# Universidad de las Ciencias Informáticas

# Facultad 6


**Título:** Componente de graficado v1.1 para el módulo Diseñador de reportes del Generador Dinámico de Reportes v2.0.

## Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

**Autores:** Yamil Alonso de la Concepción

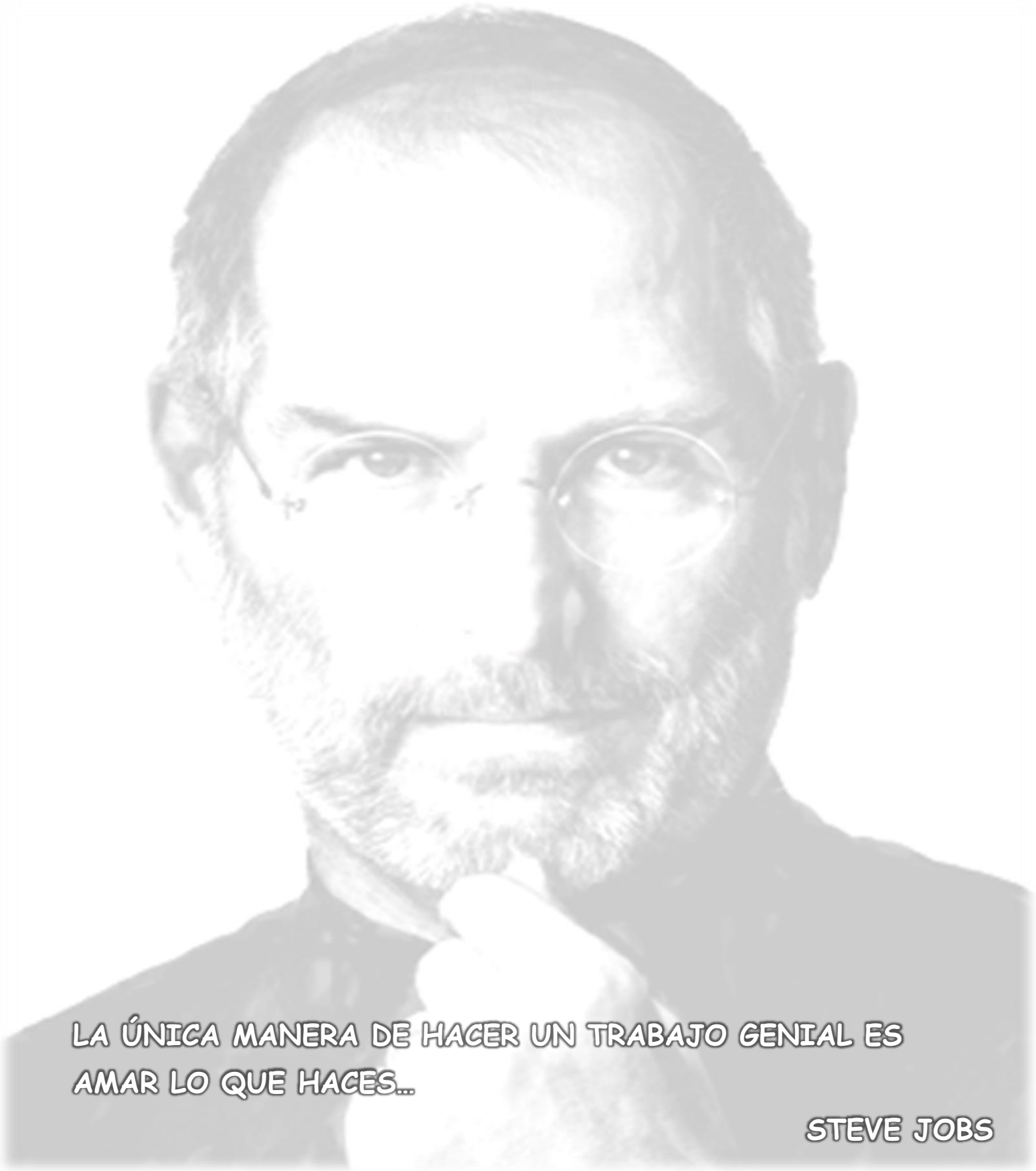
Víctor Manuel Bermejo García

**Tutores:** MsC. Yadira Robles Aranda

Ing. Juan Miguel Pérez Almaguer

La Habana, Julio 2016

"Año 58 de la Revolución"


LA ÚNICA MANERA DE HACER UN TRABAJO GENIAL ES AMAR LO QUE HACES…
STEVE JOBS

## Declaración de Autoría

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

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

**Yamil Alonso de la Concepción**

_____
Firma del Autor

**Victor Manuel Bermejo García**

_____
Firma del Autor

**MsC. Yadira Robles Aranda**

_____
Firma del tutor

**Ing. Juan Miguel Pérez Almaguer**

_____
Firma del tutor

## Datos de Contacto

**Tutor:** MsC. Yadira Robles Aranda
Universidad de las Ciencias Informáticas
La Habana, Cuba
E-mail: yrobles@uci.cu

**Tutor:** Ing. Juan Miguel Pérez Almaguer
Universidad de las Ciencias Informáticas
La Habana, Cuba
E-mail: jmalmaguer@uci.cu

## Dedicatoria

Este trabajo está dedicado a mis padres María Josefa y Victor Bermejo por ser las personas más extraordinarias del mundo. Nadie más que ustedes merecen este sueño hecho realidad. Porque han sacrificado gran parte de su vida para formarme y educarme, porque nunca podré pagar sus desvelos ni aún con las riquezas más grandes del mundo, por su ejemplo, amor y confianza; a ustedes que fueron testigos del camino andado para llegar hasta aquí. Espero que estén orgullosos de mí, pues todo lo que hago es pensando en hacerlos feliz.

***Victor Manuel Bermejo García***.

Dedico mi trabajo y estos 17 años de estudio a mis padres por ser las personas tan especiales que son para mí “los mejores padres del mundo”, por estar presente en cada momento de mi vida para guiarme de la mejor manera, y apoyarme en los momentos difíciles.

***Yamil Alonso de la Concepción***

## Agradecimientos

***Victor Manuel Bermejo García***

Sé que mil palabras no bastarían para agradecerles su apoyo, su comprensión y sus consejos en los momentos difíciles, pero ahora mismo esta es la manera que tengo de darle las gracias.

A mi mamá que por razones de la vida no puede estar físicamente a mi lado en este momento, lo cual me duele infinitamente, pero estoy seguro que su corazón y toda su atención están aquí conmigo. Por creer en mí cuando nadie lo hacía. Porque todas mis tristezas eran tuyas, todos mis tropiezos los convertías en experiencia y junto a mi disfrutaste mis momentos de triunfo y por darme esa fuerza infinita que tiene cuando estoy a punto de rendirme.

A mi papá por ser quien me enseña a tener el carácter y la disciplina necesaria para ser un hombre de bien. Por siempre haber estado presente en los momentos más complicados de mi vida. Porque la nobleza y la sencillez son tus características principales. Estoy orgulloso de ser tu hijo y no hay forma humana ni palabras coherentes que me permitan expresar el agradecimiento que siento por ti. Porque eres un ejemplo a seguir en todos los sentidos y espero algún día llegar a ser como tú.

A mis hermanos y abuelos, por estar siempre a mi lado y brindarme su apoyo incondicional.

No puedo dejar de agradecer a 4 personas que se han convertido en mi familia, mis amigos Daynelis, Sael y mis dos hermanos y mejores amigos Yamil y Blanco. Ustedes se ganaron mi respeto y mi admiración, hoy quiero retribuirles los momentos que me apoyaron cuando más lo necesité. En parte este logro también es suyo, gracias porque personas como ustedes son únicas.

En especial agradezco a Yamil por ser también mi dúo de tesis que contribuyó y se esforzó para que todas las cosas siempre salieran de la mejor manera posible. Gracias porque sin ti no hubiera logrado cumplir esta meta.

A mis tutores Yadira y Juan Miguel por habernos guiado en todo momento de este trabajo. Gracias por el esfuerzo realizado y por la contribución que hicieron para convertirnos en ingenieros. También reconocer a la profe y amiga Glennis por todo el esfuerzo realizado por sacarnos adelante en todo momento, por todos sus desvelos y regaños para lograr terminar este trabajo con éxito.

A mis amigos Cuso, Yosbel, Elian, a todos mis compañeros de aula desde primer año y a todos aquellos que de una forma u otra me han ayudado en este largo camino.

***Yamil Alonso de la Concepción***

Agradezco a todas aquellas personas que siempre desearon que este sueño se hiciera realidad especialmente a mi madre, mi padre, mi tía y a mis abuelos, los mejores del mundo, por estar siempre pendientes de mí, por apoyar mis decisiones y guiar mis pasos por la vida, porque sin ellos nada de esto hubiera sido posible.

A mis 3 hermanos y mis primas que son la luz de mi vida.

A mi familia, mis compañeros, mis vecinos y todos los que simplemente alguna vez han preguntado, “cuando te vas”, “” cuando regresas”, o sencillamente, “está todo bien, necesitas ayuda”.

A mis amigos, porque compartir este sueño con ustedes fue empezar a hacerlo realidad, gracias por compartir conmigo estos 5 años en los que he vivido experiencias inolvidables.

A mi dúo de tesis que además de compartir este trabajo, vivió conmigo esta carrera tan maravillosa compartiendo momentos de alegría, de preocupación, de tristeza, en fin, fuiste como un hermano en todo este tiempo por todo eso y por ser de mis mejores amigos, gracias.

A los tutores por su atención y dedicación, este resultado es tanto mío como vuestro.

A los especialistas del departamento de Componentes del centro DATEC por su apoyo al desarrollo de este trabajo.

Al tribunal por sus críticas constructivas y por el tiempo que incondicionalmente me dedicaron.

A los profesores, que con tanto sacrificio me han dado un poco más que conocimiento y me han ayudado a formarme como profesional.

A la UCI, por ser a lo largo de cinco años más que una escuela, haber sido nuestra casa, nuestro barrio, nuestro pueblo. Por dotarnos de tantos conocimientos y experiencias inolvidables.

A Glennis por su apoyo incondicional, por haber estado presente en cada instante de este proceso, por haberme acompañado la mitad de mi camino en la universidad, por haberme enseñado muchas cosas no sólo de la carrera, si no de la vida, gracias por estar siempre para mí, gracias por ser mi pareja, gracias por ser tú.

## Resumen

El Centro de Tecnologías de Gestión de Datos (DATEC) perteneciente a la Universidad de las Ciencias Informáticas (UCI), cuenta entre las disímiles soluciones que desarrolla, con el Generador Dinámico de Reporte en su versión 2.0 (GDR v2.0), solución que tiene como objetivo la creación de reportes dinámicos que pueden ser utilizados para la toma de decisiones. En el curso 2014-2015 se implementó la versión 1.0 del componente de representación gráfica para los reportes que se crean con GDR v2.0. Este componente actualmente sólo cuenta con 4 tipos de gráficas: barras, pastel, línea y área, con las cuales sólo se puede mostrar visualmente un conjunto de información muy generalizada, lo que limita el uso de estas en los reportes. El presente trabajo tiene como objetivo desarrollar el componente de graficado v1.1 para el módulo Diseñador de Reportes del Generador Dinámico de Reportes v2.0. Para ello se realizó una investigación sobre los tipos de gráficos que soporta JasperReport v5.1, motor que utiliza GDR v2.0 para la generación de los reportes. Además, se identificaron las funcionalidades y se realizó el diseño e implementación de la solución. Finalmente se realizaron las pruebas de software para comprobar el correcto funcionamiento del mismo. La nueva versión del componente brinda a los usuarios 20 tipos de gráficos que posibilitan analizar y visualizar la información en los reportes de forma clara y atractiva.

PALABRAS CLAVES: componente de graficado, reportes dinámicos.

## Abstract

The Data Management Technology Center (DATEC by its Spanish acronyms) from the University of Informatics Sciences (UCI by its Spanish acronyms) has among the different solutions it develops a Dynamic Report Generator (GDR by its Spanish acronyms) in its version 2.0 (GDR v2.0), solution that aims at creating dynamic reports that can be used for decision making. During the curse 2014-2015 it was implemented version 1.0 of a graphic representation component for the reports created with GDR v2.0. Today, this component has just four types of graphics: bar, pie, line and area, with which can only be visually displayed a set of very generalized information, which makes impossible to compare multiple sets of information in the reports, thus limiting their use in presentations. The present work aims at developing the graphic component v1.1 for the module Reports Designer of the Dynamic Report Generator v2.0. In order to achieve that, an investigation was made on the kind of graphics supported by JasperReport v5.1, engine that uses GDR v2.0 to generate reports. The functionalities were also identified and the design and implementation of the solution were made as well. Finally, software tests were made to check its proper functioning. The new version of the component provides a wide range of graphics that allow analyzing and visualizing the information in the reports in a clear and intuitive way.

KEY WORDS: graphed component, dynamic reports.

# Índice de contenido

# Índice de figuras

# Índice de tablas

# Introducción

La información que se gestiona y se almacena dentro de una empresa puede ser utilizada como un recurso para el uso, manejo y alcance de su éxito, por lo que se considera de vital importancia para lograr un alto nivel de competitividad y desarrollo. Los continuos cambios a los cuales deben enfrentarse estas entidades obligan a sus directivos a tomar decisiones de manera cada vez más acertada y oportuna.

Un factor clave para el apoyo a la toma de decisiones es el análisis de información, donde los reportes juegan un papel fundamental. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar. El reporte, de esta forma, confiere una mayor utilidad a los datos. Existen numerosas formas de representación de la información dentro de estos, entre los cuales se encuentran los textos, las tablas, las imágenes y los gráficos. La utilización de estos últimos brinda un apoyo visual, que permite, de manera sencilla y sugerente, interpretar la información para llegar a conclusiones de manera más rápida y efectiva.

Los numerosos avances tecnológicos, así como la inclusión de las Tecnologías de la Información y las Comunicaciones (TIC) en los procesos de negocios de empresas e instituciones, han beneficiado la generación de reportes sobre la información que estas almacenan, surgiendo herramientas informáticas capaces de informatizar la generación de los mismos.

En instituciones como la Universidad de las Ciencias Informáticas (UCI) específicamente en el Centro de Tecnologías de Gestión de Datos (DATEC), que tiene como objetivo crear productos informáticos, desarrollar tecnologías y proveer servicios relacionados con la gestión de datos y el análisis de información, se lleva a cabo el desarrollo del sistema Generador Dinámico de Reportes (GDR) en su versión 2.0. Esta es una herramienta web que permite a sus clientes consultar los gestores de bases de datos de sus organizaciones y generar reportes con la información que estos manejan. Para el diseño de los reportes GDR 2.0 utiliza diferentes componentes entre los que se encuentra el de graficado, quien actualmente cuenta con solo 4 tipos de gráficas (Pastel, Barras, Línea y Área), lo que trae consigo que:

- Sólo se puede mostrar visualmente un conjunto de información muy generalizada, lo que dificulta comparar desde los reportes la información contenida en series de datos distintas, limitando a su vez el uso de dichos reportes en presentaciones.
- Se reduce la perspectiva de visualizar la información para realizar diferentes tipos de análisis, por ejemplo, evaluación de patrones y comportamiento de datos en el transcurso del tiempo y/o la

realización del seguimiento y control del progreso de cada una de las etapas de un proyecto, entre otros.

Teniendo en cuenta la situación descrita surge como **problema de la presente investigación:** ¿Cómo brindar al usuario final de GDR v2.0 una mejor representación visual de la información que apoye a la toma de decisiones?, definiéndose como **objeto de estudio:** La representación gráfica de datos, enmarcado en el **campo de acción:** Representación gráfica de los datos en los reportes creados con el Generador Dinámico de Reportes v2.0. Para darle solución al problema planteado se define como **objetivo general**: Desarrollar el componente de graficado v1.1 para el módulo Diseñador de reportes del Generador Dinámico de Reportes v2.0, que permita una mejor representación visual de la información que apoye a la toma de decisiones.

Para guiar la lógica de la investigación se enumeran las siguientes **preguntas científicas**:

- ¿En qué se fundamenta teóricamente el proceso de representación de la información contenida en un reporte mediante gráficos?
- ¿Qué tecnologías, metodología y herramientas utilizar para el desarrollo del componente de graficado v1.1 del GDR v2.0?
- ¿Qué características y capacidades se deben tener en cuenta para lograr el correcto funcionamiento del componente de graficado v1.1 para GDR v2.0?
- ¿Cómo estructurar el proceso de implementación del componente de graficado v1.1 para GDR v2.0 que permita una mejor representación de los componentes a desarrollar?
- ¿Qué pruebas aplicar para comprobar el correcto funcionamiento del componente de graficado v1.1 para GDR v2.0?

Para lograr el cumplimiento del objetivo general se realizarán las siguientes **tareas de investigación:**

- Caracterización de las técnicas y herramientas existentes para la graficación de datos.
- Selección de la metodología, tecnologías y herramientas para el desarrollo del componente de graficado v1.1 del Generador Dinámico de Reportes v2.0.
- Identificación de los requisitos funcionales y no funcionales para la implementación del componente de graficado v1.1 de GDR v2.0.
- Diseño de la solución a partir de los requisitos identificados y la arquitectura de software definida para el desarrollo del componente de graficado v1.1 de GDR v2.0.
- Implementación de los requisitos identificados para obtener el componente de graficado v1.1 de GDR v2.0.

- Diseño de los casos de pruebas para validar el correcto funcionamiento de la solución desarrollada.
- Aplicación de los casos de prueba para validar que la solución cumple con los requisitos identificados.

La investigación se sustenta en los siguientes **métodos de investigación:**

**Métodos empíricos:**

- Entrevista: Se utiliza para obtener información que contribuye con el desarrollo de la investigación y aporta conocimientos específicos relacionados con la representación gráfica de datos. Se realiza a los especialistas del proyecto GDR v2.0.(Ver Anexo 3)
- Análisis estático: Se emplea para hacer un análisis minucioso de la estructura del componente de graficado anteriormente desarrollado y del IReport v5.1.

**Métodos teóricos:**

- Analítico-sintético: Se utiliza en el estudio de las soluciones existentes, en el análisis de las herramientas a utilizar, en la selección de las pruebas de software, así como en cada uno de los análisis que se realizaron a lo largo de la investigación.
- Modelación: Se utiliza para representar, mediante modelos, la arquitectura y los distintos procesos que se realizan en la construcción del componente de graficado de GDR 2.0 utilizando el Lenguaje de Modelado Unificado (UML v2.0).

El siguiente trabajo está estructurado en 3 capítulos, los cuales se describen a continuación:

- **Capítulo I. Fundamentación teórica de la investigación:** En este capítulo se identifican y describen los conceptos asociados a la representación gráfica de información. Se realiza un análisis sobre los principales sistemas diseñados para la representación de datos mediante gráficas existentes en el mundo. Además, se realiza la selección y descripción de las tecnologías, herramientas y metodología de desarrollo de software a utilizar para el desarrollo de la solución propuesta.
- **Capítulo II. Análisis y diseño de la solución propuesta:** En este capítulo se realiza la descripción del componente a desarrollar, para lograr una mayor claridad y comprensión por parte de los desarrolladores. Los puntos que se abordarán son los siguientes: descripción del modelo de dominio, especificación de los requisitos funcionales y no funcionales, representación de los actores y casos de uso a través de un diagrama de casos de uso del sistema, la descripción textual de cada uno de ellos, así como los patrones de diseño GoF y GRASP que serán utilizados y los diagramas de clases del diseño.

- **Capítulo III. Implementación y prueba de la solución:** En el presente capítulo se describe la implementación de los componentes a partir de los requisitos identificados. Se presenta el diagrama de componentes correspondiente a cada uno de los casos de uso a implementar. Además, se muestran ejemplos de las pruebas de software aplicadas, que permite examinar la aplicación desarrollada y garantizar la calidad del software.

# Capítulo I: Fundamentación teórica de la investigación

En este capítulo se identifican y describen los conceptos asociados a la representación gráfica de información. Se realiza un análisis sobre los principales sistemas diseñados para la representación de datos mediante gráficas existentes en el mundo. Además, se realiza la selección y descripción de las tecnologías, herramientas y metodología de desarrollo de software a utilizar para el desarrollo de la solución.

## 1.1 Conceptos asociados al dominio del problema

La descripción y el entendimiento de los principales elementos relacionados con el problema de investigación son esenciales para lograr un resultado práctico a partir del conocimiento adquirido. Por lo cual a continuación se definen los principales elementos relacionados con la investigación.

***Reporte***

Varias son las definiciones de reportes dadas en la bibliografía consultada como por ejemplo en (Vazquez Ocampo, 2013), en (Definición, 2015), en (Quees, 2015), pero para la investigación se decidió utilizar la de (Definición.de, 2008-2015) por ser la más relacionada con el tema de la investigación que plantea:

En el ámbito de la informática, los reportes son informes que organizan y exhiben la información contenida en una base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. El reporte, de esta forma, confiere una mayor utilidad a los datos.

***Sistemas generadores de reportes***

Los generadores de reportes son herramientas complementarias de los sistemas de información. Utilizan una especie de lenguaje transparente para el usuario por medio del cual éste realiza consultas a las bases de datos y obtiene información de ellas en forma de reporte. (Hernández, 2003)

Los generadores de reportes son entonces herramientas que se encargan del diseño y generación de reportes a partir de datos que estén almacenados en las bases de datos.

***Gráfico***

Un gráfico es todo tipo de representación visual que incluye figuras y/o signos para comunicar uno o una serie de conceptos. Puede consistir en una imagen a la que se le han adosado una o más leyendas explicativas, o un complejo sistema de esquemas y símbolos que representan la vinculación de varios conceptos abstractos entre sí. Por ejemplo, un gráfico puede dar cuenta de un procedimiento científico, el progreso en un proyecto, o el informe sobre los resultados de un trabajo organizacional. (DefiniciónABC, 2013)

***Elementos que componen un gráfico***

Un gráfico está formado por diferentes partes las cuales facilitan la forma de representar e interpretar la información. A continuación, se describen cada una de las partes que componen un gráfico según (Vera, 2015).

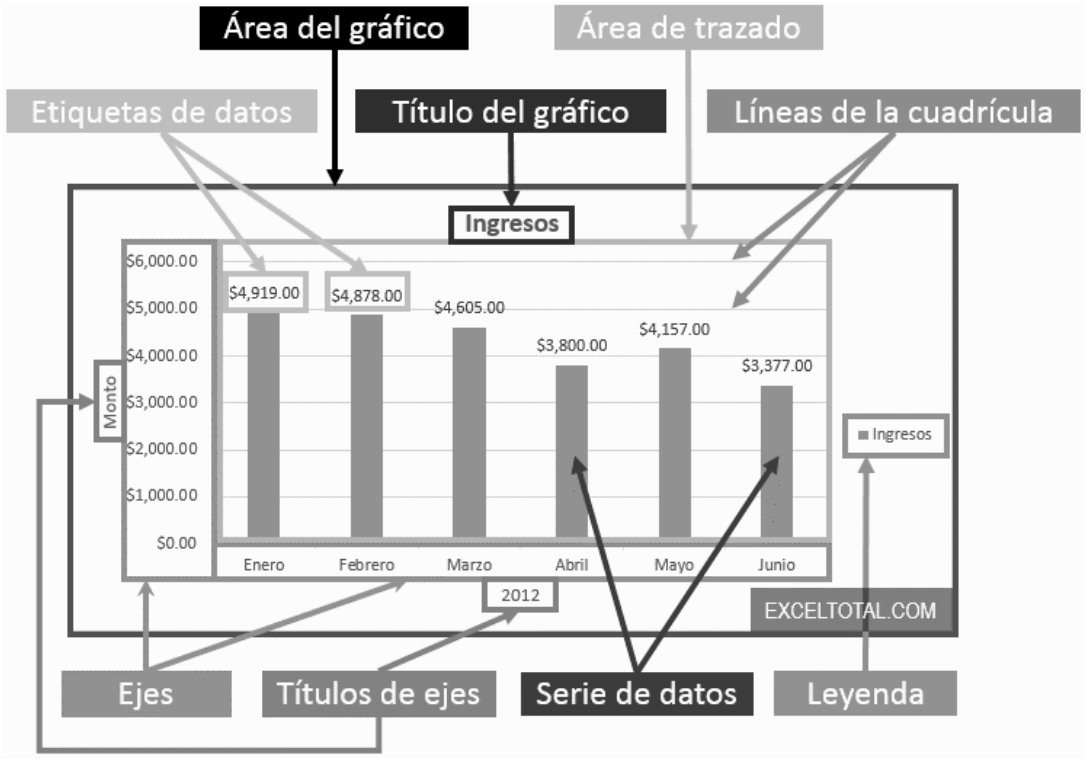



Fig. 1 Elementos de un gráfico.

- **Área del gráfico**: se encuentra definida por el marco del gráfico y que incluye todas sus partes.
- **Título del gráfico**: es un texto descriptivo del gráfico que se coloca en la parte superior.
- **Etiquetas de datos**: símbolo dentro del gráfico que representa un solo valor dentro de la hoja de Excel, es decir, que su valor viene de una celda.
- **Series de datos**: 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**: 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:** área delimitada por los ejes, donde se incluyen todas las series de datos.

- **Líneas de la cuadrícula**: líneas opcionales que extienden los valores de los ejes de manera que faciliten su lectura e interpretación.
- **Título de eje**: es un texto descriptivo que se alinea automáticamente al eje correspondiente.
- **Leyenda**: cuadro que ayuda a identificar los colores asignados a las series de datos.

## 1.2 Análisis de soluciones existentes

***Generador Dinámico de Reportes v2.0 (GDR v2.0)***

El Generador Dinámico de Reportes en su versión 2.0 es una aplicación multiplataforma 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 gestor de bases de datos que utilicen ya sea MySQL, SQLite o PostgreSQL. Su desarrollo está basado en tecnología web, considerándose una ventaja para las organizaciones, pues permite su acceso desde cualquier estación de trabajo conectada al servidor. Fue implementado con tecnologías como: PHP, JavaScript, Symfony y Ext-JS. GDR v2.0, para generar los reportes, consume un servicio web que brinda el Servidor Dinámico de Reportes v1.0 (SDR v1.0), quien utiliza a su vez como motor para la generación de reportes a JasperReport v5.1.

Para diseñar los reportes GDR v2.0 cuenta con un módulo de diseño. Este está compuesto por un conjunto de componentes que permiten el diseño y construcción de un reporte. Dentro de esta gama de componentes está el de graficado, que permite la inclusión de gráficas de pastel, área, línea y barras dentro del diseño. Para lograr la generación de estas, el componente utiliza el JasperReport v5.1, construyendo para ello un XML con los datos necesarios que el motor utiliza para generar las mismas. Además, el componente se basa en las funcionalidades de IReport v5.1 para la generación de las gráficas soportadas por JasperReport v5.1. Este componente fue desarrollado utilizando la arquitectura Modelo-Vista-Controlador, así como las tecnologías que utiliza GDR v2.0 en su implementación como son ExtJS 3.4 y Symfony 2.0.

***Servidor Dinámico de Reportes v1.0 (SDR v1.0)***

Es un software desarrollado con tecnologías libres basado en una arquitectura orientada a servicios, que puede integrarse fácilmente a las aplicaciones encargadas de diseñar reportes. Garantiza la gestión y exportación de los reportes y sub-reportes a múltiples formatos, además permite la programación de tareas automáticas, el cálculo estadístico del uso de los reportes, y brinda una interfaz de usuario para la administración visual de los recursos del servidor. Posibilita que aplicaciones con interfaces basadas en JSON o XML (AJAX) puedan conectarse, publicar y consumir recursos, lo que permite que el servicio sea

utilizado por distintos clientes escritos en diferentes lenguajes, corriendo en diversas plataformas y dispositivos. Está desarrollado completamente en Java utilizando las librerías de código abierto JasperReport v5.1. (Montes Oliver, y otros, 2015)

***JasperReport v5.1***

JasperReport v5.1 es una librería de código abierto escrita en Java con el propósito de ayudar a los desarrolladores a generar reportes para las aplicaciones. Dado que no es una herramienta independiente no se puede instalar por sí sola, la misma debe integrarse a una aplicación en java para su empleo. JasperReport v5.1 es capaz de generar reportes profesionales incluyendo imágenes y gráficos. Algunas de sus mejores características incluyen: tener un posicionamiento de los reportes, siendo capaz de presentar datos en forma de textos o gráficos (permite representar 20 tipos de gráficos diferentes), permite además diferentes orígenes de datos, así como generar marca de agua y exportar estos reportes en varios formatos de archivos como PDF, XML, HTML, CSV, XLS, RTF y TXT. (Heffelfinger, 2006)

***IReport v5.1***

Es una herramienta visual que sirve para generar ficheros XML (plantillas de informes) utilizando la herramienta de generación de informes JasperReport v5.1. IReport v5.1 provee a los usuarios de JasperReport v5.1 una interfaz visual para construir reportes. También permite que los usuarios corrijan visualmente informes complejos con cartas, imágenes y sub-informes. Está escrito en Java y es totalmente gratuito. GDR v2.0 basa varias de sus funcionalidades en esta herramienta, entre ellas el graficado de datos en los reportes. IReport v5.1 permite gestionar los 20 tipos de gráficos que soporta JasperReport v5.1 en los reportes. (Herrera, 2013)

A partir del estudio realizado se decidió desarrollar la versión 1.1 del componente de graficado para GDR v2.0, tomando como base para el desarrollo el componente anterior y añadiendo a esta, nuevas funcionalidades, teniendo como referencia la arquitectura sobre la cual fue implementado. De la herramienta IReport v5.1 se tendrá en cuenta, durante el levantamiento de requisitos, las funcionalidades que brinda para la generación de gráficos. Se selecciona por la relación que tiene con GDR v2.0 y el componente de graficado anteriormente desarrollado el motor de reportes JasperReport v5.1 junto a SDR para la construcción de las gráficas en los reportes.

Teniendo en cuenta que JasperReport v5.1 soporta soporta 20 tipos de gráficos, además, sabiendo que ya existen en GDR v2.0 4 tipos de estos, se decide que la nueva versión del componente a desarrollar debe

mantener los ya existentes (área, línea, pastel y barras) e incorporar los 16 tipos de gráficos restantes que a continuación se describen.

## 1.3 Tipos de gráficas para la representación de información

***Pastel 3D***

Un gráfico de pastel 3D presenta la información de una manera muy atractiva y, a la vez, consigue que las personas que lo ven tengan una idea clara de la relación existente entre las diferentes series de datos. Se utiliza para representar una serie de valores con respecto a un total, mostrando en cada sector la parte proporcional del total. (Microsoft, 2016)

Se emplea para representar las distribuciones o proporciones de poblaciones; es empleado para representar resultados de encuestas de opinión y de mercadeo. Representa las partes o secciones en que se distribuye un todo; muchas veces este tipo de gráfico representa porcentajes en la distribución de un hecho basado en un criterio especial de interés para el investigador.

Se utiliza un gráfico de pastel cuando:

- Sólo tenga una serie de datos que desee representar.
- Ninguno de los valores que desee representar sea negativo.
- Ninguno de los valores que desee representar sea igual a cero.
- No tiene más de siete categorías.
- Las categorías representan partes de un todo en el gráfico circular.

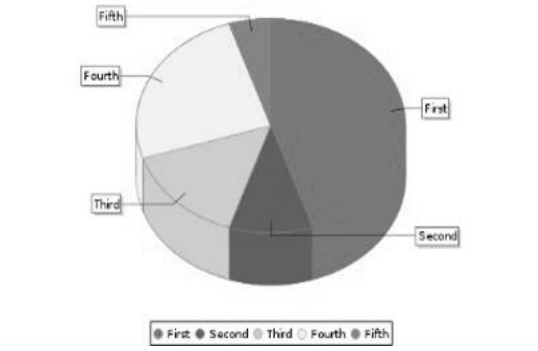



Fig. 2 Gráfica de pastel 3D

***Barras 3D***

Los gráficos de barras son una buena solución para representar una o varias categorías de datos, sobre todo si estas incluyen subcategorías. Permiten expresar de forma visual la diferencia entre los puntos de datos de cada una de las categorías. (Microsoft, 2016)

En los gráficos de barras, las categorías se suelen organizar a lo largo del eje horizontal, mientras que los valores lo hacen a lo largo del vertical.

Los datos se pueden organizar en filas, donde cada una de las filas representa una barra en el gráfico. Las columnas son las etiquetas o las clasificaciones de cada fila. En las demás columnas deben incluirse datos numéricos. Se puede asignar un nombre a cada una de las columnas que contienen datos en la hoja de cálculo.

Considere la posibilidad de utilizar un gráfico de barras cuando:

- Tenga una sola serie de datos que desee representar.
- Sus datos incluyan valores positivos y cero.
- Desee comparar los datos de varias categorías en paralelo.

Una gráfica de barras 3D muestra los mismos datos que una gráfica de barras 2D con datos mediante una perspectiva 3D. No se emplea un tercer eje de valores (eje de profundidad).

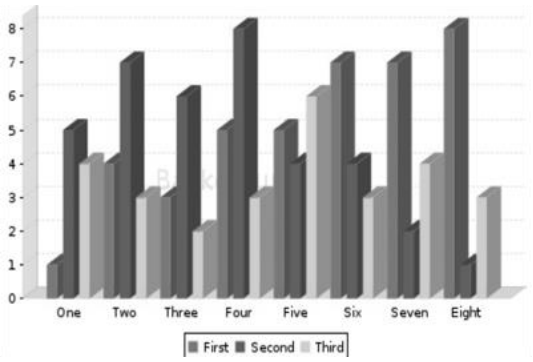



Fig. 3 Gráfica de barras 3D.

***Barras XY***

Los gráficos de barras XY se emplean generalmente para comparar magnitudes en varias categorías, para ver la evolución en el tiempo de la magnitud de una categoría o para una mezcla de ambos objetivos. Los datos se representan por medio de rectángulos cuya altura es proporcional al valor de la magnitud graficada mientras que el ancho es el mismo para todos los elementos en la gráfica. (Figueroa Zarraga, 2014)

En este gráfico se representa en el eje de las abscisas (X), las distintas categorías de la variable que en este caso son sustituidas por valores, y en el eje de las ordenadas (Y), la frecuencia absoluta o relativa.

La gráfica de barras XY presenta la ventaja de que su construcción es muy simple, con un mínimo de cálculos y un nivel de complejidad muy elemental.

Con los datos representados en los gráficos de barras XY se puede interpretar rápidamente y de manera visual la información, facilitando su posterior análisis.

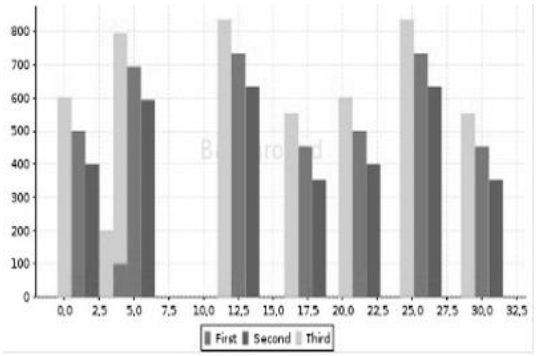



Fig. 4 Gráfico de barras xy.

***Barras apiladas y barras apiladas 3D***

En los gráficos de barras apiladas varias series se apilan verticalmente. Si sólo hay una serie en el gráfico, el gráfico de barras apilada se mostrará igual que un gráfico de barras. Se usan normalmente para mostrar comparaciones entre grupos, donde cada grupo contenga varias series de datos. (Microsoft, 2016)

Estos gráficos colocan las series una encima de otra para crear una barra apilada. Brindan la opción de separar el gráfico de barras apiladas en varios conjuntos de pilas para cada categoría. Las series apiladas agrupadas se muestran unas junto a las otras. Se puede tener cualquier número de series apiladas agrupadas en un gráfico.

Los gráficos de barras apiladas presentan entre otras las siguientes características:

- Representa los datos de dos o más series o conjuntos de datos.
- Cada serie se representa en un mismo color.
- Cada barra representa una categoría de la variable, y se divide en segmentos que representan cada una de las series de datos.

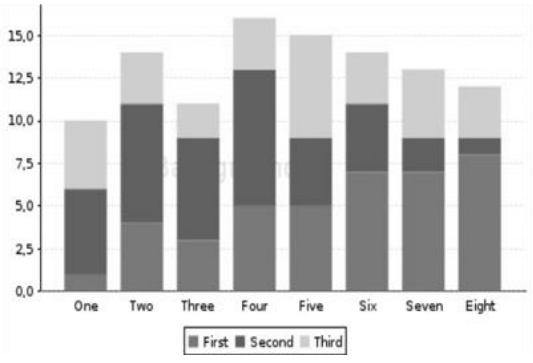



Fig. 5 Gráfico de barras apiladas.

Los gráficos de barras apiladas muestran la relación de elementos individuales con el conjunto. Un gráfico de barras apiladas en 3D muestra rectángulos horizontales en formato 3D; no presenta los datos en tres ejes.

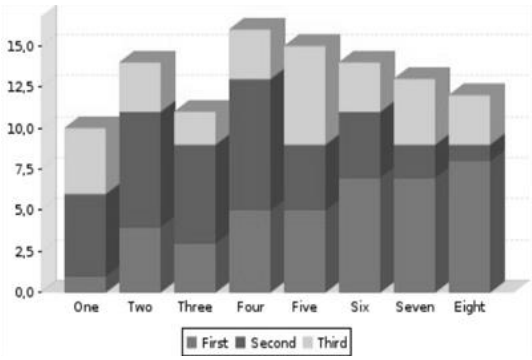



Fig. 6 Gráfico de barras apiladas 3D.

***Líneas XY***

Los gráficos de líneas muestran los datos en forma de puntos y todos los puntos de la misma serie se unen mediante una línea, de ahí su nombre. (Instituto de Tecnologías Educativas, 2016)

Se pueden trazar datos que se organizan en columnas o filas. Los gráficos de líneas pueden mostrar datos continuos en el tiempo, establecidos frente a una escala común y, por tanto, son ideales para mostrar tendencias en datos a intervalos iguales como meses, trimestres, años o ejercicios fiscales. En un gráfico de líneas XY, los datos de categoría son sustituidos por valores numéricos y se distribuyen uniformemente en el eje horizontal, y todos los datos de valor asociados a las categorías se distribuyen uniformemente en el eje vertical. Si se tienen más de diez series numéricas, no es recomendado utilizar un gráfico de líneas XY.

Un ejemplo estándar puede ser la forma en que se desarrolla a lo largo del tiempo un valor de una empresa determinada en el mercado de valores. Sin embargo, no es necesario que el valor del eje X sea el tiempo, sino que se puede usar cualquier dato que se comporte como una función con respecto a la variable en el eje X. Los gráficos de líneas XY enfatizan el flujo de tiempo y el ritmo del cambio en lugar de la cantidad de cambios. Se pueden usar varias escalas en el eje Y cuando se desea comparar varias líneas con rangos de valor significativamente distintos.

Los gráficos de líneas XY son una buena solución para representar datos numéricos de manera visual. Resultan especialmente útiles para expresar los cambios que se producen en los valores entre las distintas categorías de datos.

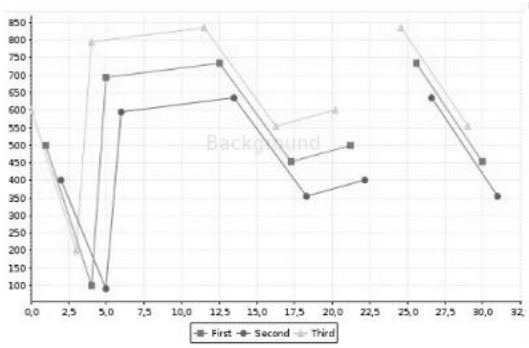



Fig. 7 Gráfico de líneas xy.

***Área XY***

Los gráficos de áreas XY muestran las series como un conjunto de puntos ubicados en coordenadas x,y conectados por una línea, con un área rellena por debajo de la línea. Una gráfica de área evalúa las contribuciones a un total en el tiempo. Las gráficas de áreas XY muestran múltiples series de tiempo apiladas en el eje Y versus intervalos de tiempo con igual separación en el eje X. Cada línea de la gráfica representa la suma acumulada, de manera que usted puede ver la contribución de cada serie a la suma y cómo cambia la composición de la suma en el tiempo. (Microsoft, 2016)

Hay que tener en cuenta que el área de algunos valores puede opacar u ocultar el de otros. Es por ello que se debe ser muy cuidadoso al usar estás gráficas, conocer los valores exactos es una prioridad.

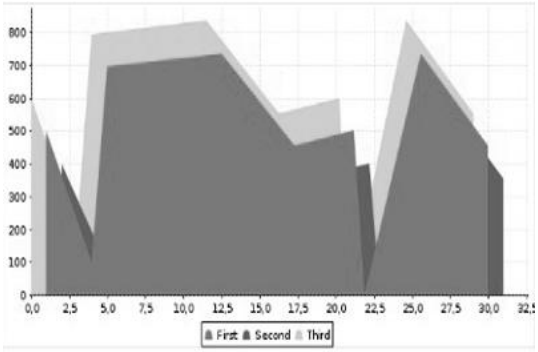



Fig. 8 Gráfico de área xy.

***Áreas Apiladas***

En los gráficos de áreas apiladas varias series se apilan verticalmente. Si sólo hay una serie en el gráfico, el gráfico de área apilada se mostrará igual que un gráfico de áreas. Muestra todos los valores apilando las distintas series unas encima de las otras para cada categoría; no es adecuado para comparar los datos en valores absolutos, pues los valores Y no se dibujan como absolutos, pero permiten mostrar la contribución de cada valor a su categoría y ningún valor queda oculto. (OpenOffice, 2015)

En la fig. 9 se muestra un ejemplo de gráfico de área apilada donde los datos se adaptan perfectamente a este tipo de gráfica porque en él se pueden mostrar los totales para todas las series y la proporción con la que cada serie contribuye al total.

Si el gráfico de área apilada contiene espacios, es posible que el conjunto de datos incluya valores vacíos, que se mostrarán como una sección vacía en el gráfico. Muestra la tendencia de la contribución de cada valor a lo largo del tiempo u otros datos de una categoría.

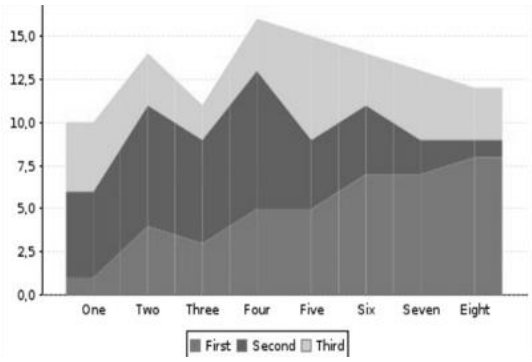



Fig. 9 Gráfico de área apilada.

***Dispersión***

El gráfico de dispersión permite analizar si existe algún tipo de relación entre dos variables**.** Por ejemplo, puede ocurrir que dos variables estén relacionadas de manera que al aumentar el valor de una se incremente el de la otra. En este caso se hablaría de la existencia de una correlación positiva. También podría ocurrir que al producirse una en un sentido la otra derive en el sentido contrario; por ejemplo, al aumentar el valor de la variable x se reduzca el de la variable y. Entonces, se estaría ante una correlación negativa. Si los valores de ambas variables se revelan independientes entre sí, se afirmaría que no existe correlación. (Atlantic International University, 2016)

Es una herramienta gráfica que permite realizar análisis con el siguiente:

Por ejemplo, calculando el coeficiente de correlación entre dos variables, permite cuantificar el grado de relación entre ambas, así como su signo. El valor de este coeficiente puede estar comprendido entre −1 y +1. Cuando toma un valor próximo a −1, la correlación es fuerte y negativa. Si el valor es cercano a +1, la correlación es fuerte y positiva. Si el coeficiente de correlación lineal presenta un valor próximo a 0, la correlación es débil. Un coeficiente de 0 indicaría independencia total entre ambas variables. A su vez, un coeficiente de correlación lineal de +1 ó de -1 señalaría que entre ambas variables hay dependencia funcional, positiva o negativa según el signo del coeficiente.

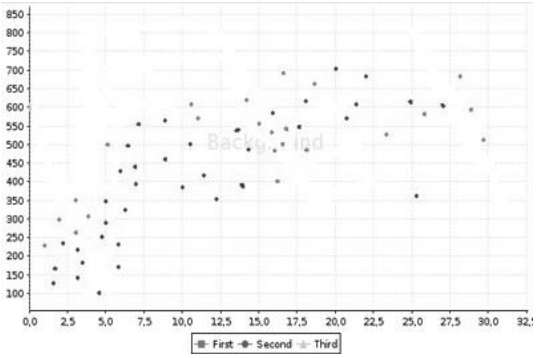



Fig. 10 Gráfico de dispersión.

***Burbujas***

Un gráfico de burbujas es una variación de un gráfico de dispersión en el que se sustituyen los puntos de datos con burbujas y se representa una dimensión adicional de los datos en el tamaño de las burbujas. Al igual que un gráfico de dispersión, un gráfico de burbujas no utiliza un eje de categorías, y los ejes horizontales y verticales son ejes de valores. (Microsoft, 2016)

Se considera la posibilidad de utilizar un gráfico de burbujas si los datos incluyen tres valores ya que estos son los necesarios para cada burbuja. Estos valores pueden estar ordenados en filas o columnas, pero deben estar en el siguiente orden: valor de x, valor de y, y a continuación el valor de z.

Los gráficos de burbujas son los más indicados para responder a las siguientes preguntas:

- ¿Cómo hacer para resaltar un valor en particular que es diferente del resto?
- ¿Se tiene un valor muy grande o muy pequeño y se quiere comparar con otros valores en una visualización?

Resultan un método ineficaz para mostrar los datos en los que no se pueda percibir claramente las diferentes áreas. Si los valores que se están usando sólo tienen una pequeña variación, entonces se recomienda usar un gráfico de barras o de columnas. Pero si por el contrario se tiene un valor atípico, entonces el gráfico de burbujas ayuda a mostrarlo mejor.

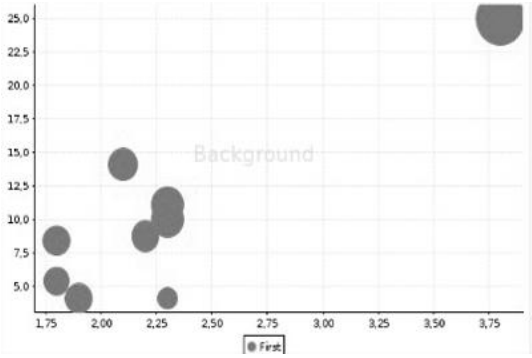



Fig. 11 Gráfico de burbujas.

***Series de tiempo***

Las series de tiempo se refieren a datos estadísticos que se recopilan, observan o registran en intervalos de tiempo regulares (diario, semanal, semestral, anual, entre otros). El término serie de tiempo se aplica, por ejemplo, a datos registrados en forma periódica que muestran, por ejemplo, las ventas anuales totales de almacenes. (Universidad de Sonora, 2015)

Una gráfica de series de tiempo se puede utilizar para evaluar patrones y comportamiento de datos en el transcurso del tiempo. Muestra observaciones en el eje Y con respecto a intervalos de tiempo con igual separación en el eje X. Es decir, una serie de tiempo se representa por medio de una gráfica de líneas sobre cuyo eje horizontal se representan los períodos y en cuyo eje vertical se representan los valores de la serie de tiempo.

Las gráficas de series de tiempo son especialmente útiles para comparar patrones de datos de diferentes grupos.

Tiene como objetivos, entre otros:

- Determinar si se presentan ciertos patrones o pautas no aleatorias.
- Aislar y entonces estudiar sus componentes a fin de proporcionar claves para movimientos futuros.
- Hace posible pronosticar los movimientos futuros, así como otros aspectos que estén sincronizados.

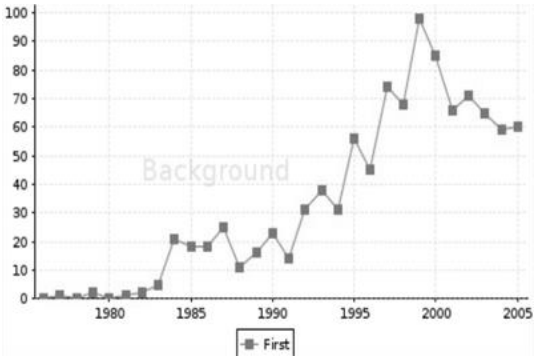



Fig. 12 Gráfico de serie de tiempo.

***Alto y Bajo***

Un gráfico open-high-low-close (también gráfico OHLC) es un tipo de gráfico que se suele utilizar para ilustrar los movimientos en el precio de un instrumento financiero a través del tiempo. Cada línea vertical en el gráfico muestra el rango de precios (los precios más altos y más bajos) en una unidad de tiempo, por ejemplo, un día o una hora. Las marcas en cada lado de la línea indican el precio de apertura (por ejemplo, para un gráfico de barras diarias este sería el precio de salida para ese día) a la izquierda, y el precio de cierre de ese período de tiempo a la derecha. Las barras se pueden mostrar en diferentes tonalidades dependiendo de si los precios aumentaron o disminuyeron en ese período. (Noreña Ossa, 2011)

Una variante sencilla en el gráfico OHLC es el gráfico de alto-bajo-cierre HLC que identifica el rango de acción del precio de la unidad de tiempo (alto – bajo) y el resultado final de la acción del precio de la unidad de tiempo (cierre).

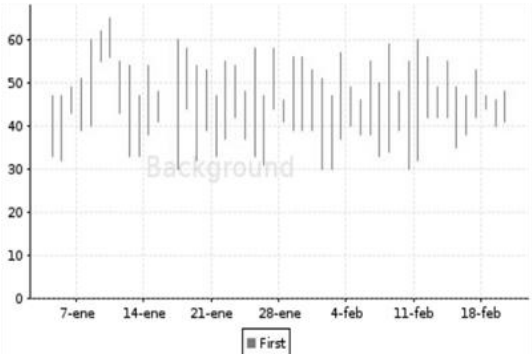



Fig. 13 Gráfico alto y bajo.

***Vela***

De modo similar al gráfico de barras, el gráfico de velas muestra información de precio de apertura, cierre, máximo y mínimo. Además, utiliza colores para mostrar si el precio de cierre fue mayor o menor que el precio de apertura.

El cuerpo de la vela muestra la diferencia entre el precio de apertura y cierre, y los correspondientes valores. El color del cuerpo depende de si el precio de cierre fue mayor o menor al precio de apertura. Si el precio de cierre fue mayor al precio de apertura, generalmente se representa al cuerpo de la vela con un color verde; si el precio de cierre fue menor al precio de apertura, se representa el cuerpo de la vela con un color rojo (estos colores pueden variar, por ejemplo, pueden ser negro y blanco).

Arriba y abajo del cuerpo de la vela puede haber dos líneas que indican los precios máximos y mínimos del período en cuestión; estas líneas se pueden denominar *sombras*. Si no aparecen alguna de estas líneas o las dos, esto indica que uno de los precios de apertura o de cierre (o los dos) fueron el máximo o mínimo del período. (Americanbulls, 2016)

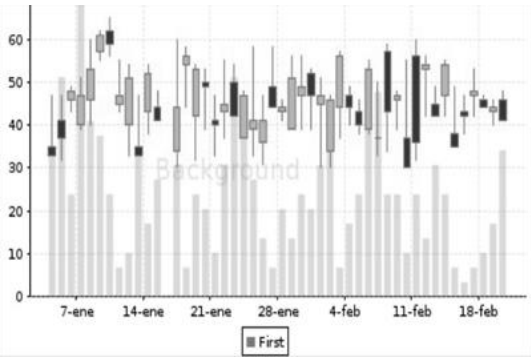



Fig. 14 Gráfico de vela.

***Diagrama de Gantt***

El diagrama de Gantt es una herramienta que se emplea para planificar y programar tareas a lo largo de un período determinado de tiempo. Gracias a una fácil y cómoda visualización de las acciones a realizar, permite dar seguimiento y control del progreso de cada una de las etapas de un proyecto. Reproduce gráficamente las tareas, su duración y secuencia, además del calendario general del proyecto y la fecha de finalización prevista. El diagrama de Gantt es una útil herramienta gráfica cuyo objetivo es exponer el tiempo de dedicación previsto para diferentes tareas o actividades a lo largo de un tiempo total determinado. (Yasen, 2014)

Los diagramas de Gantt se han convertido en una herramienta básica en la gestión de proyectos de todo tipo, con la finalidad de representar las diferentes fases, tareas y actividades programadas como parte de un proyecto o para mostrar una línea de tiempo en las diferentes actividades.

A pesar de esto, el diagrama de Gantt no indica las relaciones existentes entre actividades. Dada la posición de cada tarea a lo largo del tiempo, se pueden identificar dichas relaciones e interdependencias. Por esta

razón, para la planificación del desarrollo de proyectos complejos (superiores a 25 actividades) se requiere además el uso de técnicas basadas en redes de precedencia. El diagrama de Gantt, sin embargo, resulta útil para la relación entre tiempo y carga de trabajo.

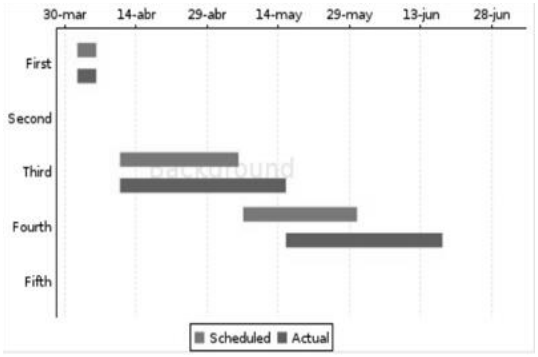



Fig. 15 Diagrama de Gantt.

***Sector***

Los gráficos de sectores permiten observar los datos de una variable en forma global, haciendo una comparación inmediata del porcentaje o la frecuencia de cada una de sus categorías, facilitando el reconocimiento del aporte de cada una de ellas al total de la variable. Generalmente este tipo de gráfico se utiliza para la descripción de variables categóricas, aunque en algunas ocasiones se puede aplicar a las variables de escala. (Martínez, 2015)

El gráfico de sectores simple se emplea para resumir las categorías de una o varias variables dentro de un sólo gráfico; el tamaño de cada sector representa la frecuencia, el porcentaje o una función de resumen.
Un diagrama de sectores se puede utilizar para todo tipo de *variables*, pero se usa frecuentemente para las variables cualitativas**.**

Los datos se representan en un círculo, de modo que el ángulo de cada sector es proporcional a la frecuencia absoluta correspondiente.

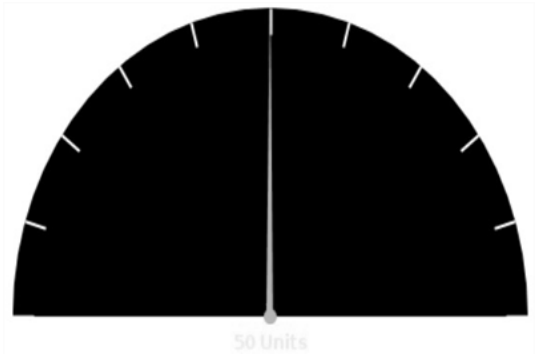



Fig. 16 Gráfico de sector.

***Termómetro***

Un gráfico de termómetro es un gráfico del progreso hacia una meta en particular. Por ejemplo, podría ser el avance hacia un determinado objetivo de recaudación de fondos para una organización benéfica. Una característica de estos gráficos es que se actualizan periódicamente, por lo que el software debe permitir esto. A pesar de la cantidad de información en un gráfico de termómetro este es relativamente ligero.

Este tipo de gráfico permite visualizar indicadores de cumplimiento, medir el alcance que se obtuvo en los presupuestos diseñados en el área de ventas, costos, contabilidad, compras, etc., ofreciéndonos una percepción gráfica muy rápida de la situación que se tiene. (Copyright , 2016)


Fig. 17 Gráfico de termómetro.

## 1.4 Metodología, herramientas y tecnologías

Para la selección de las herramientas, metodologías y tecnologías a emplear en el desarrollo de la solución se tuvieron en cuenta las definidas en el marco del proyecto GDR v2.0 y que a su vez fueron utilizadas en el desarrollo del componente de graficado desarrollado anteriormente.

### *1.4.1 Metodología de desarrollo de software OpenUP*

OpenUP es una metodología de Proceso Unificado que aplica enfoques iterativos e incrementales dentro de un ciclo de vida estructurado. Utiliza una filosofía ágil que se enfoca en la naturaleza de colaboración para el desarrollo de software, que contiene el conjunto mínimo de prácticas que ayudan a un equipo de desarrollo de software a realizar un producto de alta calidad. (LinkedIn, 2014)

Es una metodología apropiada para proyectos pequeños, de bajos recursos y de corta duración, lo que permite disminuir las probabilidades de fracaso e incrementar las probabilidades de éxito. Además, evita la elaboración de documentación, diagramas e iteraciones innecesarios. Al ser una metodología ágil, centra su atención en el cliente, permite el desarrollo iterativo e incremental, es dirigido por casos de uso y centrado en la arquitectura.

### *1.4.2 El lenguaje unificado de modelado (UML v2.0)*

UML “Lenguaje Unificado de Modelado”. Se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de software. (Krall, Cesar, 2016)

### *1.4.3 Herramienta de modelado*

**Visual Paradigm for UML v8.0**

Este es un software de modelado UML que permite analizar, diseñar, codificar, probar y desplegar. Permite la modelación de todo tipo de diagramas UML, genera código fuente a partir de dichos diagramas y también posibilita la elaboración de documentos. Se decide utilizar esta herramienta ya que el equipo de desarrollo tiene experiencia acumulada en el uso de ella para la realización de los diagramas que propone la metodología a emplear.

### *1.4.4 Lenguajes a emplear en el desarrollo de la solución*

**XML (“Lenguaje de Marcas Extensible”)**

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.

Cuando se desarrolla un programa con interfaz gráfica es necesario organizar todas las imágenes de manera que se vayan cargando a medida que se necesiten, y XML es de gran ayuda en estos casos ya que permite agruparlas, etiquetarlas, especificar su ubicación y relacionarlas con otros datos, según las necesidades de los diseñadores. (Lamarca Lapuente, 2013)

**PHP 5.3**

PHP es un lenguaje de script interpretado en el lado del servidor utilizado para la generación de páginas web dinámicas, embebido en páginas HTML y ejecutado en el servidor, lo que permite acceder a los recursos que tenga el servidor como por ejemplo a una base de datos. El programa PHP es ejecutado en el servidor y el resultado es enviado al navegador. El resultado es normalmente una página HTML. (Figueroa, Viridiana, 2015)

Al ser un lenguaje libre dispone de características que lo convierten en la herramienta ideal para la creación de páginas web dinámicas:

- Soporte para varios gestores de bases de datos: MySQL, PostgreSQL, SQLite, Oracle y SQL Server, entre otras.
- Integración con varias bibliotecas externas, permite generar documentos en PDF y analizar código XML.

**JavaScript**

Es un lenguaje de programación que se puede utilizar para construir sitios web y para hacerlos más interactivos. Aunque comparte muchas de las características y de las estructuras del lenguaje Java, fue desarrollado independientemente. El lenguaje JavaScript puede interactuar con el código HTML, permitiendo a los programadores web utilizar contenido dinámico.

Tiene como características principales las siguientes:

- Es interpretado (no compilado) por el cliente.
- Está basado en objetos.
- Su código se integra en las páginas HTML, incluido en las propias páginas.
- No es necesario declarar los tipos de variables que van a utilizarse.
- Las referencias a objetos se comprueban en tiempo de ejecución, por lo tanto, no se compila.
- No permite escribir automáticamente en el disco duro.

**Marcos de trabajo**

Un *framework* es un conjunto estandarizado de conceptos, prácticas y criterios para hacer frente a un tipo común de problema, que puede ser usado para ayudar a resolverlo de forma rápida y eficaz. El objetivo de los *frameworks* es proporcionar una estructura común, de modo que los desarrolladores no tienen que hacer el código de cero cada vez y pueden volver a utilizar la gran mayoría. (Gómez, 2014)

**Symfony v2.0**

Symfony es un *framework* diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación.

Características fundamentales de Symfony: (LibrosWeb.es, 2015)

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

**ExtJS v3.4**

ExtJS v3.4 es una biblioteca de JavaScript para la creación de aplicaciones enriquecidas del lado del cliente. Sus características principales son: gran desempeño, componentes de interfaz de usuario personalizables, con buen diseño y documentación. Usa tecnologías como AJAX, DHTML y DOM.

### *1.4.5 Entorno de desarrollo Integrado*

Un Entorno de Desarrollo Integrado, traducido del inglés *Integrated Development Environment* (IDE) es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un sólo lenguaje de programación o bien, puede utilizarse para varios. Los IDE proveen de un marco de trabajo amigable para la mayoría de los lenguajes de programación. Un IDE puede funcionar como un sistema en tiempo de ejecución, en donde se permite utilizar el lenguaje de programación en forma interactiva, sin necesidad de trabajo orientado a archivos de texto.

**NetBeans IDE v8.1**

Es un entorno de desarrollo integrado (IDE), modular, de base estándar (normalizado), escrito en el lenguaje de programación Java. El proyecto NetBeans consiste en un IDE de código abierto y una plataforma de aplicación, las cuales pueden ser usadas como una estructura de soporte general (*framework*) para compilar cualquier tipo de aplicación. (Oracle, 2016)

NetBeans permite crear aplicaciones Web con PHP 5, un potente *debugger* integrado y además posee soporte para Symfony, framework escrito en PHP.

### *1.4.6 Servidor de aplicaciones web*

Un servidor web es un programa que sirve para atender y responder a las diferentes peticiones de los navegadores, proporcionando los recursos que soliciten usando el protocolo HTTP o el protocolo HTTPS.

**Apache v2.4.7**

Apache es un servidor web multiplataforma, rápido, continuamente actualizado. Permite la creación y publicación de documentos PHP con una estabilidad comprobada. Es una tecnología gratuita de código abierto. Trabaja con PHP y otros lenguajes script, incluye todo el soporte que se necesita para tener páginas dinámicas.

## Conclusiones del capítulo

Al término de este capítulo se concluye lo siguiente:

El estudio de las soluciones existentes para el graficado en los reportes, permitió identificar la necesidad de desarrollar la versión 1.1 del componente de graficado para GDR v2.0 que incluya todas las gráficas que genera JasperReport v5.1. Teniendo en cuenta las características de la aplicación a desarrollar se va a utilizar como metodología OpenUp, y como herramientas y tecnologías: Visual Paradigm v8.0, UML v2.0, JavaScript y PHP v5.3, NetBeans v8.1, Apache v2.4.7 y los frameworks ExtJS v3.4 y Symfony v2.0.

# Capítulo II: Análisis y diseño de la solución propuesta

En este capítulo se realiza la descripción del componente a desarrollar, para lograr una mayor claridad y comprensión por parte de los desarrolladores. Los puntos que se abordarán son los siguientes: descripción del modelo de dominio, especificación de los requisitos funcionales y no funcionales, representación de los actores y casos de uso a través de un diagrama de casos de uso del sistema, la descripción textual de cada uno de ellos, así como los patrones de diseño GoF y GRASP que serán utilizados y los diagramas de clases del diseño.

## 2.1 Modelo de dominio

Un modelo del dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. También se les denomina modelos conceptuales (término utilizado en la primera edición del libro de Larman), modelo de objetos del dominio y modelos de objetos de análisis. Utilizando la notación UML, un modelo del dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. (Larman, 2003)

Pueden mostrar:

- Objetos del dominio o clases conceptuales.
- Asociaciones entre las clases conceptuales.
- Atributos de las clases conceptuales.

Se realizó el modelo de dominio del componente de graficado v1.1 para proporcionar un mejor entendimiento de los principales conceptos que se manejan en el negocio de la manera más cómoda y entendible.

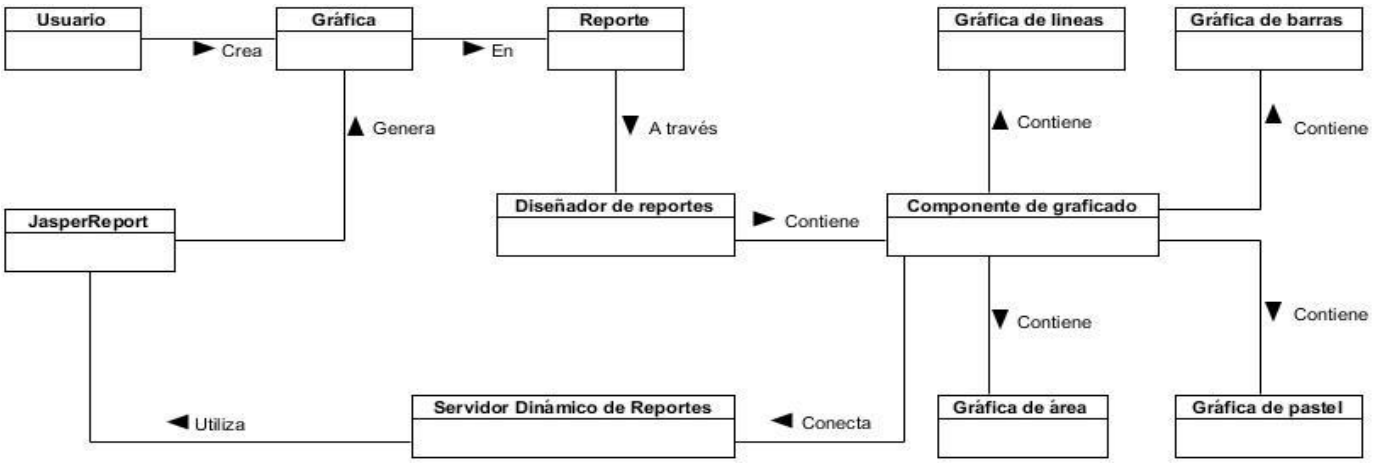



Fig. 18 Modelo de dominio.

Descripción de las clases del dominio:

- **Usuario:** Persona encargada de diseñar los reportes con GDR v2.0.
- **Gráfica:** Representación visual que incluye figuras y/o signos para comunicar uno o una serie de conceptos.
- **Diseñador de reportes:** Módulo del GDR v2.0 que permite diseñar los reportes que luego son generados.
- **Reporte:** Informe o documento que transmite una información mediante elementos como: textos, tablas, diagramas, imágenes, figuras y gráficas.
- **Componente de graficado:** Componente que permite adicionar, modificar y eliminar gráfica a los reportes.
- **Servidor Dinámico de Reportes:** Sistema independiente basado en una arquitectura orientada a servicios, que puede integrarse fácilmente a las aplicaciones encargadas de diseñar los reportes.
- **JasperReport v5.1:** 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.
- **Gráfica de barras:** Tipo de gráfica que se puede incluir en el diseño de un reporte.
- **Gráfica de línea:** Tipo de gráfica que se puede incluir en el diseño de un reporte.
- **Gráfica de pastel:** Tipo de gráfica que se puede incluir en el diseño de un reporte.
- **Gráfica de área:** Tipo de gráfica que se puede incluir en el diseño de un reporte.

## 2.2 Especificación de los requisitos del sistema

Los requisitos funcionales son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que este debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares. En algunos casos, los requerimientos funcionales de los sistemas también pueden declarar explícitamente lo que el sistema no debe hacer. Los requerimientos 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 requerimientos. (Jason, 2009)

Partiendo del análisis realizado en el capítulo anterior referente a los tipos de gráficos que se tendrán en cuenta en el desarrollo de la solución se procedió al levantamiento de requisitos de conjunto con el cliente, por lo que la solución a desarrollar debe cumplir con los requisitos funcionales y no funcionales que a continuación se describen.

### *2.2.1 Requisitos funcionales*

**RF1 Adicionar nueva gráfica de dispersión al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de dispersión en un reporte.

**RF2 Adicionar nueva gráfica de burbujas al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de burbujas en un reporte.

**RF3 Adicionar nueva gráfica de línea XY al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de línea XY en un reporte.

**RF4 Adicionar nueva gráfica de área XY al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de área XY en un reporte.

**RF5 Adicionar nueva gráfica de barras XY al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de barras XY en un reporte.

**RF6 Adicionar nueva gráfica de pastel 3D al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de pastel 3D en un reporte.

**RF7 Adicionar nueva gráfica de barras 3D al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de barras 3D en un reporte.

**RF8 Adicionar nueva gráfica de barras apiladas al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de barras apiladas en un reporte.

**RF9 Adicionar nueva gráfica de barras apiladas 3D al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de barras apiladas 3D en un reporte.

**RF10 Adicionar nueva gráfica de área apilada al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de área apilada en un reporte.

**RF11 Adicionar nueva gráfica de serie de tiempo al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de serie de tiempo en un reporte.

**RF12 Adicionar nueva gráfica de vela al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de vela en un reporte.

**RF13 Adicionar nuevo diagrama de Gantt al reporte.** Esta funcionalidad permite al usuario insertar un diagrama de Gantt en un reporte.

**RF14 Adicionar nueva gráfica de sector al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de sector en un reporte.

**RF15 Adicionar nueva gráfica de termómetro al reporte.** Esta funcionalidad permite al usuario insertar una gráfica de termómetro en un reporte.

**RF16 Adicionar nueva gráfica alto y bajo al reporte.** Esta funcionalidad permite al usuario insertar una gráfica alto y bajo en un reporte.

**RF17 Modificar gráfica de dispersión en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de dispersión en cuanto a: tamaño, posición, Expresión de la serie, Valor X de la expresión, Valor Y de la expresión, Expresión de etiqueta.

**RF18 Modificar gráfica de burbujas en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de burbujas en cuanto a: tamaño, posición, Expresión de la serie, Valor X de la expresión, Valor Y de la expresión, Valor Z de la expresión.

**RF19 Modificar gráfica de línea XY en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de línea XY en cuanto a: tamaño, posición, Expresión de la serie, Valor X de la expresión, Valor Y de la expresión, Expresión de etiqueta.

**RF20 Modificar gráfica de área XY en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de área XY en cuanto a: tamaño, posición, Expresión de la serie, Valor X de la expresión, Valor Y de la expresión, Expresión de etiqueta.

**RF21 Modificar gráfica de barras XY en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de barras XY en cuanto a: tamaño, posición, Expresión de la serie, Valor X de la expresión, Valor Y de la expresión, Expresión de etiqueta.

**RF22 Modificar gráfica de pastel 3D en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de pastel 3D en cuanto a: tamaño, posición, Expresión clave, Valor X de la expresión, Etiqueta de expresión.

**RF23 Modificar gráfica de barras 3D en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de barras 3D en cuanto a: tamaño, posición, Expresión de la serie, Expresión de la categoría, Valor X de la expresión, Expresión de etiqueta.

**RF24 Modificar gráfica de barras apiladas en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de barras apiladas en cuanto a: tamaño, posición, Expresión de la serie, Expresión de la categoría, Valor X de la expresión, Expresión de etiqueta.

**RF25 Modificar gráfica de barras apiladas 3D en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de barras apiladas 3D en cuanto a: tamaño, posición, Expresión de la serie, Expresión de la categoría, Valor X de la expresión, Expresión de etiqueta.

**RF26 Modificar gráfica de área apilada en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de área apilada en cuanto a: tamaño, posición, Expresión de la serie, Expresión de la categoría, Valor X de la expresión, Expresión de etiqueta.

**RF27 Modificar gráfica de serie de tiempo en el reporte.**

Descripción: Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de serie de tiempo en cuanto a: tamaño, posición, Expresión de la serie, Periodo de tiempo, Valor de la expresión, Expresión de etiqueta.

**RF28 Modificar gráfica de vela en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de vela en cuanto a: tamaño, posición, Expresión de la serie, Fecha de expresión, Expresión alta, Expresión baja, Abrir expresión, Cerrar expresión, Volumen de la expresión.

**RF29 Modificar diagrama de Gantt en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un diagrama de Gantt en cuanto a: tamaño, posición, Expresión de la serie, Tarea expresión, Subtarea expresión, Fecha comienzo expresión, Fecha fin expresión, Porciento expresión.

**RF30 Modificar gráfica de sector en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de sector en cuanto a: tamaño, posición, Valor de la expresión

**RF31 Modificar gráfica de termómetro en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico de termómetro en cuanto a: tamaño, posición, Valor de la expresión.

**RF32 Modificar gráfica alto y bajo en el reporte.** Esta funcionalidad permite al usuario modificar las propiedades de visualización de un gráfico alto y bajo en cuanto a: tamaño, posición, Expresión de la serie, Fecha expresión, Expresión alta, Expresión baja, Abrir expresión, Cerrar expresión, Volumen de la expresión.

**RF33 Visualizar datos de gráfica de dispersión.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de dispersión que haya sido creado anteriormente.

**RF34 Visualizar datos de gráfica de burbujas.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de burbujas que haya sido creado anteriormente.

**RF35 Visualizar datos de gráfica de línea XY.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de línea XY que haya sido creado anteriormente.

**RF36 Visualizar datos de gráfica de área XY**. Esta funcionalidad permite al usuario visualizar los datos de un gráfico de área XY que haya sido creado anteriormente.

**RF37 Visualizar datos de gráfica de barras XY.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de barras XY que haya sido creado anteriormente.

**RF38 Visualizar datos de gráfica de pastel 3D.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de pastel 3D que haya sido creado anteriormente.

**RF39 Visualizar datos de gráfica de barras 3D.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de barras 3D que haya sido creado anteriormente.

**RF40 Visualizar datos de gráfica de barras apiladas.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de barras apiladas que haya sido creado anteriormente.

**RF41 Visualizar datos de gráfica de barras apiladas 3D.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de barras apiladas 3D que haya sido creado anteriormente.

**RF42 Visualizar datos de gráfica de área apilada.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de área apilada que haya sido creado anteriormente.

**RF43 Visualizar datos de gráfica de serie de tiempo.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de serie de tiempo que haya sido creado anteriormente.

**RF44 Visualizar datos de gráfica de vela.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de vela que haya sido creado anteriormente.

**RF45 Visualizar datos de diagrama de Gantt.** Esta funcionalidad permite al usuario visualizar los datos de un diagrama de Gantt que haya sido creado anteriormente.

**RF46 Visualizar datos de gráfica de sector.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de sector que haya sido creado anteriormente.

**RF47 Visualizar datos de gráfica de termómetro.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico de termómetro que haya sido creado anteriormente.

**RF48 Visualizar datos de gráfica alto y bajo.** Esta funcionalidad permite al usuario visualizar los datos de un gráfico alto y bajo que haya sido creado anteriormente.

**RF49 Visualizar en vista previa gráfica de dispersión.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de dispersión que haya sido creado anteriormente.

**RF50 Visualizar en vista previa gráfica de burbujas.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de burbujas que haya sido creado anteriormente.

**RF51 Visualizar en vista previa gráfica de línea XY.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de línea XY que haya sido creado anteriormente.

**RF52 Visualizar en vista previa gráfica de barras XY.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de barras XY que haya sido creado anteriormente.

**RF53 Visualizar en vista previa gráfica de área XY.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de área XY que haya sido creado anteriormente.

**RF54 Visualizar en vista previa gráfica de pastel 3D.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de pastel 3D que haya sido creado anteriormente.

**RF55 Visualizar en vista previa gráfica de barras 3D.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de barras 3D que haya sido creado anteriormente.

**RF56 Visualizar en vista previa gráfica de barras apiladas.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de barras apiladas que haya sido creado anteriormente.

**RF57 Visualizar en vista previa gráfica de barras apiladas 3D.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de barras apiladas 3D que haya sido creado anteriormente.

**RF58 Visualizar en vista previa gráfica de área apilada.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de área apilada que haya sido creado anteriormente.

**RF59 Visualizar en vista previa gráfica de serie de tiempo.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de serie de tiempo que haya sido creado anteriormente.

**RF60 Visualizar en vista previa gráfica de vela.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de vela que haya sido creado anteriormente.

**RF61 Visualizar en vista previa diagrama de Gantt.** Esta funcionalidad permite al usuario visualizar en vista previa un diagrama de Gantt que haya sido creado anteriormente.

**RF62 Visualizar en vista previa gráfica de sector.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de sector que haya sido creado anteriormente.

**RF63 Visualizar en vista previa gráfica de termómetro.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico de termómetro que haya sido creado anteriormente.

**RF64 Visualizar en vista previa gráfica alto y bajo.** Esta funcionalidad permite al usuario visualizar en vista previa un gráfico alto y bajo que haya sido creado anteriormente.

### *2.2.2 Requisitos no funcionales*

Los requisitos no funcionales son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo sobre el proceso de desarrollo y estándares. Los requerimientos no funcionales a menudo se aplican al sistema en su totalidad. Normalmente apenas se aplican a características o servicios individuales del sistema. Los requisitos no funcionales, como su nombre sugiere, son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema. (Nobelo Can, 2010)

Los requisitos no funcionales definidos para el componente de graficado v1.1 se basan en los definidos en el marco de proyecto del GDR v2.0.

**Requisitos no funcionales**

***Software requerido para desplegar y utilizar la aplicación:***

**RNF1 PC Servidor de 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, Debian 4 GNU/Linux o superior.
- Paquetes: apache2, php5, libapache2-mod-php5, php5-cli, php5-mysql, php5-pgsql, php5-sqlite, php5-sybase, php5-xsl, php5-gd y php-apc.

**RNF2 PC Cliente**

La PC cliente debe cumplir con los siguientes requisitos de software:

- Navegador web Firefox v32.

***Hardware requerido para desplegar y utilizar la aplicación:***

**RNF3 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 520MB.

**RNF4 PC Servidor de Aplicación**

- Ordenador Dual Core o superior, con 1.7 GHz de velocidad de microprocesador.
- Memoria RAM mínimo 2GB.
- Disco Duro con 10GB de capacidad para instalar el sistema.

***Restricciones de diseño e implementación:***

**RNF5 Lenguaje y marco de trabajo «framework» para el desarrollo del sistema del lado del servidor**

- El sistema deberá ser implementado en el lenguaje de programación PHP versión 5.3 o superior. Como framework de desarrollo se usará Symfony v2.0 el cual propone una arquitectura modular en tres capas: el modelo, la vista y el controlador.

**RNF6 Lenguaje y marco de trabajo «framework» para el desarrollo del sistema del lado del cliente**

- Una de las bibliotecas fundamentales que se deberá utilizar en el desarrollo de la herramienta es ExtJS v3.4 la cual permite el diseño de interfaces visuales interactivas usando metodologías como AJAX y la creación de aplicaciones web con apariencia de escritorio.

**RNF7 Componente para construcción dinámica de reportes**

- Otro componente importante y necesario a utilizar en el desarrollo de la herramienta es el JasperReport v5.1, el cual constituye el núcleo del proceso de generación de gráficas en los reportes.

***Requisitos para la documentación de usuarios en línea y ayuda del sistema:***

**RNF 8 Documentación de usuarios**

- Se entregará a los clientes un manual de usuarios que guía paso a paso las acciones a seguir para trabajar con el componente.

**RNF9 Interfaz:**

- El usuario deberá acceder a la aplicación a través del protocolo HTTP usando el navegador Firefox v32 o una versión superior.

### 2.3 Modelo de casos de uso del sistema

Un caso de uso es una descripción de los pasos o actividades que deberán realizarse para llevar a cabo algún proceso. Los personajes o entidades que participan en un caso de uso se denominan actores. (SlideShare, 2013)

El modelo de casos de uso del sistema contiene actores, casos de uso y sus relaciones. Describe lo que hace el sistema para cada tipo de usuario, representados por uno o varios actores, los cuales representan a individuos o sistemas externos que colaboran con este.

Se identificó el siguiente actor para el componente de graficado:

**Tabla. 1** Actor para el componente de graficado v1.1.

| Actor | Descripción |
|---|---|
| Especialista en diseño de reportes | Persona encargada de diseñar los reportes con GDR v2.0. |

#### *2.3.1 Patrones de caso de uso utilizados*

Un patrón es la pareja de problema / solución con un nombre, que estandariza buenos principios y sugerencias relacionados frecuentemente con la asignación de responsabilidades.

La experiencia en la utilización de casos de uso ha evolucionado en un conjunto de patrones que permiten con más precisión reflejar los requisitos reales, haciendo más fácil el trabajo con los sistemas, y mucho más simple su mantenimiento.

Son comportamientos que deben existir en el sistema, ayudan a describir qué es lo que el sistema debe hacer, es decir, describen el uso del sistema y cómo este interactúa con los usuarios. Estos patrones son utilizados generalmente como plantillas que describen como debería ser estructurados y organizados los casos de uso. Son patrones que capturan mejores prácticas para modelar casos de uso.

A continuación, se describen los patrones de caso de uso utilizados:

**CRUD**

Este patrón se utiliza en los casos donde se quiere realizar altas, bajas, cambios y consultas a alguna entidad del sistema. Su nombre es un acrónimo de las palabras en inglés *Create, Read, Update, Delete*. El **CRUD parcial** se utiliza cuando alguna de las alternativas del caso de uso puede ser modelada como un caso de uso independiente. Ejemplo de la utilización de este patrón es el caso de uso "Gestionar gráfico alto y bajo" que incluye los requisitos: Adicionar gráfico alto y bajo, Modificar gráfico alto y bajo, Mostrar los datos de un gráfico alto y bajo y Mostrar gráfico alto y bajo en vista previa.

**Concordancia por especialización**

Es una relación de un caso de uso hijo a un caso de uso padre que especifica cómo el hijo puede especializar todo el comportamiento y características descritas para el padre. Se utiliza para mostrar que los flujos comparten la estructura, objetivo y comportamiento. Un caso de uso padre puede especializarse en uno o más casos de uso hijos que representan formas más específicas del padre. Ejemplo de la utilización de este patrón es la relación que existe entre el caso de uso Administrar gráficos con los restantes casos de uso por solo mencionar algunos: Administrar gráfico de burbujas, Administrar gráfico de dispersión, Administrar gráfico de pastel 3D y Administrar gráfico alto y bajo.

Los requisitos funcionales fueron agrupados utilizando los patrones de casos de uso Concordancia por Especialización y CRUD parcial en un caso de uso generalizado del cual especializan otros 16. Además, se representa la relación de estos con el actor Especialista en diseño de reportes.

### 2.3.2 Diagrama de casos de uso del sistema

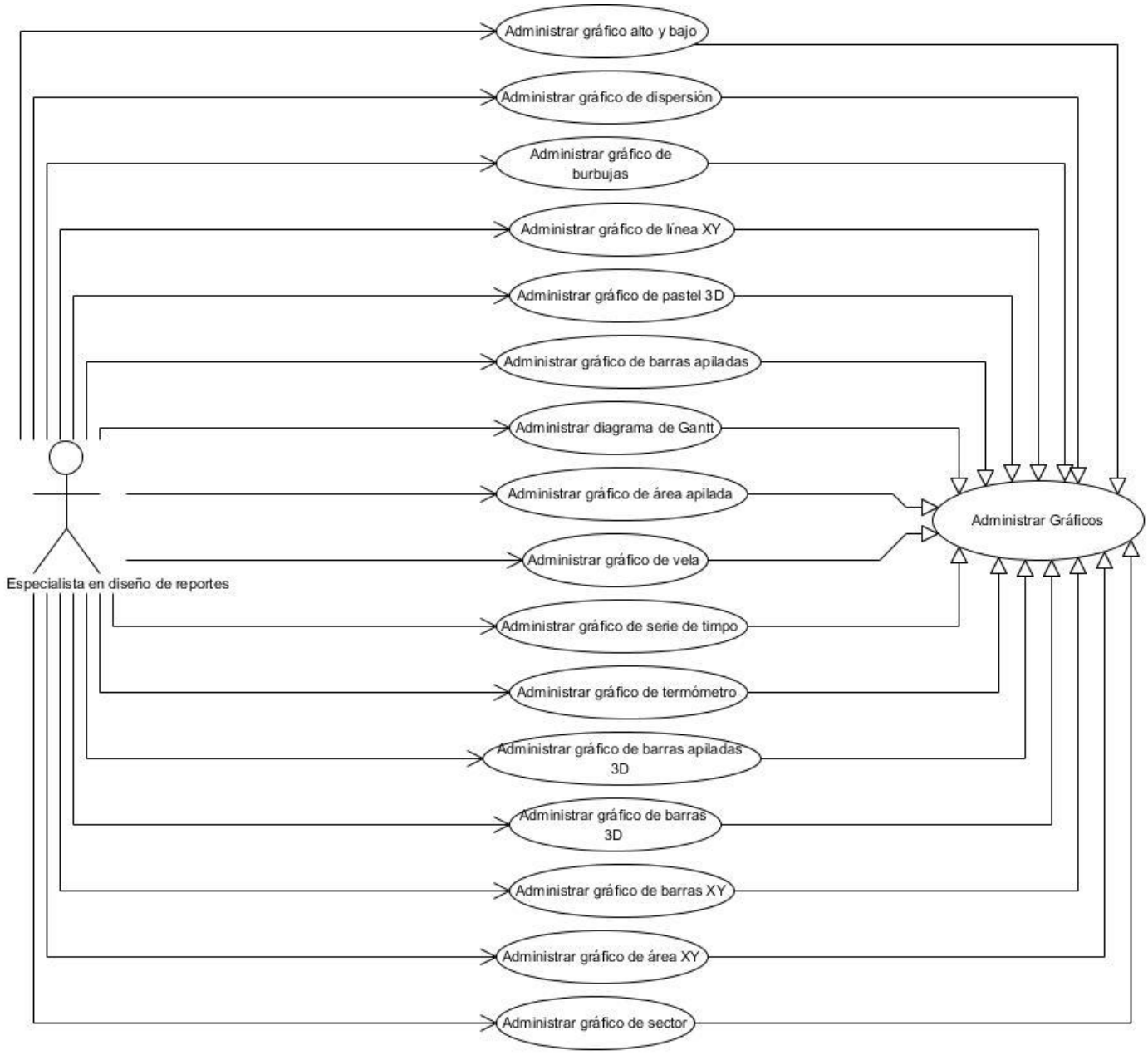



Fig. 19 Diagrama de casos de uso.

### 2.3.3 Descripción textual de los casos de uso del sistema

**Tabla 2.** Descripción textual del caso de uso generalizado Administrar gráficos.

| | |
|---|---|
| **Caso de uso** | Administrar gráficos (Caso de uso base generalizado) |
| **Objetivo** | |

| | |
|---|---|
| | Este caso de uso tiene como objetivo insertar, eliminar, modificar, mostrar datos y mostrar en vista previa una Gráfica en un reporte. |
| **Actores** | Especialista en diseño de reportes |
| **Resumen** | El caso uso se inicia cuando alguno de los demás casos de uso, invoca al comportamiento del caso de uso en cuestión, que generaliza acciones comunes en los mismos. |
| **Complejidad** | Alta. |
| **Prioridad** | Crítico. |
| **Precondiciones** | Existe un reporte creado en el sistema GDR v2.0. |
| **Postcondiciones** | Gráfico insertado en el reporte, gráfico eliminado en el reporte, gráfico modificado en el reporte, datos de gráfico en el reporte, gráfico en vista previa. |

| **Sección "Adicionar gráfico " .Flujo básico** | |
|---|---|
| **Actor** | **Sistema** |
| 1. El diseñador selecciona el gráfico que desea insertar de la paleta de Gráficas y lo arrastra hacia el área de diseño del reporte. | 2. El sistema adiciona el gráfico en el área de diseño y muestra la interfaz de configuración. |
| 3. El diseñador introduce los datos: según el tipo de gráfica.<br>4. El diseñador selecciona la opción aceptar. | 5. El sistema almacena el gráfico en el reporte. |

| **Sección "Adicionar gráfico" .Flujo alterno** | |
|---|---|
| **Actor** | **Sistema** |
| 4.1 El diseñador selecciona la opción cancelar. | 4.1.2 El sistema cierra la interfaz de configuración y elimina el gráfico del área de diseño. |

| **Sección "Mostrar datos de un gráfico" .Flujo básico** | |
|---|---|
| **Actor** | **Sistema** |
| 1.El diseñador selecciona un gráfico existente en el reporte. | 2. El sistema muestra los datos del gráfico en la paleta de "Propiedades" |

| **Sección "Modificar gráfico" .Flujo básico** | |
|---|---|
| **Actor** | **Sistema** |
| 1. El diseñador modifica las propiedades de un gráfico en la paleta de "Propiedades". | |
| 2. El diseñador configura las distintas propiedades en cuanto a tamaño, color, posición. | 3. El sistema almacena las nuevas configuraciones. |
| 3.1. El diseñador cancela la opción de eliminar | 3.1.2. El sistema no elimina la gráfica y el reporte no sufre ningún cambio. |
| **Sección "Mostrar gráfico en vista previa" .Flujo básico** | |

| Actor | Sistema |
|---|---|
| 1.El diseñador selecciona un gráfico existente en el reporte.<br>2.El diseñador selecciona la opción vista previa que se encuentra en el menú vistas en la barra de menús | 3. El sistema muestra una vista previa del gráfico en el reporte. |

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

**Tabla 3.** Descripción textual del caso de uso Administrar gráfico alto y bajo.

| | |
|---|---|
| **Caso de uso** | Administrar gráfico alto y bajo (Especialización) |
| **Objetivo** | Este caso de uso tiene como objetivo insertar, eliminar, modificar, mostrar datos y mostrar en vista previa una Gráfica alto y bajo en un reporte. |
| **Actores** | Especialista en diseño de reportes. |
| **Resumen** | El caso de uso (CU) se inicia cuando el actor va a realizar algunas de las siguientes operaciones:<br>• **Adicionar gráfico alto y bajo:** cuando el actor desea Adicionar un gráfico alto y bajo, introduce los datos necesarios para su construcción y el sistema registra los mismos en el reporte, finalizando así el CU.<br>• **Mostrar datos de un gráfico alto y bajo:** el actor selecciona el gráfico y el sistema muestra las propiedades del mismo, finalizando así el CU.<br>• **Modificar gráfico alto y bajo:** el actor selecciona el gráfico a modificar, el sistema muestra los datos correspondientes, el actor los modifica, finalizando así el caso de uso.<br>• **Visualizar gráfico alto y bajo en vista previa:** el diseñador selecciona la opción de opción vista previa, el sistema muestra la vista previa del gráfico en el reporte, finalizando así el CU. |
| **Complejidad** | Alta. |
| **Prioridad** | Crítico |
| **Precondiciones** | Para Adicionar gráfico alto y bajo en el reporte:<br>• Existe un reporte creado en el sistema GDR v2.0.<br>Para Modificar, Mostrar datos y Mostrar en vista previa un gráfico alto y bajo:<br>• Existe un reporte creado en el sistema GDR v2.0.<br>• Existe un gráfico alto y bajo en el reporte. |
| **Postcondiciones** | Gráfico insertado en el reporte, gráfico eliminado en el reporte, gráfico modificado en el reporte, datos de Gráfico en el reporte, gráfico en vista previa. |
| **Flujo de eventos Administrar grafica alto y bajo.** | |
| **Actor** | **Sistema** |
| 1. El diseñador desea gestionar un gráfico alto y bajo. | |

| | |
|---|---|
| 2. El diseñador puede:<br>**a.** Adicionar gráfico alto y bajo: ir a la sección: **"Adicionar gráfico alto y bajo".**<br>b. Visualizar los datos de un gráfico alto y bajo existente en el reporte: ir a la sección: **"Mostrar datos de un gráfico alto y bajo".**<br>c. Modificar un gráfico alto y bajo existente en el reporte: ir a la sección: **"Modificar gráfico alto y bajo".**<br>d. Visualizar un gráfico alto y bajo existente en el reporte en vista previa: ir a la sección: **"Mostrar gráfico alto y bajo en vista previa".** | |

| **Sección "Adicionar gráfico alto y bajo" .Flujo básico** | |
|---|---|
| **Actor** | **Sistema** |
| Se invoca a la sección **"Adicionar gráfico"** del caso de uso generalizado **"Gestionar gráficos"** con los siguientes datos de entrada: Expresión de la serie, Fecha expresión, Expresión alta, Expresión baja, Abrir expresión, Cerrar expresión y Volumen de la expresión | |

| **Sección "Mostrar datos de un gráfico alto y bajo". Flujo básico.** | |
|---|---|
| **Actor** | **Sistema** |
| Se invoca a la sección **"Mostrar datos de un gráfico"** del caso de uso generalizado **"Administrar gráficos".** | |

| **Sección "Modificar gráfico alto y bajo" .Flujo básico** | |
|---|---|
| **Actor** | **Sistema** |
| Se invoca a la sección **"Modificar gráfico"** del caso de uso generalizado **"Administrar gráficos"** . | |
| 1.El diseñador puede modificar algunos de los siguientes valores: Expresión de la serie, Fecha expresión, Expresión alta, Expresión baja, Abrir expresión, Cerrar expresión y Volumen de la expresión.<br>3. El diseñador modifica los valores que desea modificar.<br>4. El diseñador selecciona la opción aceptar. | 2. El sistema muestra la interfaz de configuración.<br>5. El sistema almacena los cambios en el reporte. |
| **Sección "Modificar gráfico alto y bajo" .Flujo alterno.** | |

| Actor | Sistema |
|---|---|
| 4.2 El diseñador selecciona la opción cancelar. | 4.2.1 El sistema cierra la interfaz de configuración. |
| **Sección "Mostrar gráfico alto y bajo en vista previa". Flujo básico** | |
| **Actor** | **Sistema** |
| Se invoca a la sección **"Mostrar gráfico en vista previa"** del caso de uso generalizado **"Administrar gráficos"** | |

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

## 2.4 Arquitectura de software

La arquitectura de software es un conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un software, permitiendo a los programadores, analistas y todo el conjunto de desarrolladores compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de la aplicación. Es considerada el nivel más alto en el diseño de la arquitectura de un sistema puesto que establecen la estructura, funcionamiento e interacción entre las partes del software.

### Patrón arquitectónico

Los patrones arquitectónicos, o de arquitectura, son patrones de software que ofrecen soluciones a problemas de arquitectura de software en ingeniería de software. Especifican un conjunto predefinido de subsistemas con sus responsabilidades y una serie de recomendaciones para organizar los distintos componentes. (Venete, 2011)

Teniendo en cuenta que la versión anterior del componente de graficado posee una arquitectura basada en el patrón Modelo Vista Controlador, este patrón arquitectónico será el utilizado en la solución.

### Modelo Vista Controlador (MVC)

En líneas generales, MVC es una propuesta de diseño de software utilizada para implementar sistemas donde se requiere el uso de interfaces de usuario. Surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado, donde se potencie la facilidad de mantenimiento, reutilización del código y la separación de conceptos. Su fundamento es la separación del código en tres capas diferentes, acotadas por su responsabilidad, en lo que se llaman Modelos, Vistas y Controladores. (Alvarez, 2014)

### Modelo

Es la capa donde se trabaja con los datos, por tanto, contendrá mecanismos para acceder a la información y también para actualizar su estado. Los datos estarán habitualmente en una base de datos, por lo que en el Modelo se tendrán todas las funciones que accederán a las tablas y harán las correspondientes operaciones (*selects, inserts y updates)* sobre estas.

**Vista**

La vista, como su nombre indica, contienen el código de la aplicación que va a permitir la visualización de las interfaces de usuario, o sea, el código que permitirá renderizar los estados de la aplicación en HTML. En las vistas nada más se tiene los códigos HTML y PHP que permiten mostrar la salida.

**Controlador**

Contiene el código necesario para responder a las acciones que se solicitan en la aplicación, como visualizar un elemento, realizar una compra o una búsqueda de información. En concreto, es una capa que sirve de enlace entre las vistas y los modelos, respondiendo a los mecanismos que puedan requerirse para implementar las necesidades de la aplicación.

## 2.5 Patrones de diseño

Un patrón de diseño es una descripción de clases y objetos comunicándose entre sí, adaptada para resolver un problema de diseño general en un contexto particular. Un patrón de diseño identifica: clases, instancias, roles, colaboraciones y la distribución de responsabilidades.

En el diseño del componente de graficado v1.1 se emplearon distintos patrones los cuales se describen a continuación.

**Patrones GRASP**

**Experto:** Este patrón es el principio básico de asignación de responsabilidades. Indica 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. Este patrón se utiliza en las especializaciones de la clase Chart, por ejemplo, la clase HighLowChart que es quien contiene la información del gráfico alto y bajo.
**Creador:** El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento. Este patrón se utiliza en la clase ReportDesignerArea que se encarga de la construcción de las gráficas conectándose con las clases especializadas por ejemplo HighLowChart.
**Controlador:** Un controlador es un objeto de interfaz no destinado al usuario que se encarga de manejar un evento del sistema y que define además el método de su operación. Este patrón se utiliza en la clase ReportDesignerArea que es el controlador frontal y es quien maneja las peticiones de los usuarios.
**Alta cohesión:** La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. En la solución se encuentra varias clases Chart que contienen

funcionalidades las que poseen un único propósito, no desempeñado por el resto de los elementos, siendo estas funcionalidades, las encargadas administrar las propiedades de los gráficos. Esto hace posible que el software sea flexible a cambios sustanciales con efecto mínimo.

**Bajo Acoplamiento:** El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras. Este patrón se pone de manifiesto en todo el componente de graficado v1.1 y se puede observar en los diagramas de cases donde cada una de estas se comunican con el menor número de clases posible, logrando así un bajo acoplamiento entre las mismas.

**Patrones GoF**

**Factory Method:** Este patrón se define como una interfaz para la creación de cierto tipo de objeto, permitiendo que las subclases decidan qué clase concreta necesitan instanciar. En la solución desarrollada la clase Chart deja la construcción del componente gráfico a las clases especializadas, por ejemplo, la clase BubbleChart.

**Visitor:** Es un patrón de comportamiento que permite definir una operación sobre objetos de una jerarquía de clases sin modificar las clases sobre las que opera. Representa una operación que se realiza sobre los elementos que conforman la estructura de un objeto. Este patrón se evidencia en las clases ComponentNodeToXmlDocVisitor y XmlDocToObjectVisitor donde se visitan cada uno de los atributos y etiquetas de la primera para convertirlos a JSON y los de la segunda para convertirlos a XML.

## 2.6 Modelo de diseño

Un diagrama de clases de diseño muestra la especificación para las clases de una aplicación. Incluye la siguiente información: (GNU, 2008)

- Clases, asociaciones y atributos.
- Interfaces, con sus operaciones y constantes.
- Métodos.
- Navegabilidad.
- Dependencias.

Para el diseño de la solución se utiliza la arquitectura definida por GDR v2.0 que emplea el patrón MVC, ya que este se desarrolla con Symfony 2. Específicamente la solución repercute en la capa Vista y dentro de ella presentando un comportamiento MVC en las capas Vista y Controlador, sin intervención de la capa Modelo ya que no hay interacción con la base de datos. Teniendo en cuenta lo anterior se muestra a continuación el diagrama de clases de diseño del caso de uso Administrar gráfico alto y bajo, donde:

En el paquete **Controlador** se encuentra la clase controlador frontal ReportDesignerArea, donde se registran todos los componentes que se crean en el diseñador de reportes; además se encuentra la clase Preview, encargada de construir la vista previa de los componentes; HighLowChart es la clase específica para la construcción del gráfico alto y bajo y a su vez contiene la clase CfgHighLowChartEditor, que es la encargada de crear la interfaz donde se van a realizar las configuraciones del gráfico.

En el paquete **Vista** se encuentra la clase rd-htm.twing que es la interfaz principal del módulo Diseñador de Reportes donde se encuentra la paleta de componentes de graficado; las clases SP_WindowEditorHighLowChart, CP_WindowEditorHighLowChart, FR_WindowEditorHighLowChart y FR_PropiertiesPanel son las encargadas de la recopilación de la información necesaria para la gestión de las gráficas.

Los marcos de trabajo Symfony y ExtJS así como el Servidor Dinámico de Reportes (SDR) son representados como subsistemas independientes que son utilizados en la implementación del componente de graficado v1.1.

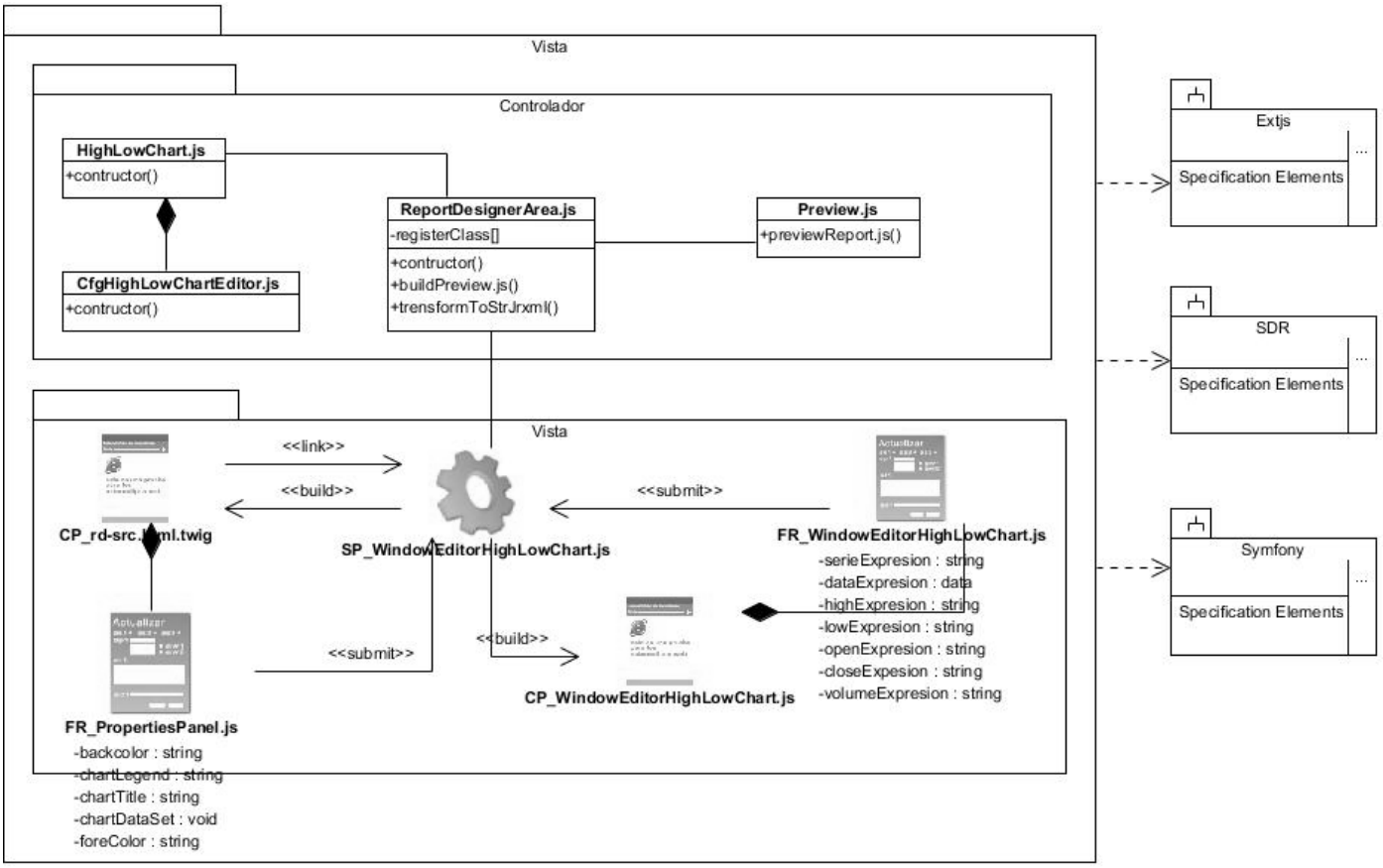



Fig. 20 Diagrama de clases del diseño del caso de uso Administrar gráfico alto y bajo.

Además de los diagramas de clases fueron realizados los diagramas de secuencia de cada uno de los casos de usos identificados. (Ver Anexo 2)

## 2.7 Modelo de despliegue

Un diagrama de despliegue modela la arquitectura en tiempo de ejecución de un sistema. Esto muestra la configuración de los elementos de hardware (nodos) y muestra cómo los elementos y artefactos del software se trazan en esos nodos. (Sparck, 2007)

El diagrama de despliegue realizado es el correspondiente al GDR v2.0, en el cual están presentes cinco nodos principales:

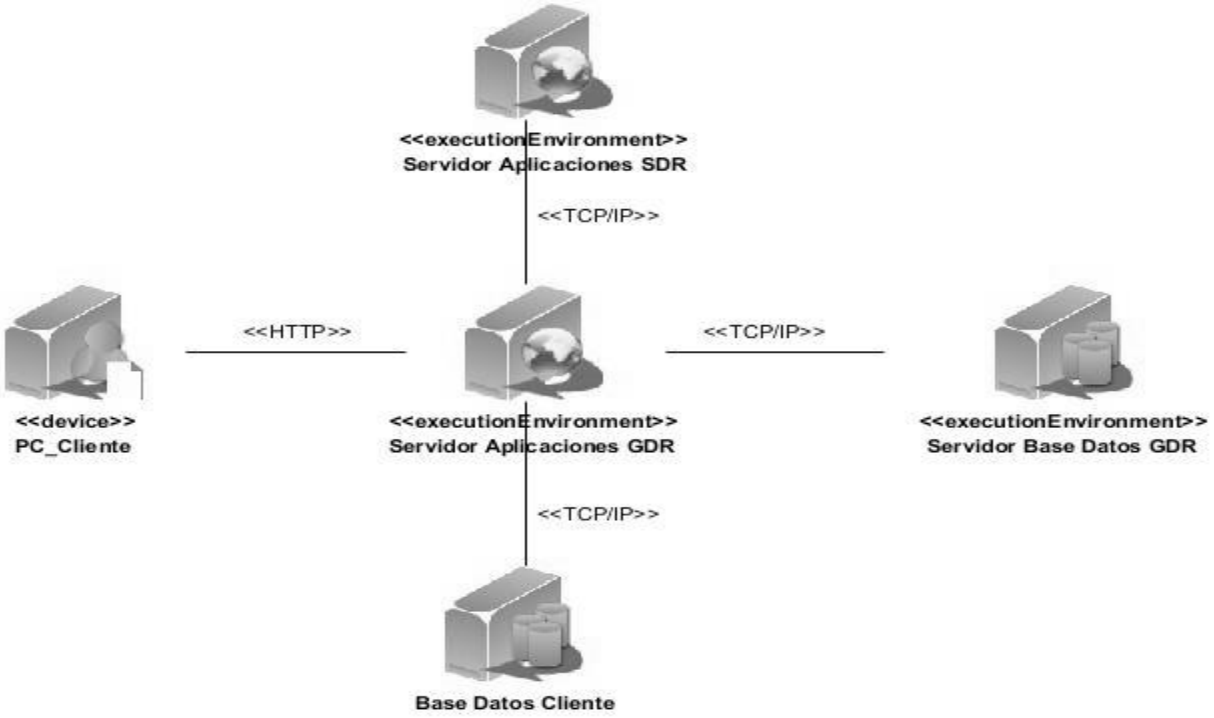



Fig. 21 Diagrama de despliegue.

- El nodo PC_Cliente que debe cumplir con las siguientes características:
  - Navegador web Firefox v32
  - Ordenador Pentium IV o superior, con 1.7 GHz de velocidad de microprocesador.
  - Memoria RAM mínimo 520MB.
- El nodo Servidor de Aplicaciones GDR v2.0 es donde se encuentra el servidor web con GDR v2.0 y debe contar con las siguientes características:
  - SO: GNU/Linux preferentemente Ubuntu GNU/Linux 8.04 o superior, Debian 4 GNU/Linux o superior.
  - Paquetes: apache2, php5, libapache2-mod-php5, php5-cli, php5-mysql, php5-pgsql, php5-sqlite, php5-sybase, php5-xsl, php5-gd y php-apc.
  - Ordenador Dual Core o superior, con 1.7 GHz de velocidad de microprocesador.
  - Memoria RAM mínimo 2GB y disco duro con 10GB de capacidad para instalar el sistema.
- El nodo Base de Datos Cliente es el servidor de bases de datos donde se encuentran los orígenes de datos que serán cargados por GDR v2.0 y que deben estar almacenados en MySQL, PostgreSQL o SQLite.

- El nodo Servidor Base Datos GDR v2.0 contiene la base de datos de GDR v2.0 en PostgreSQL v 9.1.
- El nodo Servidor de Aplicaciones SDR que contiene al SDR v1.0 y debe contar con las siguientes características:
    - SO: GNU/Linux preferentemente Ubuntu 14.4, apache tomcat v7.034 o superior, máquina virtual de java (JDK v7.0)
    - Ordenador con 1GB mínimo de memoria RAM y disco duro con 80GB de capacidad.

El nodo PC_Cliente está conectado con el nodo Servidor de Aplicaciones GDR v2.0 a través del protocolo de transferencia de hipertexto (HTTP) como protocolo de comunicación, este segundo nodo mantiene conexión con otros tres nodos, a través del protocolo de comunicación TCP/IP, el nodo Base de Datos Cliente, el nodo Servidor Base Datos GDR v2.0 y el nodo Servidor de Aplicaciones SDR.

## Conclusiones del capítulo

Al término de este capítulo se concluye lo siguiente:

La realización del modelo de dominio conformado por 11 clases conceptuales y sus relaciones, permitió un mejor entendimiento de los conceptos asociados al dominio de aplicación. Del levantamiento de requisitos realizado, se identificaron 9 requisitos no funcionales y 64 funcionales, estos últimos agrupados en 17 casos de uso utilizando los patrones CRUD parcial y Concordancia por Especialización, lo que permitió identificar las características y capacidades que se deben tener en cuenta para lograr el correcto funcionamiento del componente de graficado v1.1. La definición de la arquitectura mediante el patrón MVC y la utilización de los patrones: Experto, Creador, Controlador, Alta cohesión, Bajo acoplamiento, Factory Method y Visitor permitió estructurar el diseño del componente de graficado v1.1 para GDR v2.0

# Capítulo III: Implementación y prueba de la solución

En el presente capítulo se describe la implementación de los componentes a partir de los requisitos identificados. Se presenta el diagrama de componentes correspondiente a cada uno de los casos de uso a implementar. Además, se muestran ejemplos de las pruebas de software aplicadas, que permite examinar la aplicación desarrollada y garantizar la calidad del software.

## 3.1 Estándar de codificación

Los estándares de codificación permiten entender de manera rápida, fácil y sencilla el código empleado en el desarrollo de un software. Es de gran importancia usar técnicas de codificación y realizar buenas prácticas de programación con vista a generar un código de alta calidad. Si se aplica de forma continua un estándar de codificación bien definido, y posteriormente se efectúan revisiones del código, caben muchas posibilidades de que un proyecto de software se convierta en un sistema fácil de comprender, garantizando un mantenimiento óptimo de dicho código por parte del programador.

El estándar de codificación utilizado en la versión anterior del componente de graficado es el mismo que se utilizó en el desarrollo del GDR v2.0 y a su vez se va a utilizar en la nueva versión del componente de graficado.

A continuación, se muestran los estándares de codificación que se utilizaron para la implementación del componente de graficado v1.1.

- **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, 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;
```

Fig. 22 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 tener nombres coherentes 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 identado con una tabulación con respecto a la misma. La implementación de los métodos debe estar identada 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, 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;
    }
};
```

Fig. 23 Método getChildsByName

- **Operadores:** Los operadores como =, +, +=, -, -=, entre otros, 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 inmediata superior a la línea de código fuente a la cual se desea aclarar.

### 3.2 Modelo de implementación

En el modelo de implementación se describe cómo se implementan los elementos del modelo de diseño. También se describe la ubicación de los componentes de acuerdo a la estructura y modularidad disponible en la implementación y los lenguajes empleados en la misma. Fundamentalmente se muestra la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes.

Los diagramas de componentes ilustran las piezas del software. Tienen un nivel más alto de abstracción que los diagramas de clases. Usualmente un componente se implementa por una o más clases en tiempo de ejecución. Representa la separación de un sistema de software en componentes físicos (archivos, cabeceras, módulos, paquetes) y muestra las dependencias entre estos componentes. Estos diagramas contienen: componentes, interfaces, relaciones de dependencia, generalización, asociación y realización de paquetes o subsistemas. (Hureta Cruz, 2012).

A continuación, se muestra un ejemplo del diagrama de componentes correspondiente al caso de uso del sistema Administrar gráfico alto y bajo.

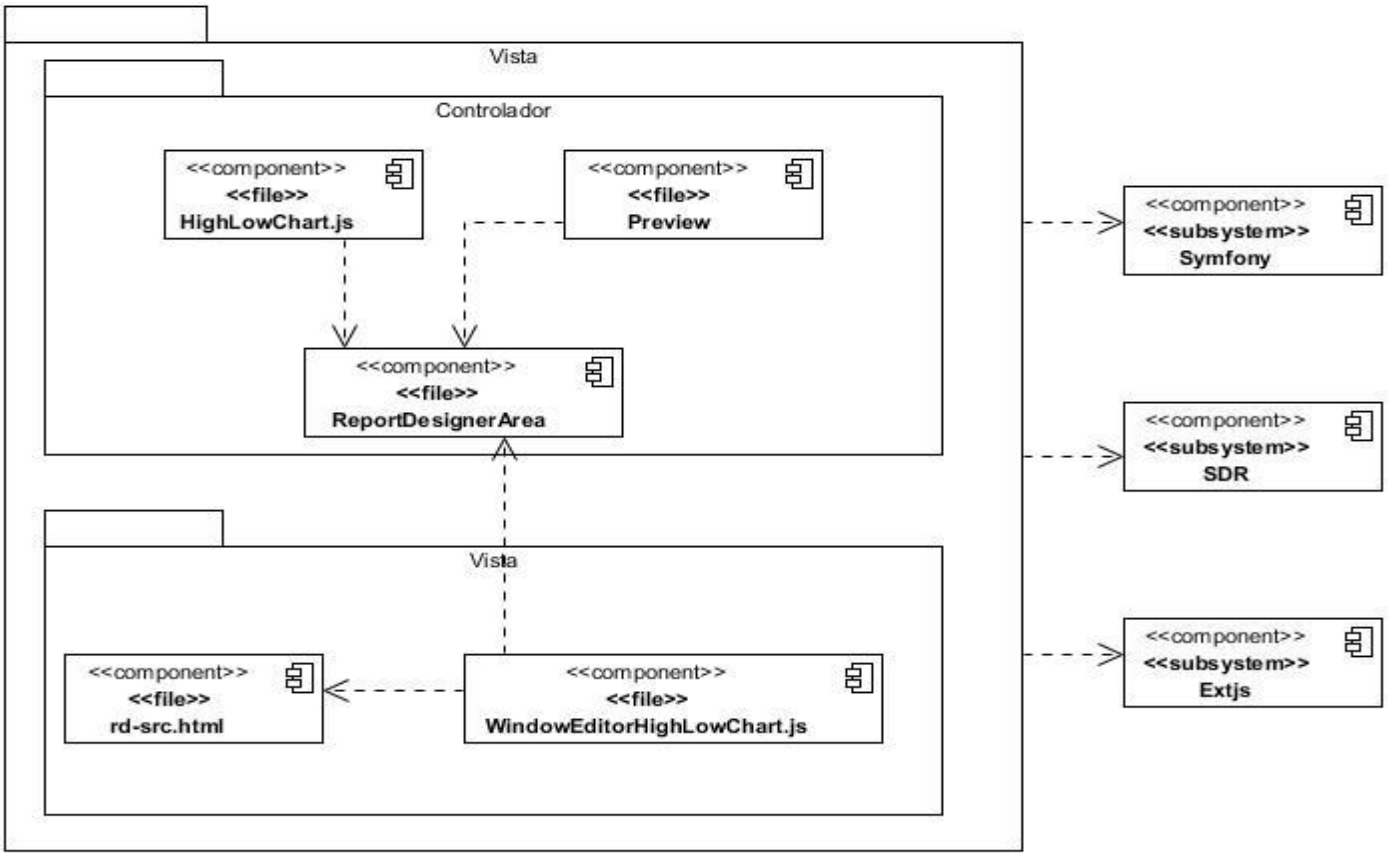



Fig. 24 Diagrama de componentes del caso de uso Administrar gráfico alto y bajo.

El flujo de acciones del diagrama de componentes presentado se define de la siguiente forma: los componentes identificados se agrupan en dos paquetes de implementación básica, la Vista y el Controlador. En la Vista se encuentran los componentes que permiten la interacción directa entre el usuario y el sistema, mostrando y recogiendo información. En el Controlador están los componentes que manipulan los eventos del usuario. Se utilizan componentes definidos en los subsistemas ExtJS y Symfony.

### 3.3 Pantallas principales de la aplicación

El componente de graficado se puede observar en la interfaz principal del módulo Diseñador de Reportes en la parte derecha de la pantalla. En la paleta de componentes se despliega el componente Gráficas y se pueden observar los tipos de gráficas que se pueden insertar en el reporte. Cuando se inserta un tipo de gráfica se arrastra la misma hacia el área del reporte y al instante se muestra la interfaz Editor de propiedades de dicha gráfica donde se introducen los valores requeridos para la creación de la misma, y a su vez se muestra en la parte inferior de la paleta de componentes el Panel de propiedades donde se podrán editar las mismas una vez insertado el gráfico. En la siguiente imagen se muestra un ejemplo de insertar un gráfico alto y bajo.

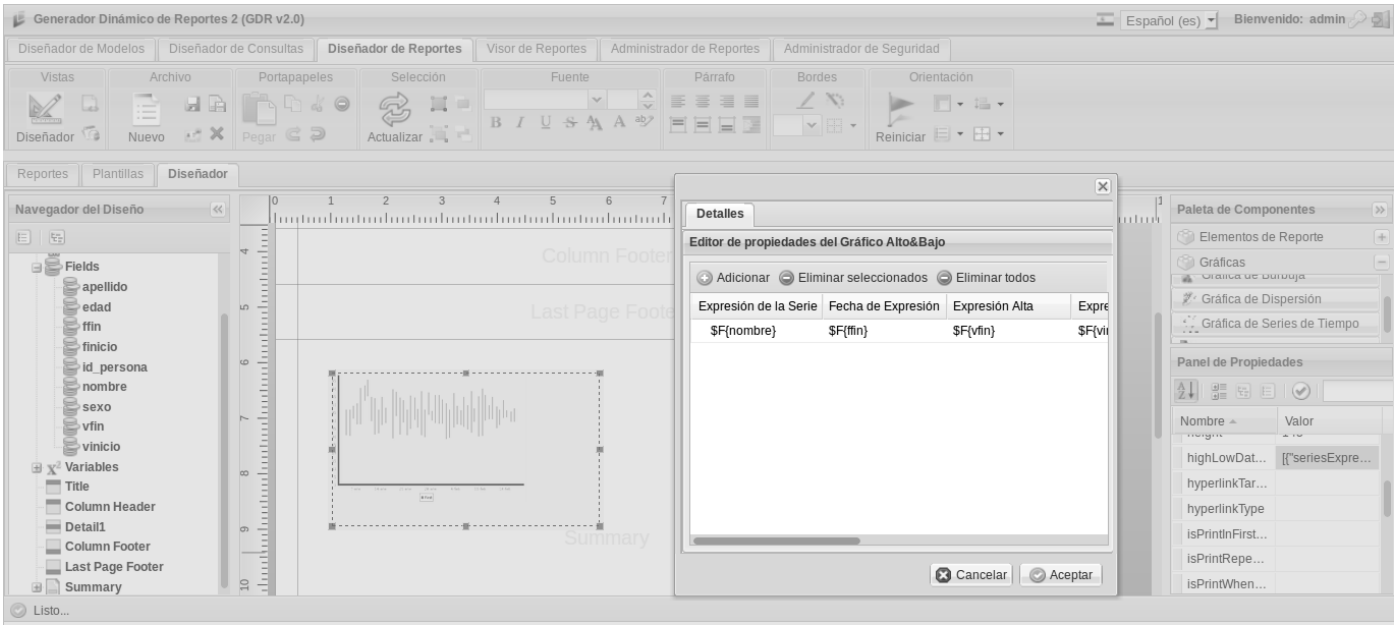



Fig. 25 Interfaz principal del componente de graficado v1.1.

Cuando se inserta un gráfico con éxito se puede proceder a observarlo utilizando la opción de vista previa que se encuentra en la barra de Menú en el menú Vistas. En la siguiente imagen se muestra un ejemplo de vista previa de un gráfico alto y bajo.

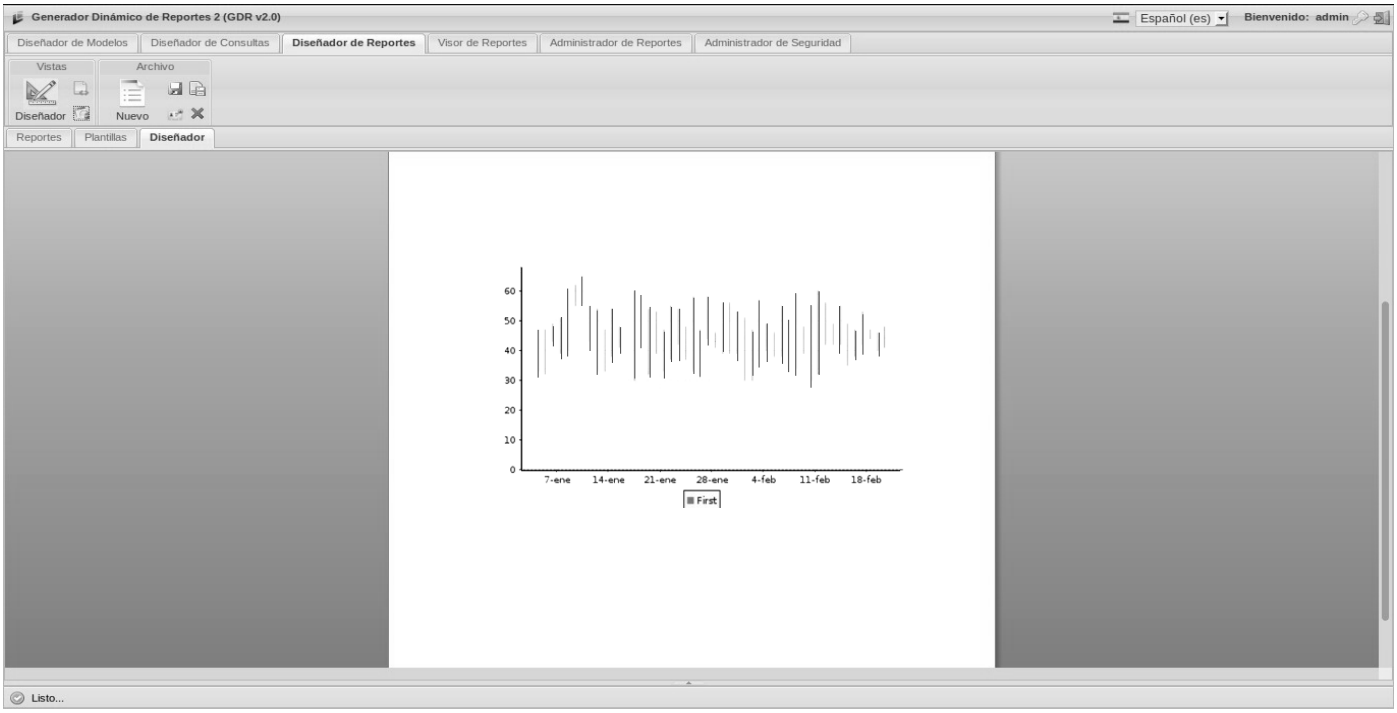



Fig. 26 Interfaz de vista previa.

### 3.4 Modelo de Pruebas de software

Según Roger S. Pressman, las pruebas de software son una función del control de calidad que tienen un objetivo principal: detectar errores.

Una estrategia para probar el software también puede verse en el contexto de la espiral. La prueba de unidad comienza en el vértice de la espiral y se concentra en cada unidad (por ejemplo, componente, clase o un objeto de contenido de una webapp) del software como se implementó en el código fuente. La prueba avanza al moverse hacia afuera a lo largo de la espiral, hacia la prueba de integración, donde el enfoque se centra en el diseño y la construcción de la arquitectura del software. Al dar otra vuelta hacia afuera de la espiral se encuentra la prueba de validación, donde los requerimientos establecidos como parte de su modelado se validan confrontándose con el software que se construyó. Finalmente, se llega a la prueba del sistema, donde el software y otros elementos del sistema se prueban como un todo. (Pressman, 2011)

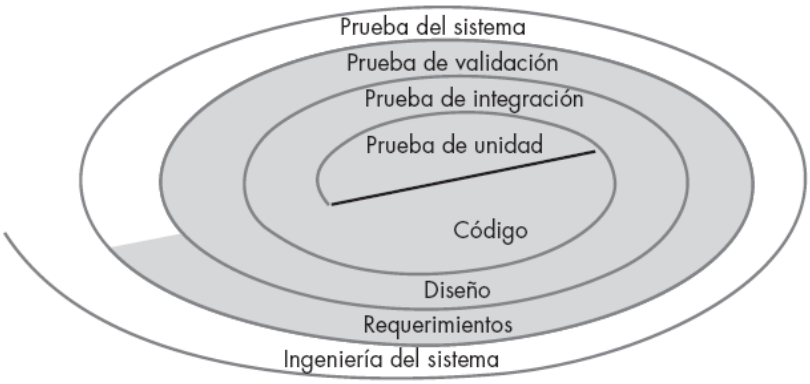



Fig. 27 Estrategia de pruebas de software.

**Prueba de unidad:** Inicialmente se realizarán las pruebas de unidad las cuales se enfocan en cada componente de manera individual. (Pressman, 2011)

Estas pruebas se fueron aplicando a medida que se fue desarrollando, con ayuda del IDE NetBeans v8.1 se fueron detectando los errores de programación en los distintos métodos y clases y se fueron corrigiendo en el momento.

**Prueba de integración:** Las pruebas de integración es la fase de la prueba de software en la cual módulos individuales de software son combinados y probados como un grupo. (Pressman, 2011)

La versión 1.1 del componente se desarrolló sobre el código de GDR v2.0 por lo que las pruebas de integración se realizaron en conjunto con las de validación puesto que el componente ya está integrado. A continuación, los pasos para integrar cada componente gráfico al sistema:

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 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. A la clase ComponentNodeToXmlDocVisitor se le implementan las funciones necesarias para convertir los objetos JSON asociados a los componentes gráficos creados en el jrxml correspondiente y a la clase XmlDocToObjectVisitor se le incluyen 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.

**Prueba de validación:** Después de integrar (construir) el software, se realiza una serie de pruebas de orden superior. Deben evaluarse criterios de validación establecidos durante el análisis de requerimientos. La prueba de validación proporciona la garantía final de que el software cumple con todos los requerimientos funcionales. (Pressman, 2011)

Para las pruebas de validación se aplicaron pruebas de caja negra utilizando el método partición de equivalencia.

**Las pruebas de caja negra**, también llamadas pruebas de comportamiento, se enfocan en los requerimientos funcionales del software, es decir, las técnicas de prueba de caja negra le permiten derivar conjuntos de condiciones de entrada que revisarán por completo todos los requerimientos funcionales para un programa. (Pressman, 2011)

**La partición de equivalencia** es un método de prueba de caja negra que divide el dominio de entrada de un programa en clases de datos de los que pueden derivarse casos de prueba. Un caso de prueba ideal descubre de primera mano una clase de errores. (Pressman, 2011)

Teniendo en cuenta que se realizaron pruebas de caja negra se diseñó un caso de prueba por cada caso de uso. El siguiente es un ejemplo donde se detalla el requisito funcional Adicionar gráfico alto y bajo perteneciente al caso de uso Administrar gráfico alto y bajo. A continuación, se detallan las variables que se encuentran en las interfaces asociadas al caso de uso Administrar gráfico alto y bajo.

**Tabla. 4** Descripción de las variables del diseño de casos de prueba del caso de uso Administrar gráfico alto y bajo.

| No | Nombre de campo | Clasificación | Valor Nulo | Descripción |
|---|---|---|---|---|
| 1 | Expresión de la serie | Campo de texto | No | Campo que debe tener la siguiente estructura: $F{} y dentro de las llaves el nombre del campo asociado al reporte que se quiere graficar. El campo debe ser de tipo String |
| 2 | Fecha de expresión | Campo de texto | No | Campo que debe tener la siguiente estructura: $F{} y dentro de las llaves el nombre del campo asociado al reporte que se quiere graficar. El campo debe ser de tipo fecha |
| 3 | Expresión alta | Campo de texto | No | Campo que debe tener la siguiente estructura: $F{} y dentro de las llaves el nombre del campo asociado al reporte que se quiere graficar. El campo debe ser numérico |

| | | | | |
|---|---|---|---|---|
| 4 | Expresión baja | Campo de texto | No | Campo que debe tener la siguiente estructura: $F{} y dentro de las llaves el nombre del campo asociado al reporte que se quiere graficar. El campo debe ser String |
| 5 | Abrir expresión | Campo de texto | No | Campo que debe tener la siguiente estructura: $F{} y dentro de las llaves el nombre del campo asociado al reporte que se quiere graficar. El campo debe ser numérico |
| 6 | Cerrar expresión | Campo de texto | No | Campo que debe tener la siguiente estructura: $F{} y dentro de las llaves el nombre del campo asociado al reporte que se quiere graficar. El campo debe ser numérico |
| 7 | Volumen de la expresión | Campo de texto | No | Campo que debe tener la siguiente estructura: $F{} y dentro de las llaves el nombre del campo asociado al reporte que se quiere graficar. El campo debe ser String |

**Caso de prueba Insertar gráfico alto y bajo del caso de uso Administrar gráfico alto y bajo.**

**Tabla. 5** Caso de prueba Insertar gráfico alto y bajo del caso de uso Administrar gráfico alto y bajo.

| **Escenario** | **Descripción** | **V1** | **V2** | **V3** | **V4** | **V5** | **V6** | **V7** | **Respuesta del sistema** | **Flujo central** |
|---|---|---|---|---|---|---|---|---|---|---|
| EC 1.1 Añadir gráfico alto y bajo con los datos correctos | Se añade un gráfico alto y bajo con los datos correctamente | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | Registra el gráfico alto y bajo correctamente | 1-Se despliega el componente gráfico de la paleta de componentes.<br>2-Se selecciona la gráfica alto y bajo y se arrastra hasta el área del reporte donde se quiere insertar<br>3-Se insertan los datos en el formulario ¨Editor del Gráfico alto y bajo¨ que aparece.<br>4-Se selecciona la opción aceptar.<br>5-Se añade la gráfica correctamente |
| EC 1.2 Añadir gráfico alto y bajo con datos nulos | Se añade un gráfico alto y bajo con datos nulos | I (-) | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | No permite seleccionar la opción aceptar puesto que es | 1-Se despliega el componente gráfico de la paleta de componentes.<br>2-Se selecciona la gráfica |
| | | V $F{} | I (-) | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | | |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | V $F{} | V $F{} | I (-) | V $F{} | V $F{} | V $F{} | V $F{} | obligatorio llenar todos los campos | alto y bajo y se arrastra hasta el área del reporte donde se quiere insertar<br>3-Se insertan algunos datos en el formulario ¨Editor del Gráfico alto y bajo¨ que aparece dejando campos en blanco |
| | | V $F{} | V $F{} | V $F{} | I (-) | V $F{} | V $F{} | V $F{} | | |
| | | V $F{} | V $F{} | V $F{} | V $F{} | I (-) | V $F{} | V $F{} | | |
| | | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | I (-) | V $F{} | | |
| | | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | I (-) | | |
| EC 1.3 Añadir gráfico alto y bajo con los datos incorrectos | Se añade un gráfico alto y bajo con los datos incorrectamente | I | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | Adiciona la gráfica pero no se muestra en la vista previa | 1-Se despliega el componente gráfico de la paleta de componentes.<br>2-Se selecciona la gráfica alto y bajo y se arrastra hasta el área del reporte donde se quiere insertar<br>3-Se insertan datos incorrectos en el formulario ¨Editor del Gráfico alto y bajo¨ que aparece.<br>4-Se selecciona la opción aceptar<br>5-Se añade la gráfica incorrectamente |
| | | V $F{} | I | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | | |
| | | V $F{} | V $F{} | I | V $F{} | V $F{} | V $F{} | V $F{} | | |
| | | V $F{} | V $F{} | V $F{} | I | V $F{} | V $F{} | V $F{} | | |
| | | V $F{} | V $F{} | V $F{} | V $F{} | I | V $F{} | V $F{} | | |
| | | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | I | V $F{} | | |
| | | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | V $F{} | I | | |
| EC 1.4 Cancelar la operación de añadir gráfico alto y bajo | Se cancela la operación de añadir gráfico alto y bajo | N/A | N/A | N/A | N/A | N/A | N/A | N/A | El sistema cancela la adición del gráfico y lo elimina del reporte | 1-Se despliega el componente gráfico de la paleta de componentes.<br>2-Se selecciona la gráfica alto y bajo y se arrastra hasta el área del reporte donde se quiere insertar<br>3-Se insertan los datos en el formulario Editor del Gráfico alto y bajo¨ que aparece.<br>4-Se selecciona la opción cancelar.<br>5-Se cierra el formulario.<br>6-No se inserta la gráfica |

**Prueba de Sistema:** El software, una vez validado, debe combinarse con otros elementos del sistema (por ejemplo, hardware, personal, bases de datos). La prueba del sistema verifica que todos los elementos se mezclan de manera adecuada y que se logra el funcionamiento y/o rendimiento global del sistema. (Pressman, 2011)

Las pruebas de sistema se realizaron en conjunto con las pruebas de aceptación que se realizaron con los especialistas del proyecto GDR v2.0 que a la vez son los clientes del componente de graficado v1.1.

**Prueba de Aceptación:** El test de aceptación es la última de las pruebas que debe atravesar una aplicación. Una vez que ya se ha probado que cada módulo funciona bien por separado, que la aplicación puede utilizarse bajo condiciones de operación extremas, que todos los módulos se integran correctamente y que el software ofrezca las funciones esperadas, llega el momento de escuchar la opinión del impulsor del proyecto: el usuario final.

La aplicación de estas pruebas fue realizada en laboratorio de producción del proyecto GDR v2.0 llevadas a cabo por tres de sus especialistas y con la presencia del equipo de desarrollo donde se procedió a la revisión de la solución y del manual de usuario elaborado. El cliente después de haber realizado las pruebas satisfactoriamente hizo entrega de la carta de aceptación (Ver anexo 1).

**Resultados de las pruebas de software aplicadas**

Los resultados de las pruebas que no fueron satisfactorias pasaron a ser no conformidades y se emitieron en el registro de defectos y dificultades detectados. A continuación, se muestran las no conformidades detectadas durante las pruebas a la aplicación.

**Tabla. 6** No conformidades.

| Elemento | No. | No conformidad | Etapa de detección | Significativa | No significativa | Estado NC | Respuesta del equipo de desarrollo |
|---|---|---|---|---|---|---|---|
| Aplicación | 1 | Cuando se inserta un gráfico de burbujas con los datos correctos no se construye en la vista previa. | Pruebas de funcionalidad | X | | PD 10/5/16 RA 11/5/16 | Se corrigió un error de implementación en la clase ComponentNodeToXmlDocVisitor.js. |
| Aplicación | 2 | Cuando se inserta un gráfico de termómetro con los datos correctos no se construye en la vista previa. | Pruebas de funcionalidad | X | | PD 10/5/16 RA 11/5/16 | Se corrigió un error de implementación en la clase ComponentNodeToXmlDocVisitor.js. |
| Aplicación | 3 | Cuando se inserta un gráfico de sector con los datos correctos no se construye en la vista previa. | Pruebas de funcionalidad | X | | PD 10/5/16 RA 12/5/16 | Se corrigió un error de implementación en la clase ComponentNodeToXmlDocVisitor.js. |
| Aplicación | 4 | Cuando se quiere | Pruebas de | X | | PD | Se corrigió un error de |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | insertar un gráfico de barras 3D, se arrastra el componente hacia el área del reporte y no sale la interfaz de configuración de los atributos. | funcionalidad | | | 10/5/16<br>RA<br>12/5/16 | implementación en la clase CfgBar3DChartEditor.js. |
| Aplicación | 5 | Cuando se quiere insertar un gráfico de dispersión, se arrastra el componente hacia el área del reporte y no sale la interfaz de configuración de los atributos. | Pruebas de funcionalidad | X | | PD<br>13/5/16<br>RA<br>15/5/16 | Se corrigió un error de implementación en la clase CfgScatterChartEditor.js. |
| Aplicación | 6 | Para adicionar la gráfica de barras 3D se muestra un dato de entrada innecesario para la graficación. “Expresión de etiqueta” | Pruebas de funcionalidad | | X | PD<br>10/5/16<br>RA<br>12/5/16 | Se eliminó el campo del formulario en la clase WindowEditorBar3DChart.js “Expresión de etiqueta” |
| Aplicación | 7 | Para adicionar la gráfica de barras apiladas 3D se muestra un dato de entrada innecesario para el graficado. “Expresión de etiqueta” | Pruebas de funcionalidad | | X | PD<br>10/5/16<br>RA<br>12/5/16 | Se eliminó el campo del formulario en la clase WindowEditorStaketBar3DChart.js “Expresión de etiqueta |
| Aplicación | 8 | Para adicionar la gráfica de barras apiladas se muestra un dato de entrada innecesario para la graficación. “Expresión de etiqueta” | Pruebas de funcionalidad | | X | PD<br>13/5/16<br>RA<br>15/5/16 | Se eliminó el campo del formulario en la clase WindowEditorStaketBarChart.js “Expresión de etiqueta |

Se evidencian a continuación, los resultados de las no conformidades de las pruebas efectuadas en las 3 iteraciones realizadas, donde en la primera iteración se identificaron un total de 5 no conformidades, 4 significativas y 2 no significativas, en la segunda iteración se identificaron 1 significativa y 1 no significativa y en la tercera iteración no se identificaron no conformidades.

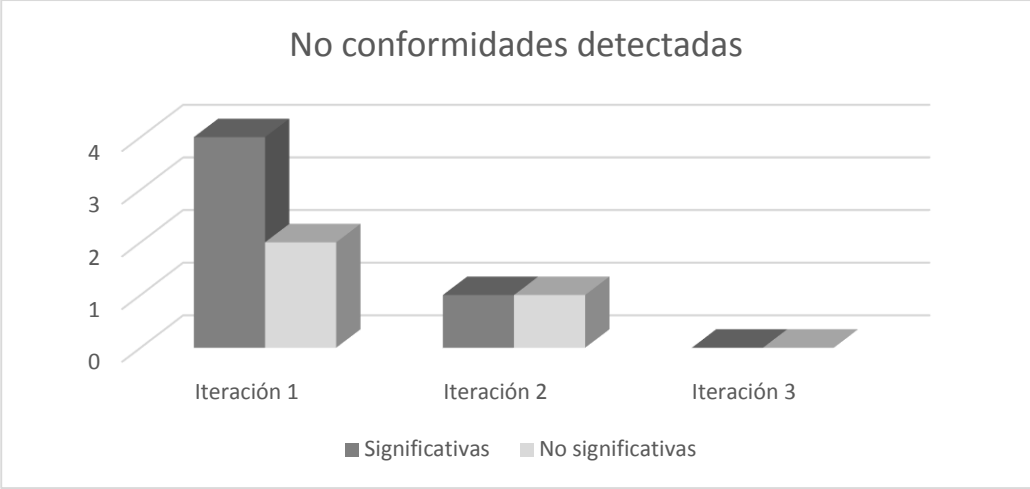



Fig. 28 Gráfica de no conformidades.

## Conclusiones del capítulo

Al término de este capítulo se concluye lo siguiente:

La definición del estándar de codificación permitió una estructuración más clara del código fuente de la solución. La realización de los diagramas de componentes permitió estructurar el proceso de implementación del componente de graficado v1.1. La realización de las pruebas de software: Unidad, Integración, Validación (Pruebas de caja negra con el método Partición de equivalencia) y Sistema (Pruebas de Aceptación) permitió comprobar que la aplicación cumple satisfactoriamente con los requisitos funcionales y no funcionales identificados.

## Conclusiones generales

Una vez finalizada la investigación se concluye que:

- El estudio realizado sobre las características del negocio permitió la selección de las tecnologías y las herramientas adecuadas para desarrollar el componente de graficado v1.1 para GDR v2.0, así como la selección de OpenUP como metodología para guiar todo el proceso de desarrollo del mismo.
- La elaboración de los artefactos de acuerdo a la metodología propuesta, además utilizando los patrones arquitectónicos y de diseño, permitió establecer las pautas para el desarrollo del componente de graficado v1.1 para GDR v2.0.
- La implementación de los elementos definidos en los diagramas de componentes permitió el cumplimiento de los 64 requisitos funcionales y 9 no funcionales identificados, y la obtención de la versión 1.1 del componente de graficado de GDR v2.0.
- La ejecución de las pruebas de software permitió verificar el correcto funcionamiento del componente de graficado v1.1 para GDR v2.0 y su ajuste a las necesidades del cliente.

## Recomendaciones

Valorar la inclusión, en próximas versiones del componente, de la gráfica de Araña soportada por el motor de reporte JasperReport v5.6.

## Referencias bibliográficas


1. **Alvarez, Miguel Angel. 2014.** DesarrolloWeb. *DesarrolloWeb.* [En línea] 2 de enero de 2014. [Citado el: 12 de enero de 2016.] http://www.desarrolloweb.com/articulos/que-es-mvc.html.
2. **Americanbulls. 2016.** candlesticker. *candlesticker.* [En línea] 2016. [Citado el: 16 de 4 de 2016.] http://www.candlesticker.com/AboutUs.aspx?lang=es.
3. **Atlantic International University. 2016.** Open Courses. *Open Courses.* [En línea] 2016. [Citado el: 13 de 6 de 2016.] http://cursos.aiu.edu/Fundamentos%20de%20Estad%C3%ADstica/pdf/Tema%205.pdf.
4. **Copyright . 2016.** Copyright . *Copyright .* [En línea] 2016. [Citado el: 13 de 6 de 2016.] http://www.knorrigt.com/como-crear-un-grafico-de-termometro/.
5. **Definición. 2015.** Definición . *Definición .* [En línea] 2015. [Citado el: 25 de 11 de 2015.] http://definicion.mx/reporte/.
6. **Definición.de. 2008-2015.** Definicion de Reporte. *Definición .de.* [En línea] 2008-2015. [Citado el: 3 de 10 de 2015.] http://definicion.de/reporte/.
7. **DefiniciónABC. 2013.** DefiniciónABC. [En línea] 2013. http://www.definicionabc.com/general/grafico.php.
8. **Figueroa Zarraga, Ysis A. 2014.** Graficos estadisticos y matrices. *Graficos estadisticos y matrices.* [En línea] 16 de 6 de 2014. [Citado el: 14 de 6 de 2016.] http://www.monografias.com/trabajos101/graficos-estadisticos-y-matrices/graficos-estadisticos-y-matrices.shtml.
9. **Figueroa, Viridiana. 2015.** Scribd. *Scribd.* [En línea] 2015. [Citado el: 1 de 11 de 2015.] https://es.scribd.com/doc/50288837/Caracteristicas-de-PHP#scribd.
10. **GNU. 2008.** GNU. *GNU.* [En línea] 3 de noviembre de 2008. [Citado el: 20 de enero de 2016.] http://www.gnu.org/copyleft/fdl.html.
11. **Gómez, Edgar. 2014.** Que es un framework. *Que es un framework.* [En línea] 14 de 5 de 2014. [Citado el: 5 de 12 de 2015 .] http://edgargomez.es/que-es-un-framework/.
12. **Heffelfinger, David R. 2006.** Jasperreports Reporting for Java Developers. [aut. libro] David R Heffelfinger. *Jasperreports Reporting for Java Developers.* 2006.
13. **Hernández, Iliana Amabely Silva. 2003.** Generador Automático de Reportes Dinámicos. *Departamento de Computación.* [En línea] 2003. http://www.cs.cinvestav.mx/tesisgraduados/2003/resumenIlianaAma.html.
14. **Herrera, Cristhan. 2013.** AdictosAlTrabajo. *AdictosAlTrabajo.* [En línea] 2013. [Citado el: 5 de 12 de 2015.] https://www.adictosaltrabajo.com/tutoriales/ireport/.
15. **Hureta Cruz, Daniel. 2012.** SlideShare. *SlideShare.* [En línea] 19 de diciembre de 2012. [Citado el: 27 de marzo de 2016.] http://es.slideshare.net/dhuertacruz/diagrama-de-componentes-15705604.
16. **Instituto de Tecnologías Educativas. 2016.** ite Instituto de Tecnologías Educativas. *ite Instituto de Tecnologías Educativas.* [En línea] 2016. [Citado el: 13 de 6 de 2016.] http://www.ite.educacion.es/formacion/materiales/180/cd/m4_10/grficos_de_lneas.html.
17. **Jason. 2009.** Support. *Support.* [En línea] 2 de marzo de 2009. [Citado el: 28 de marzo de 2016.] http://support.scribd.com/forums/33939/entries/25580.

18. **Krall, Cesar. 2016.** aprenderaprogramar.com. *aprenderaprogramar.com.* [En línea] 2016. [Citado el: 2 de 5 de 2016.] http://aprenderaprogramar.com/index.php?option=com_content&view=article&id=688:ique-es-y-para-que-sirve-uml-versiones-de-uml-lenguaje-unificado-de-modelado-tipos-de-diagramas-uml&catid=46:lenguajes-y-entornos&Itemid=163.
19. **Lamarca Lapuente, María Jesús. 2013.** Hipertexto, el nuevo concepto de documento en la cultura de la imagen. *Hipertexto, el nuevo concepto de documento en la cultura de la imagen.* [En línea] 8 de 12 de 2013. [Citado el: 20 de 2 de 2016.] http://www.hipertexto.info/documentos/xml.
20. **Larman, Craig . 2003.** UPM. *UPM.* [En línea] 5 de abril de 2003. [Citado el: 30 de marzo de 2016.] http://is.ls.fi.upm.es/docencia/is2/documentacion/ModeloDominio.pdf.
21. **LibrosWeb.es. 2015.** LibrosWeb.es. *LibrosWeb.es.* [En línea] 25 de 5 de 2015. [Citado el: 20 de 1 de 2016.] http://librosweb.es/libro/symfony_1_4/capitulo_1/symfony_en_pocas_palabras.html.
22. **LinkedIn, Corporation. 2014.** LinkedIn Corporation. [En línea] 26 de marzo de 2014. [Citado el: 20 de abril de 2016.] https://www.linkedin.com/legal/copyright-policy.
23. **Martínez, Andrés G. 2015.** SPSS FREE. *SPSS FREE.* [En línea] 2015. [Citado el: 13 de 6 de 2016.] http://www.spssfree.com/curso-de-spss/graficos-en-spss/grafico-de-sectores-en-spss.html.
24. **Microsoft. 2016.** Presentar datos en un gráfico de áreas. *Presentar datos en un gráfico de áreas.* [En línea] 2016. [Citado el: 14 de 6 de 2016.] https://support.office.com/es-es/article/Presentar-datos-en-un-gr%C3%A1fico-de-%C3%A1reas-f4842b1c-a29b-4766-be07-3b61d2e77d39.
25. **—. 2016.** Presentar los datos en un gráfico circular. *Presentar los datos en un gráfico circular.* [En línea] 2016. [Citado el: 14 de 6 de 2016.] https://support.office.com/es-es/article/Presentar-los-datos-en-un-gr%C3%A1fico-circular-1a5f08ae-ba40-46f2-9ed0-ff84873b7863.
26. **—. 2016.** Presentar los datos en un gráfico de burbuja. *Presentar los datos en un gráfico de burbuja.* [En línea] 2016. [Citado el: 13 de 6 de 2016.] https://support.office.com/es-es/article/Presentar-los-datos-en-un-gr%C3%A1fico-de-burbuja-424d7bda-93e8-4983-9b51-c766f3e330d9.
27. **—. 2016.** Presentar los datos en un gráfico de columnas. *Presentar los datos en un gráfico de columnas.* [En línea] 2016. [Citado el: 14 de 6 de 2016.] https://support.office.com/es-es/article/Presentar-los-datos-en-un-gr%C3%A1fico-de-columnas-d89050ba-e6b6-47de-b090-e9ab353c4c00.
28. **Montes Oliver, Keimer, Hernández Cervantes, Beatriz y Mendoza Garnache, Alberto. 2015.** Laccei. *Laccei.* [En línea] 21 de julio de 2015. http://www.laccei.org/LACCEI2015-SantoDomingo/RefereedPapers/RP034.pdf.
29. **Nobelo Can, Carolina. 2010.** scribd. *scribd.* [En línea] 6 de septiembre de 2010. [Citado el: 19 de febrero de 2016.] https://es.scribd.com/doc/37187866/Requerimientos-funcionales-y-no-funcionales#scribd.
30. **Noreña Ossa, Andrés. 2011.** Vitrina EIA Repositorio Institucional. *Vitrina EIA Repositorio Institucional.* [En línea] 2011. [Citado el: 13 de 6 de 2016.] http://repository.eia.edu.co/bitstream/11190/1097/1/ADMO0662.pdf.
31. **OpenOffice. 2015.** Tipos de gráficos en OpenOffice Calc. *Tipos de gráficos en OpenOffice Calc.* [En línea] 26 de 8 de 2015. [Citado el: 14 de 6 de 2016.] http://wiki.openoffice.es/Tipos_de_graficos_en_OpenOffice_Calc.

32. **Oracle. 2016.** NetBeans. *NetBeans.* [En línea] 29 de enero de 2016. [Citado el: 28 de marzo de 2016.] https://netbeans.org/community/releases/81/.
33. **Pressman, Roger S. 2011.** *Ingenieria De Software Un enfoque práctico.* Madrid : McWragHill, 2011.
34. **Quees. 2015.** Quees.la. *Quees.la.* [En línea] 2015. [Citado el: 25 de 11 de 2015.] http://quees.la/reporte/.
35. **SlideShare. 2013.** SlideShare. *SlideShare.* [En línea] 27 de febrero de 2013. [Citado el: 5 de diciembre de 2015.]
36. **Sparck. 2007.** SparckSystem. *SparckSystem.* [En línea] 2 de febrero de 2007. [Citado el: 25 de enero de 2016.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.
37. **Universidad de Sonora. 2015.** UNISON. *UNISON.* [En línea] 2015. [Citado el: 14 de 6 de 2016.] http://www.estadistica.mat.uson.mx/Material/seriesdetiempo.pdf.
38. **Vazquez Ocampo, Gerardo. 2013.** SlideShare. *SlideShare.* [En línea] 8 de 3 de 2013. [Citado el: 25 de 10 de 2015.] http://es.slideshare.net/AUREA23/reporte-escrito.
39. **Venete, Adriana. 2011.** Ptrones Arquitectura. *Ptrones Arquitectura.* [En línea] 10 de enero de 2011. [Citado el: 15 de febrero de 2016.] https://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=5&ved=0ahUKEwim_4Dbr5HLAhUCbj4KHUw4CZUQFggzMAQ&url=http%3A%2F%2Fmahara.uji.es%2Fartefact%2Ffile%2Fdownload.php%3Ffile%3D54534%26view%3D4648&usg=AFQjCNGV0SXl7IX8wf6-q0sGmoyBVuPYoQ&bvm=bv.1152.
40. **Vera, Claudia. 2015.** Universidad Santo Tomas. *Universidad Santo Tomas.* [En línea] 2015. [Citado el: 23 de 11 de 2015.] http://soda.ustadistancia.edu.co/enlinea/claudiaveraInformatica%20I_Momento%20III/elementos_de_un_grfico.html.
41. **Yasen, Noemé . 2014.** SPSS FREE. *SPSS FREE.* [En línea] 2014. [Citado el: 14 de 6 de 2016.] http://face.unt.edu.ar/web/iadmin/wp-content/uploads/sites/2/2014/12/Aplicaci%C3%B3n-pr%C3%A1ctica-Diagrama-de-Gantt-para-Jornada-IA-Handl.pdf.

## Bibliografía

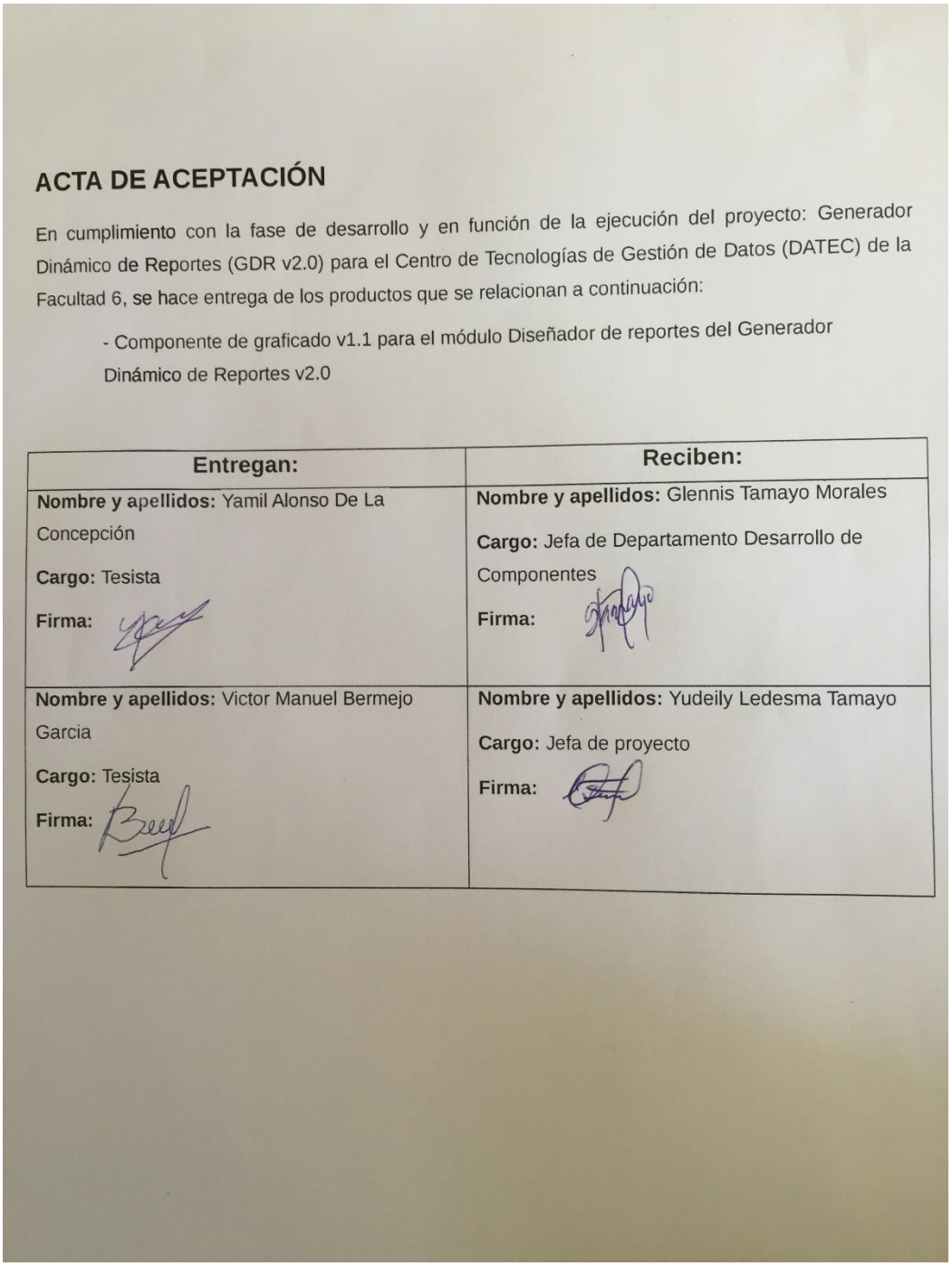

1. **Alvarez, Miguel Angel. 2014.** DesarrolloWeb. *DesarrolloWeb.* [En línea] 2 de enero de 2014. [Citado el: 12 de enero de 2016.] http://www.desarrolloweb.com/articulos/que-es-mvc.html.
2. **Americanbulls. 2016.** candlesticker. *candlesticker.* [En línea] 2016. [Citado el: 16 de 4 de 2016.] http://www.candlesticker.com/AboutUs.aspx?lang=es.
3. **Atlantic International University. 2016.** Open Courses. *Open Courses.* [En línea] 2016. [Citado el: 13 de 6 de 2016.] http://cursos.aiu.edu/Fundamentos%20de%20Estad%C3%ADstica/pdf/Tema%205.pdf.
4. **Copyright . 2016.** Copyright . *Copyright .* [En línea] 2016. [Citado el: 13 de 6 de 2016.] http://www.knorrigt.com/como-crear-un-grafico-de-termometro/.
5. **Definición. 2015.** Definición . *Definición .* [En línea] 2015. [Citado el: 25 de 11 de 2015.] http://definicion.mx/reporte/.
6. **Definición.de. 2008-2015.** Definicion de Reporte. *Definición .de.* [En línea] 2008-2015. [Citado el: 3 de 10 de 2015.] http://definicion.de/reporte/.
7. **DefiniciónABC. 2013.** DefiniciónABC. [En línea] 2013. http://www.definicionabc.com/general/grafico.php.
8. **Figueroa Zarraga, Ysis A. 2014.** Graficos estadisticos y matrices. *Graficos estadisticos y matrices.* [En línea] 16 de 6 de 2014. [Citado el: 14 de 6 de 2016.] http://www.monografias.com/trabajos101/graficos-estadisticos-y-matrices/graficos-estadisticos-y-matrices.shtml.
9. **Figueroa, Viridiana. 2015.** Scribd. *Scribd.* [En línea] 2015. [Citado el: 1 de 11 de 2015.] https://es.scribd.com/doc/50288837/Caracteristicas-de-PHP#scribd.
10. **GNU. 2008.** GNU. *GNU.* [En línea] 3 de noviembre de 2008. [Citado el: 20 de enero de 2016.] http://www.gnu.org/copyleft/fdl.html.
11. **Gómez, Edgar. 2014.** Que es un framework. *Que es un framework.* [En línea] 14 de 5 de 2014. [Citado el: 5 de 12 de 2015 .] http://edgargomez.es/que-es-un-framework/.
12. **Heffelfinger, David R. 2006.** Jasperreports Reporting for Java Developers. [aut. libro] David R Heffelfinger. *Jasperreports Reporting for Java Developers.* 2006.
13. **Hernández, Iliana Amabely Silva. 2003.** Generador Automático de Reportes Dinámicos. *Departamento de Computación.* [En línea] 2003. http://www.cs.cinvestav.mx/tesisgraduados/2003/resumenIlianaAma.html.
14. **Herrera, Cristhan. 2013.** AdictosAlTrabajo. *AdictosAlTrabajo.* [En línea] 2013. [Citado el: 5 de 12 de 2015.] https://www.adictosaltrabajo.com/tutoriales/ireport/.
15. **Hureta Cruz, Daniel. 2012.** SlideShare. *SlideShare.* [En línea] 19 de diciembre de 2012. [Citado el: 27 de marzo de 2016.] http://es.slideshare.net/dhuertacruz/diagrama-de-componentes-15705604.
16. **Instituto de Tecnologías Educativas. 2016.** ite Instituto de Tecnologías Educativas. *ite Instituto de Tecnologías Educativas.* [En línea] 2016. [Citado el: 13 de 6 de 2016.] http://www.ite.educacion.es/formacion/materiales/180/cd/m4_10/grficos_de_lneas.html.
17. **Jason. 2009.** Support. *Support.* [En línea] 2 de marzo de 2009. [Citado el: 28 de marzo de 2016.] http://support.scribd.com/forums/33939/entries/25580.

18. **Krall, Cesar. 2016.** aprenderaprogramar.com. *aprenderaprogramar.com.* [En línea] 2016. [Citado el: 2 de 5 de 2016.] http://aprenderaprogramar.com/index.php?option=com_content&view=article&id=688:ique-es-y-para-que-sirve-uml-versiones-de-uml-lenguaje-unificado-de-modelado-tipos-de-diagramas-uml&catid=46:lenguajes-y-entornos&Itemid=163.
19. **Lamarca Lapuente, María Jesús. 2013.** Hipertexto, el nuevo concepto de documento en la cultura de la imagen. *Hipertexto, el nuevo concepto de documento en la cultura de la imagen.* [En línea] 8 de 12 de 2013. [Citado el: 20 de 2 de 2016.] http://www.hipertexto.info/documentos/xml.
20. **Larman, Craig . 2003.** UPM. *UPM.* [En línea] 5 de abril de 2003. [Citado el: 30 de marzo de 2016.] http://is.ls.fi.upm.es/docencia/is2/documentacion/ModeloDominio.pdf.
21. **LibrosWeb.es. 2015.** LibrosWeb.es. *LibrosWeb.es.* [En línea] 25 de 5 de 2015. [Citado el: 20 de 1 de 2016.] http://librosweb.es/libro/symfony_1_4/capitulo_1/symfony_en_pocas_palabras.html.
22. **Lima. 2009.** *GUÍA PARA LA PRESENTACIÓN DE GRÁFICOS ESTADÍSTICOS.* Perú : Centro de Investigación y Desarrollo, 2009.
23. **LinkedIn, Corporation. 2014.** LinkedIn Corporation. [En línea] 26 de marzo de 2014. [Citado el: 20 de abril de 2016.] https://www.linkedin.com/legal/copyright-policy.
24. **Martínez, Andrés G. 2015.** SPSS FREE. *SPSS FREE.* [En línea] 2015. [Citado el: 13 de 6 de 2016.] http://www.spssfree.com/curso-de-spss/graficos-en-spss/grafico-de-sectores-en-spss.html.
25. **Microsoft. 2016.** Presentar datos en un gráfico de áreas. *Presentar datos en un gráfico de áreas.* [En línea] 2016. [Citado el: 14 de 6 de 2016.] https://support.office.com/es-es/article/Presentar-datos-en-un-gr%C3%A1fico-de-%C3%A1reas-f4842b1c-a29b-4766-be07-3b61d2e77d39.
26. **—. 2016.** Presentar los datos en un gráfico circular. *Presentar los datos en un gráfico circular.* [En línea] 2016. [Citado el: 14 de 6 de 2016.] https://support.office.com/es-es/article/Presentar-los-datos-en-un-gr%C3%A1fico-circular-1a5f08ae-ba40-46f2-9ed0-ff84873b7863.
27. **—. 2016.** Presentar los datos en un gráfico de burbuja. *Presentar los datos en un gráfico de burbuja.* [En línea] 2016. [Citado el: 13 de 6 de 2016.] https://support.office.com/es-es/article/Presentar-los-datos-en-un-gr%C3%A1fico-de-burbuja-424d7bda-93e8-4983-9b51-c766f3e330d9.
28. **—. 2016.** Presentar los datos en un gráfico de columnas. *Presentar los datos en un gráfico de columnas.* [En línea] 2016. [Citado el: 14 de 6 de 2016.] https://support.office.com/es-es/article/Presentar-los-datos-en-un-gr%C3%A1fico-de-columnas-d89050ba-e6b6-47de-b090-e9ab353c4c00.
29. **Montes Oliver, Keimer, Hernández Cervantes, Beatriz y Mendoza Garnache, Alberto. 2015.** Laccei. *Laccei.* [En línea] 21 de julio de 2015. http://www.laccei.org/LACCEI2015-SantoDomingo/RefereedPapers/RP034.pdf.
30. **Nobelo Can, Carolina. 2010.** scribd. *scribd.* [En línea] 6 de septiembre de 2010. [Citado el: 19 de febrero de 2016.] https://es.scribd.com/doc/37187866/Requerimientos-funcionales-y-no-funcionales#scribd.
31. **Noreña Ossa, Andrés. 2011.** Vitrina EIA Repositorio Institucional. *Vitrina EIA Repositorio Institucional.* [En línea] 2011. [Citado el: 13 de 6 de 2016.] http://repository.eia.edu.co/bitstream/11190/1097/1/ADMO0662.pdf.

32. **OpenOffice. 2015.** Tipos de gráficos en OpenOffice Calc. *Tipos de gráficos en OpenOffice Calc.* [En línea] 26 de 8 de 2015. [Citado el: 14 de 6 de 2016.] http://wiki.openoffice.es/Tipos_de_graficos_en_OpenOffice_Calc.
33. **Oracle. 2016.** NetBeans. *NetBeans.* [En línea] 29 de enero de 2016. [Citado el: 28 de marzo de 2016.] https://netbeans.org/community/releases/81/.
34. **Pillou, Jean Francisco. 2013.** CCM. *CCM.* [En línea] 3 de abril de 2013. [Citado el: 20 de febrero de 2016.] http://es.ccm.net/contents/304-lenguajes-de-programacion .
35. **Pressman, Roger S. 2011.** *Ingenieria De Software Un enfoque práctico.* Madrid : McWragHill, 2011.
36. **Quees. 2015.** Quees.la. *Quees.la.* [En línea] 2015. [Citado el: 25 de 11 de 2015.] http://quees.la/reporte/.
37. **SlideShare. 2013.** SlideShare. *SlideShare.* [En línea] 27 de febrero de 2013. [Citado el: 5 de diciembre de 2015.]
38. **SOMMERVILLE, IAN. 2005.** *Ingeniería del Software Séptima edición.* Madrid : Miguel Martín-Romo, 2005.
39. **Sparck. 2007.** SparckSystem. *SparckSystem.* [En línea] 2 de febrero de 2007. [Citado el: 25 de enero de 2016.] http://www.sparxsystems.com.ar/resources/tutorial/uml2_deploymentdiagram.html.
40. **Universidad de Sonora. 2015.** UNISON. *UNISON.* [En línea] 2015. [Citado el: 14 de 6 de 2016.] http://www.estadistica.mat.uson.mx/Material/seriesdetiempo.pdf.
41. **Universo Formulas. 2015.** Universo Formulas. *Universo Formulas.* [En línea] 2015. [Citado el: 5 de 3 de 2016.] http://www.universoformulas.com/estadistica/descriptiva/diagrama-barras/.
42. **Vazquez Ocampo, Gerardo. 2013.** SlideShare. *SlideShare.* [En línea] 8 de 3 de 2013. [Citado el: 25 de 10 de 2015.] http://es.slideshare.net/AUREA23/reporte-escrito.
43. **Venete, Adriana. 2011.** Ptrones Arquitectura. *Ptrones Arquitectura.* [En línea] 10 de enero de 2011. [Citado el: 15 de febrero de 2016.] https://www.google.com.cu/url?sa=t&rct=j&q=&esrc=s&source=web&cd=5&ved=0ahUKEwim_4Dbr5HLAhUCbj4KHUw4CZUQFggzMAQ&url=http%3A%2F%2Fmahara.uji.es%2Fartefact%2Ffile%2Fdownload.php%3Ffile%3D54534%26view%3D4648&usg=AFQjCNGV0SXl7IX8wf6-q0sGmoyBVuPYoQ&bvm=bv.1152.
44. **Vera, Claudia. 2015.** Universidad Santo Tomas. *Universidad Santo Tomas.* [En línea] 2015. [Citado el: 23 de 11 de 2015.] http://soda.ustadistancia.edu.co/enlinea/claudiaveraInformatica%20I_Momento%20III/elementos_de_un_grfico.html.
45. **Yasen, Noemé . 2014.** SPSS FREE. *SPSS FREE.* [En línea] 2014. [Citado el: 14 de 6 de 2016.] http://face.unt.edu.ar/web/iadmin/wp-content/uploads/sites/2/2014/12/Aplicaci%C3%B3n-pr%C3%A1ctica-Diagrama-de-Gantt-para-Jornada-IA-Handl.pdf.

# Anexos

**Anexo 1:** Carta de aceptación del cliente al producto desarrollado.



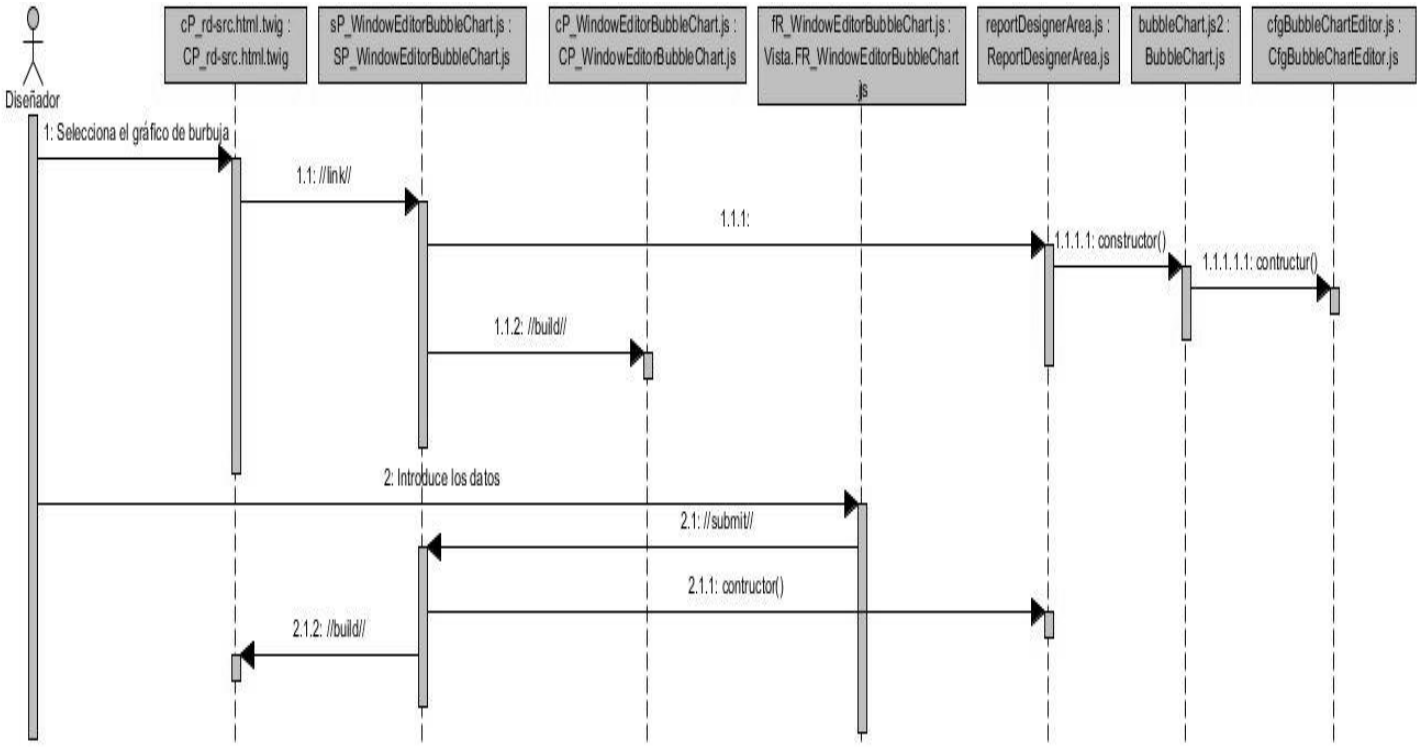
## ACTA DE ACEPTACIÓN

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

- Componente de graficado v1.1 para el módulo Diseñador de reportes del Generador Dinámico de Reportes v2.0

| Entregan: | Reciben: |
|---|---|
| **Nombre y apellidos:** Yamil Alonso De La Concepción<br>**Cargo:** Tesista<br>**Firma:** | **Nombre y apellidos:** Glennis Tamayo Morales<br>**Cargo:** Jefa de Departamento Desarrollo de Componentes<br>**Firma:** |
| **Nombre y apellidos:** Victor Manuel Bermejo Garcia<br>**Cargo:** Tesista<br>**Firma:** | **Nombre y apellidos:** Yudeily Ledesma Tamayo<br>**Cargo:** Jefa de proyecto<br>**Firma:** |

**Anexo 2:** Diagrama de secuencia correspondientes al caso de uso Gestionar gráfico de burbujas responde al requisito funcional Añadir gráfico de burbujas.




**Anexo 3:** Entrevista realizada a los especialistas del proyecto GDR v2.0

Preguntas realizadas al cliente como parte de la entrevista abierta, con el objetivo de recopilar información referente a la realización del componente de graficado v1.1:

1. ¿Cuáles son las principales deficiencias que existen en el componente de graficado v1.0?
2. ¿Cómo se generan las gráficas de los reportes en el componente de graficado v1.0?
3. ¿Cuáles son las principales funcionalidades que deben ser implementadas en el componente de graficado v1.1?
4. ¿Cuáles son los requerimientos no funcionales para el desarrollo de componente de graficado v1.1?
5. ¿Puede describir el ambiente en el que se usará la solución?
6. ¿Qué metodología, herramientas y tecnologías se utilizan en el desarrollo de GDR v2.0?