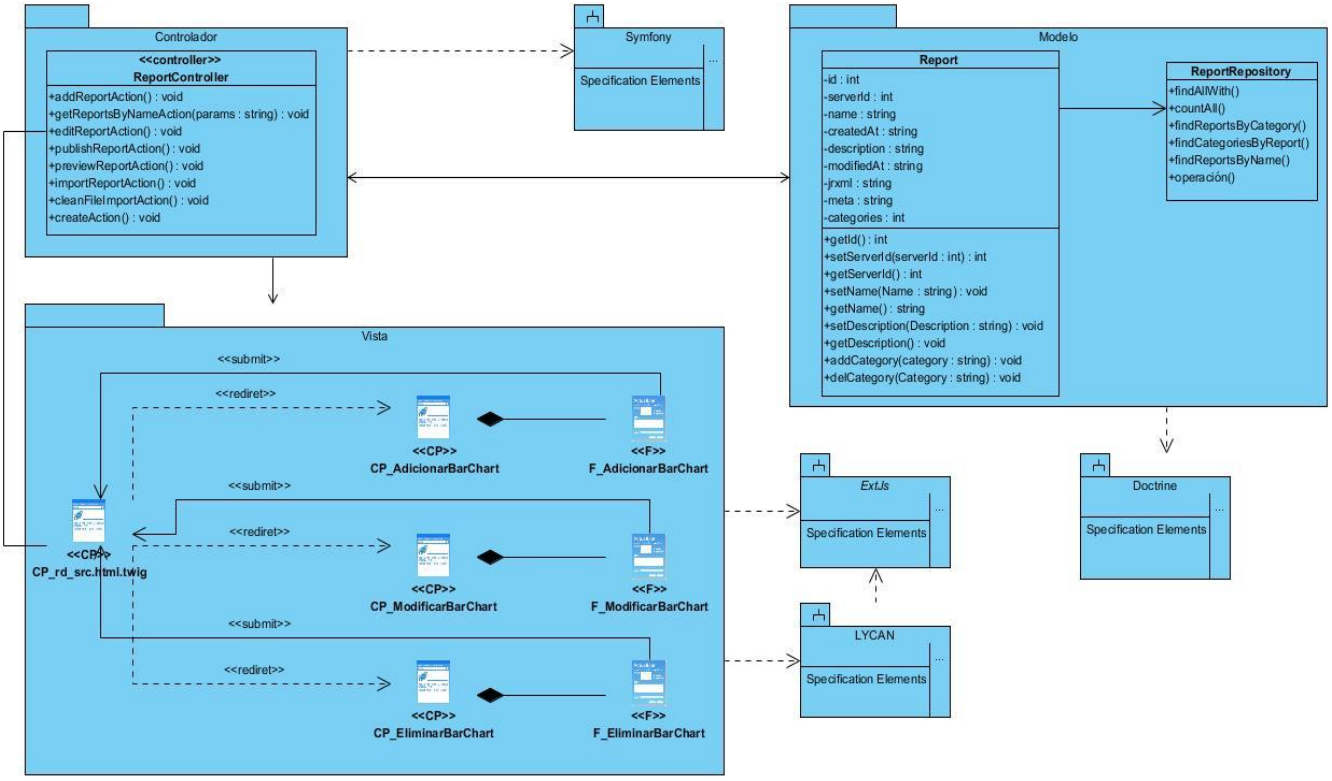

Figura 8: Diagrama de clases del diseño del caso de uso "Administrar gráfico de barras".

El diagrama de clases del diseño está formado por tres paquetes: la Vista, el Controlador y el Modelo. Dentro de la Vista se encuentran las clases encargadas de interactuar con el usuario, la principal en este paquete es la clase rd.src.html.twig, esta es la plantilla donde se incluyen todas las clases js y css

necesarias para realizar las funciones: Adicionar gráfico de barras, Modificar gráfico de barras y Eliminar gráfico de barras del componente de graficado. Por otra parte se encuentra el Controlador con la clase controladora *ReportControlles.php* en la cual se encuentran los métodos cuyas funcionalidades se relacionan con la gestión de los datos de los reportes permitiendo crear, almacenar, modificar y buscar los reportes en la base de datos. Además está el paquete Modelo, en este están las clases *ReportRepository.php* y *Report.php,* en la primera se implementan las consultas para acceder a la bases de datos y obtener los reportes almacenados y la segunda representa la entidad del reporte en la bases de datos.

Conclusiones del Capítulo

El modelado del dominio realizado posibilitó identificar las relaciones que existen entre las entidades GDR, Diseñador de Reportes, Componente y Reporte comprendidas en el ámbito del problema, así sus respectivos atributos.

Se identificó un total de 15 requisitos entre funcionales y no funcionales, agrupándose los funcionales en 4 Casos de Uso del sistema que fueron modelados y descritos.

Se emplea el patrón Modelo Vista Controlador como patrón arquitectónico para separar la lógica de funcionamiento del componente en tres partes bien delimitadas entre sí y los patrones Experto, Creador y Controlador como patrones de diseño GRASP.

Se representó, a través del diagrama de clases del diseño, la estructura que muestra gráficamente las características del componente de graficado mediante clases, atributos y sus relaciones.

.

Capítulo III: Implementación y Prueba

En el capítulo que a continuación se presenta se muestran los artefactos de los eventos de implementación y prueba que se realizaron para desarrollar el componente Gráfico del sistema Generador Dinámico de Reportes v2.0. Se realizan además pruebas al sistema para comprobar que no existan errores en la implementación, para en caso de que estén puedan ser corregidos.

3.1 Estándares de codificación

Para comenzar con la implementación del componente de graficado se establecen primeramente los estándares de codificación. La utilización de los mismos permite que todos los programadores del proyecto trabajen de forma coordinada. Además de que ofrecen la posibilidad de realizar modificaciones y actualizaciones por parte de otros programadores sin la necesidad de emplear mucho tiempo con respecto a la comprensión del código fuente. Por otra parte son los definidos por el equipo de desarrollo del GDR v2.0, por lo que se decide continuar con la misma línea de trabajo.

A continuación se muestran los estándares de codificación que se utilizaron para la implementación del componente gráfico y continuar con la línea base del proyecto:

- **LowerCamelCase**: Es la práctica de escribir frases o palabras compuestas eliminando los espacios y poniendo en mayúscula la primera letra de cada palabra, excepto la de la primera que es en minúscula. A continuación en la Figura 9 se muestra un ejemplo de un método del componente que utiliza este estándar:

```
stripChilds: function(node) {
    return (function() {
      var hash = {},
          i,
          len = node.hasChildNodes() ?
node.childNodes.length : 0,
          e;
```

Figura 9: Método stripChilds.

- **Identación:** La identación o sangrado debe estar hecha mediante tabulaciones y no se deben insertar espacios. En el caso de la aplicación es desarrollada en el IDE NetBIOS y este ya cumple con esta característica.
- **Variables**: Las variables deben usar LowerCamelCase y además tener nombres coherentes y que la identifiquen, o sea que no se le deben poner nombres como “a” o “b”, solo en caso de que la variable sea demasiado genérica como para no tener un nombre concreto, como por ejemplo el índice de un bucle (un ciclo *for* o *while*) se declarará con nombre “i”, “j”, “k” y así sucesivamente en el caso de varios bucles anidados.
- **Clases:** Los nombres de las clases deben estar en *UpperCamelCase.* Todo el contenido de la clase como atributos o métodos debe estar identados con una tabulación con respecto a la misma. La implementación de los métodos debe estar identadas con una tabulación con respecto al nombre del mismo.
- **Funciones:** Los nombres de las funciones o métodos deben usar LowerCamelCase y además deben existir los métodos como modificador de acceso para cada atributo de la clase, aunque para los atributos públicos no es necesario. A continuación en la Figura 10 se muestra un ejemplo de un método del componente que utiliza este estándar:

```
getChildsByName: function(name) {
            return hash[name];
        },
        hasChilds: function(name) {
            return hash[name] !== undefined;
        }
    };
```

Figura 10: Método getChildsByName.

- **Operadores:** Los operadores como =, +, +=, -, -=, etcétera deberán estar separados por un espacio antes y después del operador. Esta regla no se aplica para los operadores “++” y “--“, estos se utilizan sin espacios a la variable que modifican.
- **Estructuras de control:** Las estructuras de control como *if, while, do while, for, foreach, switch* deberán escribirse de acuerdo al estilo implementado por el IDE NetBIOS.

- **Comentarios:** Los comentarios de aclaración del código deberán hacerse en la línea inmediatamente superior a la línea de código fuente a la cual se desea aclarar.

3.2 Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos y sus realizaciones en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros. Muestra la organización y las dependencias entre un conjunto de componentes (Pressman, 2005).

Un diagrama de componentes permite representar los componentes en los que se divide un sistema de software, así como las relaciones de dependencia entre los mismos, donde los componentes físicos incluyen archivos, bibliotecas compartidas, módulos, ejecutables, o paquetes. A continuación se describe el diagrama de la solución propuesta. A continuación en la Figura 11 se muestra el diagrama de componentes del caso de uso Administrar gráfico de áreas.

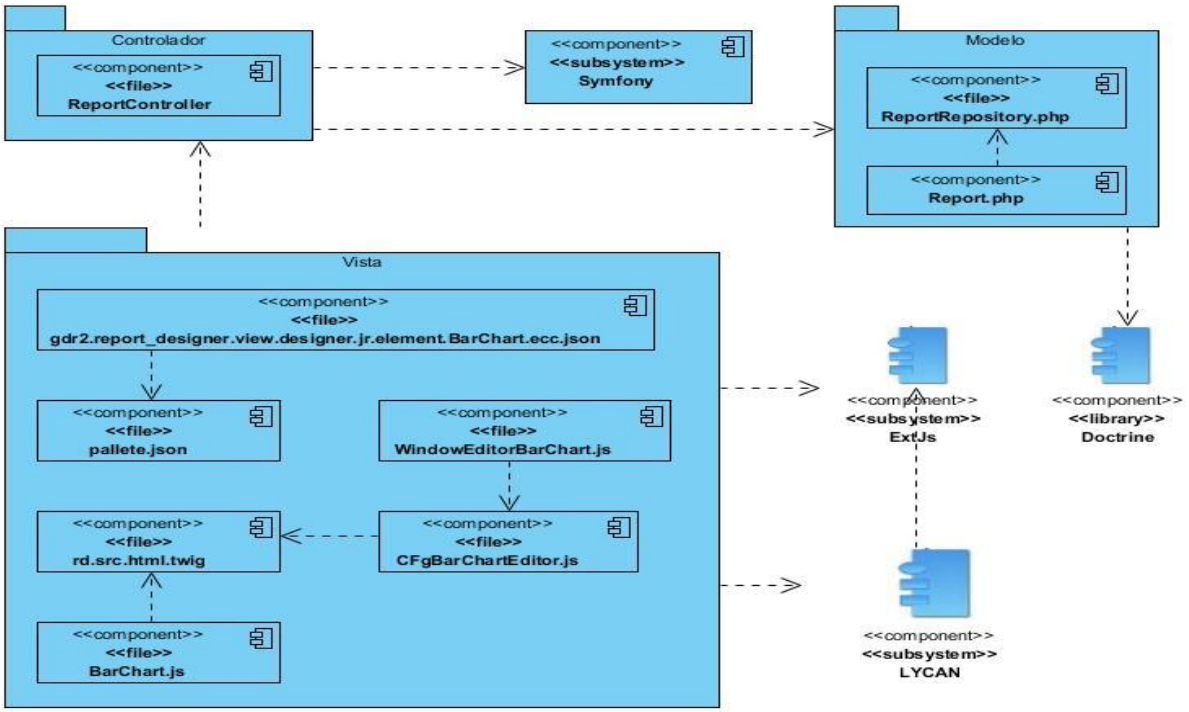

Figura 12: Diagrama de componentes del caso de uso "Administrar gráfico de áreas".

En este caso el diagrama muestra los tres paquetes fundamentales: el paquete de la vista, donde están ubicados los componentes de la interfaz principal con la que interactúa el usuario. El paquete llamado Controlador en el cual se encuentra la clase controladora encargada del funcionamiento de la aplicación y el paquete de datos que contiene las clases repositorios encargadas de almacenar las operaciones con las entidades.

3.3 Pruebas

Luego de finalizado el proceso de implementación del componente gráfico para el GDR v2.0 se hace necesario verificar el funcionamiento y la calidad del mismo y comprobar de esta manera si está listo a ser empleado en el graficado de datos en los reportes, por lo que se procede a la fase de prueba.

Las pruebas del software son el proceso que permite verificar, garantizar y mostrar la calidad de un producto, a través de resultados registrables que proporcionan una evaluación y representa una revisión final de las especificaciones, del diseño y de la codificación. Son empleadas para identificar posibles fallos durante el proceso de desarrollo. Los casos de prueba son actividades en las cuales un sistema o componente es ejecutado bajo condiciones o requerimientos especificados, permitiendo encontrar y documentar los defectos que puedan afectar la calidad del software (Rodríguez, 2012).

3.3.1 Estrategia de Prueba

Para dar inicio a las pruebas de un software lo primero es describir una estrategia de prueba donde quede plasmado los niveles de prueba a tratar, así como los tipos de prueba a emplear en cada nivel, los métodos de prueba a aplicar y las técnicas a utilizar con método.

Una estrategia de prueba describe y verifica el enfoque de la misma. Tiene como función fundamental demostrar que diferentes técnicas facilitan la prueba planificada, así como definir las mismas (manual o automática), verificando que el enfoque trabajará, producirá resultados precisos y es apropiado para los recursos disponibles. Además incluye los niveles de prueba a ser diseccionados, el tipo de prueba a ser ejecutada y los casos de prueba diseñados para lograr los objetivos. Delimita los criterios de éxitos y culminación de las pruebas y define consideraciones especial es relacionadas con los recursos necesarios para realizar esta tarea (Rodríguez, 2012).

3.3.1.1 Niveles de Pruebas

Los niveles de prueba especifican diferentes ángulos para verificar y validar un producto de software. Existen varios niveles de pruebas como: pruebas de desarrollador, independiente, unidad, integración, sistema y aceptación, donde cada nivel contiene una técnica de prueba específica según los atributos de calidad que se deseen verificar. En el desarrollo del componente se aplicarán solamente las pruebas siguientes para comprobar que el sistema da respuesta a los requisitos funcionales definidos anteriormente:

- **Nivel de Sistema:** en este nivel las pruebas tienen como propósito ejercitar profundamente el sistema para verificar que se han integrado adecuadamente todos los elementos del sistema (hardware, software) y que realizan las funciones adecuadas.
- **Nivel de Integración:** en este nivel se prueba los componentes combinados para ejecutar un caso de uso. Además se realiza la prueba para descubrir errores en las especificaciones de las interfaces de las clases.
- **Nivel de Aceptación:** en este nivel las pruebas son destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida por el cliente. Estas pruebas aseguran el comportamiento del sistema.

3.3.1.2 Tipos de Pruebas

Existen diferentes tipos de pruebas que se pueden aplicar para verificar que el componente de conexión cumple con todos los requisitos identificados en el proceso de análisis y comprobar su correcto funcionamiento. A continuación se explican los tipos de pruebas seleccionados:

- **Pruebas Funcionales:** son aquellas que tienen por objetivo demostrar que los sistemas desarrollados, cumplen con las funciones específicas para los cuales han sido creados. Es común que sea desarrollada por los analistas de pruebas con apoyo de algunos usuarios finales y están basadas en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas para el software. (Sánchez, y otros, 2013).
- **Pruebas de Integración:** son un conjunto de pruebas unitarias, funcionales, de regresión y aceptación que se realizan para probar el software. Incluye también comprobar que lo programado por los diferentes desarrolladores funciona en un entorno real. (Sánchez, y otros, 2013).

- **Pruebas de Aceptación:** una prueba de aceptación tiene como propósito demostrar al cliente el cumplimiento de los requisitos del software. Estas son básicamente pruebas funcionales sobre el sistema completo. Además tienen como objetivo obtener la aceptación final del cliente antes de la entrega del producto para su paso a producción. (Huaraca, y otros, 2013).

3.3.1.3 Método de Prueba

Dentro de los métodos de prueba son dos fundamentales: el método de la caja negra para las pruebas funcionales y de la caja blanca para las pruebas estructurales.

Se decide utilizar el método de caja negra, el cual se centra en los requerimientos funcionales del software, o sea, permite demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce un resultado correcto.

Este método intenta encontrar errores de las siguientes categorías:

- Errores de interfaz.
- Errores en estructura de datos o en acceso a base de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

La ventaja fundamental del método de caja negra es que se centra en los requisitos funcionales del software permitiendo al ingeniero del software obtener conjuntos de condiciones de entrada que ejerciten completamente todos los requisitos funcionales de un programa. Se llevan a cabo sobre la interfaz del sistema reduciendo el número de casos de prueba mediante la elección de entradas y salidas válidas y no válidas que ejercitan todas las funcionalidades del sistema.

3.4 Diseños de Casos de Pruebas

El caso de prueba específica la forma de probar un sistema incluyendo las entradas, salidas y resultados esperados, así como bajo qué condiciones debe probarse el sistema. Su tarea principal es verificar el cumplimiento de un objetivo en particular.

Para realizar las pruebas funcionales se utilizaron juegos de datos válidos e inválidos haciendo uso de la técnica de partición de equivalencia. A continuación se muestra el caso de prueba asociado al Caso de Uso Administrar gráfico de barras.

3.4.1 Descripción de las variables

Tabla 7: Descripción de las variables del diseño de casos de prueba del Caso de Uso Administrar gráfico de barras.

No	Nombre del campo	Clasificación	Valor nulo	Descripción
1	Expresión de la Serie	Campo de texto	No	Se introduce el número de la gráfica o id.
2	Expresión de la Categoría	Campo de texto	No	Los datos del eje de las X o sea las expresiones.
3	Expresión del Valor	Campo de texto	No	Los datos del eje de las Y o sea los valores numéricos.
4	Expresión de la Etiqueta	Campo de texto	No	La cantidad de sesiones que tiene la gráfica.

Tabla 8: Caso de prueba Insertar gráfico de barras del caso de uso Administrar Gráfico de barra.

Escenario	Variables					Descripción	Respuesta del sistema	Flujo Central
	1	2	3	4	5			
EC 1.1: Insertar gráfica de barras con datos correctos.	V	V	V	V	V	Se adicionan correctamente la gráfica.	El sistema adiciona la gráfica correctamente.	1-Se despliega el componente gráfico.2-Se selecciona la gráfica de barras y se arrastra hasta el área del reporte donde se quiere insertar.3-Se insertan los datos en el formulario ¨Editor del Gráfico de Barras¨ que aparece. 4-Se escoge la opción aceptar.
EC 1.2: Insertar gráfica con datos incorrectos	I	V	V	V	V	Se adiciona la gráfica con datos incorrectos.	El sistema adiciona la gráfica con datos incorrectos.	1-Se despliega el componente gráfico.2-Se selecciona la gráfica de barras y se arrastra hasta el área del reporte donde se quiere insertar.3-Se insertan los datos en el formulario “Editor del Gráfico de Barras” que aparece. 4-Se escoge la opción aceptar.
	V	I	V	V	V			
	V	V	I	V	V			
	V	V	V	I	V			
	V	V	V	V	I			

EC 1.3: Cancelar la operación.						Cancela la operación.	El sistema cancela la adición de la gráfica al reporte.	1-Se despliega el componente gráfico.2-Se selecciona la gráfica de barras y se arrastra hasta el área del reporte donde se quiere insertar.3-Se insertan los datos en el formulario ¨Editor del Gráfico de Barras¨ que aparece. 4-Se escoge la opción cancelar.5-Se cierra el formulario de datos.6-No se inserta la gráfica.

Tabla 9: Caso de prueba Eliminar gráfico de barras del caso de uso "Administrar Gráfico de barra".

Escenario	Variables					Descripción	Respuesta del sistema	Flujo Central
	1	2	3	4	5			
EC 2.1: Eliminar gráfico de barras con el clic derecho del mouse.	V	V	V	V	V	Se elimina la gráfica de barras del reporte.	El sistema elimina la gráfica de barras del reporte.	1-Se selecciona la gráfica de barras y se oprime el clic derecho.2-Se despliega un menú con varias opciones incluyendo la opción "Eliminar".3-Se selecciona la opción "Eliminar".4-Se muestra un mensaje con las opciones "Si" y "No".5-Se selecciona la opción "Si".
EC 2.2: Eliminar gráfico de barras con la tecla delete.	V	V	V	V	V	Se elimina la gráfica de barras del reporte.	El sistema elimina la gráfica de barras del reporte.	1-Se selecciona la gráfica de barras.2-Se presiona le tecla delete.3-Se muestra un mensaje con las opciones "Si" y "No".5-Se selecciona la opción "Si"

EC 2.3 Cancelar operación.						Cancela la operación.	El sistema cancela la eliminación de la gráfica de barras del reporte.	1-Se selecciona la opción “No” .2-El sistema no realiza ningún cambio.

Tabla 10: Caso de prueba Modificar gráfico de barras del caso de uso “Administrar Gráfico de barra”.

Escenario	Variables					Descripción	Respuesta del sistema	Flujo Central
	1	2	3	4	5			
EC 3.1: Modificar los datos de la gráfica de barras.	V	V	V	V	V	Se modifican los datos de la gráfica de barras.	El sistema modifica los datos de la gráfica de barras.	1-Se selecciona la gráfica de barras.2-Se despliega la paleta de propiedades correspondientes a dicha gráfica .3-Se selecciona la propiedad “CategoryDataset” .4-Se muestra el formulario “Editor del Gráfico de Barras” con los datos de la gráfica.5-Se modifican los datos.6-Se selecciona la opción aceptar.
EC 3.2: Modificar la posición de la gráfica de barras por la paleta de propiedades.	V	V	V	V	V	Se modifica la posición de barras en el reporte.	El sistema modifica la posición de la gráfica de barras en el reporte.	1-Se selecciona la gráfica de barras.2-Se despliega la paleta de propiedades correspondientes a dicha gráfica .3-Se selecciona las propiedades: “X”, “Y” .4-Se modifican los valores de las mismas.

EC 3.3: Modificar la posición de la gráfica de barras por la paleta "Orientación" que se encuentra en el segundo menú en la parte superior del módulo de "Diseñador de Reportes".	V	V	V	V	V	Se modifica la posición de barras en el reporte.	El sistema modifica la posición de la gráfica de barras en el reporte.	1-Se selecciona la gráfica de barras.2-Se selecciona la propiedad "posicionar" en la paleta "Orientación" que se encuentra en el segundo menú en la parte superior del módulo de "Diseñador de Reportes".3-Se selecciona la opción deseada.
EC 3.4: Modificar la posición de la gráfica de barras usando el menú del clic derecho.	V	V	V	V	V	Se modifica la posición de barras en el reporte.	El sistema modifica la posición de la gráfica de barras en el reporte.	1-Se selecciona la gráfica de barras y otro elemento del contenedor.2-Se oprime el clic derecho.3-Se despliega un menú con varias opciones y la opción alineación.3-Se despliega un menú a partir de la opción alineación con cuatro submenú.4-Se escoge la opción deseada.
EC 3.5: Modificar las dimensiones de la gráfica de barras por la paleta de propiedades.	V	V	V	V	V	Se modifican las dimensiones de la gráfica de barras.	El sistema modifica las dimensiones de la gráfica de barras.	1-Se selecciona la gráfica de barras.2-Se despliega la paleta de propiedades correspondientes a dicha gráfica .3-Se selecciona las propiedades: "width", "height".4- Se modifican los valores de las mismas.

EC 3.7: Modificar las dimensiones de la gráfica de barras usando el menú del clic derecho.	V	V	V	V	V	Se modifican las dimensiones de la gráfica de barras.	El sistema modifica las dimensiones de la gráfica de barras.	1-Se selecciona la gráfica de barras y otro elemento del contenedor.2-Se oprime el clic derecho.3-Se despliega un menú con varias opciones y la opción alineación.3-Se despliega un menú a partir de la opción alineación con cuatro submenú.4-Se escoge submenú redimensionar.5-Se escoge la opción deseada.
EC 3.8 Cancelar operación.	V	I	V	V	V	Cancela la operación.	El sistema no realiza ningún cambio.	1-El sistema no realiza ningún cambio.

Este proceso permitió verificar el cumplimiento de los requisitos funcionales del sistema, donde los resultados de las pruebas que no fueron satisfactorios pasaron a ser no conformidades. En el proceso de pruebas se detectaron 4 no conformidades, las cuales fueron corregidas en cada iteración. En la siguiente gráfica se muestran cuantas no conformidades fueron encontradas en cada iteración.

Figura 13: Gráfica de no conformidades de las pruebas funcionales.

3.4.2 Pruebas de Integración

Según Roger Pressman “Las pruebas de integración son una técnica sistemática para la construcción de la arquitectura de software, mientras al mismo tiempo, se aplican las pruebas para descubrir los errores asociados con la interfaz”. Existen dos tipos de integración, no incremental e incremental. No incremental es cuando se combinan todos los módulos y se prueba todo el programa en su conjunto y la integración incremental es cuando el programa se construye y se prueba en pequeños segmentos en los que los errores son más fáciles de aislar y corregir. Para realizar la prueba de integración al componente gráfico se decide emplear la integración incremental (Pressman, 2005).

La integración incremental es un enfoque donde el producto se construye y se prueba en segmentos pequeños, donde los errores son más fáciles de aislar y corregir; las interfaces se pueden probar de forma completa, y se puede aplicar un enfoque de pruebas sistemáticas (Pressman, 2005).

Una vez terminado el componente de graficado se integra al módulo Diseñador de Reportes hasta que funciona como un todo.

3.4.2.1 Pasos para realizar la integración

1. Se incluyen las clases de los componentes en la plantilla twig del módulo diseñador.

2. Se adicionan los nombres de las clases de especificación de los componentes gráficos en la clase Designer.js para que esta cargue sus propiedades y opciones de configuración.

3. Se incluyen las clases asociadas a las ventanas de edición de las opciones de configuración de los componentes gráficos en la plantilla twig del módulo diseñador.

4. En la clase ComponentNodeToXmlDocVisitor.js se le implementan las funciones necesarias para que transforme los objetos JSON asociados a los componentes gráficos creados en el jrxml correspondiente y a la clase XmlDocToObjectVisitor.js se le incluye las funciones para realizar el proceso inverso.

5. Se ejecuta el comando de Symfony 2 assets: install web - - symlink para que instale las clases de la vista incluidas que no estaban en GDR2 hasta el momento.

Para llevar a cabo las pruebas de integración se comenzó por integrar el componente de graficado al módulo Diseñador de Reportes para luego ir incorporando cada uno de los gráficos (gráfico de barras, gráfico de pastel, gráfico de línea, gráfico de área) a dicho componente. Para ello el equipo de desarrollo realizó varias pruebas cada vez que se integraba un nuevo gráfico; obteniendo un total de 50 no

conformidades, descomponiéndose en: 30 no conformidades en el gráfico de barras, 10 en el gráfico de pastel, 5 en gráfico de línea y 5 en el gráfico de área. Las no conformidades encontradas fueron tratadas hasta que el componente funcionó como un todo con el módulo antes mencionado.

Figura 14: Gráfica de no conformidades de las "Pruebas de Integración".

3.4.3 Prueba de Aceptación

Para realizar la prueba de aceptación se empleó la técnica de prueba alfa donde los interesados fueron representados por especialistas del departamento Desarrollo de Componentes del Centro de Tecnologías de Gestión de Datos de la UCI. Se evaluaron las funcionalidades del componente basándose en la especificación de requisitos del expediente de proyecto y del manual del usuario. Los interesados verificaron cada requisito funcional y comprobaron si estos correspondían con los requerimientos planteados.

Después de concluida las pruebas, el departamento Desarrollo de Componentes del Centro de Tecnologías de Gestión de Datos liberó el componente de graficado, entregándole al equipo de desarrollado la Carta de aceptación en la que consta que el componente está apto para ser utilizado y cumple con las expectativas planteadas (ver Anexo 2).

Conclusiones del Capítulo

Primeramente se determinan los estándares de codificación. Esto permite que la implementación de todo el código del componente sea entendible para que cualquier programador que interactúe con el mismo pueda realizar modificaciones o actualizaciones.

Una vez finalizado el proceso de implementación se traza un plan de estrategias de pruebas (pruebas funcionales y pruebas de integración) para verificar el funcionamiento y la calidad del mismo. Dichas pruebas detectaron cuatro no conformidades, las cuales fueron resueltas inmediatamente en cada iteración. Obteniendo así resultados positivos para el componente de graficado, donde los usuarios estén satisfechos con la solución.

Conclusiones Generales

El desarrollo del presente trabajo permitió arribar a las siguientes conclusiones:

- El análisis del marco teórico tuvo como base la representación gráfica de los datos, esto determinó la concepción que da respuesta a la problemática planteada o sea el componente de graficado.
- A través del análisis de la metodología Open UP se realizaron los diagramas que modelan el negocio, para guiar todo el proceso de desarrollo del componente de graficado.
- La selección de las herramientas y tecnologías fueron de vital importancia para lograr la implementación del componente.
- El análisis de algunos sistemas generadores de reportes permitió constatar de que todos incluyen un componente de graficado, por lo que se demuestra que este es un componente esencial para enriquecer el contenido de los reportes.
- La captura de los requisitos funcionales y no funcionales permitieron describir las funcionalidades que debe cumplir el componente. A partir de esto se modela el dominio del negocio que facilitó la captura de los objetos, eventos y las relaciones importantes entre ellos para conocer el entorno donde estará el componente de graficado, así como la descripción del flujo de actividades que permiten detallar como sería el funcionamiento del mismo.
- Se llevó a cabo el proceso de implementación para darle funcionamiento al componente.
- Se realizaron pruebas donde se detectaron algunas no conformidades que fueron eliminadas, propiciando que el componente de graficado fuera capaz de representar datos mediante gráficos en los reportes que genera el GDR v2.0, cumpliendo así con el objetivo previsto para que los usuarios visualizaran los datos de los reportes de una forma más entendible y sugerente.

Recomendaciones

Una vez concluida la investigación, la implementación y las pruebas correspondientes al componente de graficado del GDR v2.0 se recomienda:

- Incluir nuevos tipos de representaciones gráficas.

Bibliografía

1. **Sparx Systems. 2013.** Sparxsystems. *Sparxsystems.* [En línea] 2013. http://www.sparxsystems.com.ar/.
2. **B, Alexander Oré. 2010.** Calidad del Software.com. *Calidad del Software.com.* [En línea] 2010. http://www.calidadysoftware.com/testing/pruebas_unitarias1.php.
3. **Balduino, Ricardo. 2007.** eclipse. *eclipse.* [En línea] 2007. http://www.eclipse.org/epf/general/OpenUP.pdf.
4. **Bárcena, Mariano Cabrero. 2015.** Administración de una red. *Administración de una red.* [En línea] 2015. http://administraciondeunared.bligoo.com.
5. **Bouly, Yanet Ennis. 2013.** *Componentes para la generación de reportes dinámicos en el GDR sobre bases de datos en Access y Oracle 11g.* La Habana : s.n., 2013.
6. **CrystalReports. 2003.** CrystalReportsBook.com. *CrystalReportsBook.com.* [En línea] 2003. [Citado el: 12 de noviembre de 2015.] http://www.crystalreportsbook.com/Crystal_Reports_Net_Ch01_1.asp.
7. **Díaz, Ivis Pages. 2012.** *Diseño y aplicación de pruebas a la Plataforma Soberana GeneSIG.* La Habana : s.n., 2012.
8. **Eguiluz, Javier. 2015.** LIBROSWEB. *LIBROSWEB.* [En línea] 3 de enero de 2015. http://librosweb.es/libro/javascript/.
9. **Fabien Potencier, François Zaninotto. 2015.** LBROSWEB. *LBROSWEB.* [En línea] 2015. http://www.librosweb.es/symfony_1_2.
10. **Foote, R. Johnson y B. 1998.** *Journal of Object-Oriented Programming.* 1998.
11. **Frederick, Shea, Ramsay, Colin y Blades, Steve 'Cutter'. 2008.** *Learning Ext JS.* 2008.
12. **Fronckowiak, John W. 2008.** *Build Ajax applications with Ext JS.* 2008.

13. **GARCIA, JESUS. 2011.** *Ext JS in Action.* 2011.

14. **Gimson, Loraine. 2012.** *Metodologías ágiles y desarrollo basado en conocimiento.* 2012.

15. **Guerra, Yuned Rivero, y otros. 2014.** *GENERADOR DINÁMICO DE REPORTES.* VILLA CLARA : s.n., 2014.

16. **Guerrero, Rafael Martínez. 2013.** PostgresSQL-es. *PostgresSQL-es.* [En línea] 2013. http://www.postgresql.org.es/sobre_postgresql.

17. **Gutierrez, Demián. 2010.** *MVC (Model-View-Controller).* 2010.

18. **Gutiérrez, René Arenas, Sol, Juana María Romero del y Hernández, Katia García. 2014.** Revista Cubana de Informática Médica. *Revista Cubana de Informática Médica.* [En línea] 2014. http://www.rcim.sld.cu/revista_4/articulos_html/rene.htm.

19. **Hernández, Iliana Amabely Silva. 2003.** Generador Automático de Reportes Dinámicos. [En línea] 2003. [Citado el: 14 de enero de 2015.] http://www.cs.cinvestav.mx/TesisGraduados/2003/tesisSilvaHernandez.pdf.

20. **Hernández, Yulainne Alonso. 2015.** *CONFIGURACIÓN DE LA METODOLOGÍA OPEN UP V1.0.* 2015.

21. **Huaraca, Abner Gerardo Valdez, Valdez, Jorge Luis Mendoza y Pachas, Carlos Orlando. 2013.** academia.edu. *academia.edu.* [En línea] 2013. [Citado el: 11 de junio de 2015.] http://es.slideshare.net/abnergerardo/pruebas-de-sistemas-y-aceptacion-23663195.

22. **Larman, Craig. 2003.** *UML y Patrones.* Madrid : s.n., 2003.

23. **Larousse. 1999.** *Diccionario El pequeño Larousse.* 1999.

24. **Laterza, Older Osvaldo Krause. 2011.** Sistema Avanzado de Distribución. *Sistema Avanzado de Distribución.* [En línea] 11 de julio de 2011. [Citado el: 9 de junio de 2015.] http://mistock.lcompras.biz/index.php?option=com_content&view=article&id=1249:ireport&catid=59:tallersoftware&Itemid=116.

25. **López, Patricia. 2012.** *Herramienta CASE Visual Paradigm.* 2012.

26. **Lozano, Pablo Ramiro Correa. 2010.** *Análisis comparativo de los frameworks adobe flex, Java Rich Faces y Ext Js para el desarrollo de aplicaciones enriquecidas en internet (RIA).* Quito : s.n., 2010.

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

28. **Martínez, Ramón Alexander Anglada y Hernández, Alain Abel Garófalo. 2013.** Revista Cubana de Ciencias Informáticas. *Revista Cubana de Ciencias Informáticas.* [En línea] junio de 2013. http://scielo.sld.cu/scielo.php?pid=S2227-18992013000200006&script=sci_arttext. ISSN 2227-1899.

29. **Méndez, Alejandra Virrueta. 2010.** *METODOLOGÍAS DE DESARROLLO DE SOFTWARE.* Michoacán : s.n., 2010.

30. **Microsoft. 2015.** Microsoft. *Microsoft.* [En línea] 2015.

31. **Moisés. 2011.** Excel Total. *Excel Total.* [En línea] 2011. https://exceltotal.com/partes-de-un-grafico-de-excel.

32. **Montalvo, Marlene Melián. 2010.** *XML el nuevo lenguaje universal.* 2010.

33. **Oracle . 2015.** NetBeans. *NetBeans.* [En línea] 2015. [Citado el: 10 de junio de 2015.] https://netbeans.org/features/index.html.

34. **Ortiz, Leticia. 2012.** AVISTAR. *AVISTAR.* [En línea] 2012. http://www.milestone.com.mx/articulos/casos_a_incluir_casos_a_extender.htm..

35. **paradigm, visual. 2014.** visual paradigm. *visual paradigm.* [En línea] 2014. http://www.visual-paradigm.com/support/documents/vpositoryuserguide/1337/1341/81325_managingmemb.html.

36. **Visual Paradigm . 2015.** *Visual Paradigm.* [En línea] 2015. http://www.visual-paradigm.com/.

37. **php. 2015.** php. *php.* [En línea] 2015. [Citado el: 10 de junio de 2015.] http://php.net/manual/es/migration53.new-features.php.

38. Planet NetBeans **. 2015.** *Planet NetBeans.* [En línea] 2015. http://www.planetnetbeans.org/.

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

40. **Prendes, Lourdes M. Dueñas. 2004.** *Caracterización de un Sistema de Gestión de Información.* La Habana : s.n., 2004.

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

42. **Ralston, Anthony, Reilly, Edwin D. y DavidHemmendinger. 2000.** *Encyclopedia of ComputerScience, 4th Edition.* 2000.

43. **Rodríguez, Julio César Brito. 2011.** *MODEL DESIGNER FOR DYNAMIC REPORT GENERATOR 2.0.* 2011.

44. **Rodríguez, Julio César Brito . 2012.** *Módulo Diseñador de Modelos para el Generador Dinámico de Reportes v2.0.* La Habana : s.n., 2012.

45. **Rodríguez, Julio César Brito, y otros. 2013.** *Dynamics Report Generator v2.0: local development.* La Habana : s.n., 2013.

46. **Rodríguez, Yadiel Ramos y Hernández, Dayana Daniel. 2010.** *Generación de reportes con Jasper Report.* La Habana : s.n., 2010.

47. **Sampieri, Roberto Hernández. 1998.** *Metodología de la Investigación.* 1998.

48. **Sánchez, Sureny Orihuela y López, Liandry Monteslier. 2013.** *Componente de conexión para la generación de reportes mediante un servicio web en el Generador Dinámico de Reportes.* La Habana : s.n., 2013.

49. **Silva Hernández, Iliana Amabely.** 2003. Generador Automático de Reportes Dinámicos. [En línea] 2003. [Citado el: 14 de enero de 2015.] http://www.cs.cinvestav.mx/TesisGraduados/2003/tesisSilvaHernandez.pdf.v

50. **Sommerville, Ian. 2005.** *Ingeniería de Software. Séptima edición.* Madrid : s.n., 2005. ISBN/84-7829-074-5.

51. **The Eclipse Foundation. 2014.** BIRT. *BIRT.* [En línea] 2014. http://www.vogella.com/tutorials/EclipseBIRT/article.html.

52. **Villate, Jaime E. 2001.** *Introducción al XML.* Universidad de Oporto : s.n., 2001.

53. **Zayas, Carlos Alvarez de. 1995.** *METODOLOGIA DE LA INVESTIGACION CIENTIFICA .* 1995.

Anexos

Anexo 1

Guía de entrevista

Objetivo: Obtener los requisitos funcionales y no funcionales del componente de graficado.

Entrevistado: Ing. Yoander Iñiguez Bermúdez.

Ocupación: Líder del proyecto de desarrollo del GDR v2.0.

Preguntas para la entrevista:

1. ¿cuáles son las principales deficiencias que existen en el GDR v2.0 a la hora de elaborar reportes?
2. ¿cuáles son las principales funcionalidades que deben ser implementadas en el componente de graficado?
3. ¿cuáles son los requerimientos de software que necesita el GDR v2.0 tener instalados para llevar a cabo el proceso de graficado de datos con éxito?
4. ¿cuáles son los requerimientos de hardware que necesita el GDR v2.0 tener instalados para llevar a cabo el proceso de graficado de datos con éxito?
5. ¿Cuáles son los tipos de gráficos más prudenciales para incluir en los reportes?

Anexo 2

UCI Universidad de las Ciencias Informáticas

CARTA DE ACEPTACIÓN

En cumplimiento con la fase de desarrollo y en función de la ejecución del proyecto. Generador Dinámico de Reportes (GDR v2.0)para el Departamento de Desarrollo de Componentes del Centro de Tecnología de Gestión de Datos de la facultad 6. DATEC, se hace entrega de los productos que se relacionan a continuación:

- Componente para la representación gráfica de datos en los reportes diseñados en el Generador Dinámico de Reportes 2. (código fuente).

Entrega	Recibe	Recibe
Nombre y apellidos: Tahimi Nuñez Rondón Laura Mercedes Camacho González	**Nombre y apellidos**: Glennis Tamayo Morales	**Nombre y apellidos**: Yoander Iñiguez Bermúdez
Cargo: Tesistas	**Cargo:** Jefe de Departamento Desarrollo de Componentes	**Cargo:** Jefe de proyecto GDRv2.0
Firma:	**Firma:**	**Firma:**

UCI Universidad de las Ciencias Informáticas
Decanato Facultad 6

Fecha: 15/06/2015